AUTODESK Instructables

Raspberry Pi Planet Finder

By snowbiscuit in CircuitsRaspberry Pi

Introduction: Raspberry Pi Planet Finder

Outside the Science Centre in my city there is a large metal structure which could turn and point at where the planets were in the sky. I never saw it working, but I always thought it would be magical to know where these unreachable other worlds actually were in relation to my tiny self.

When I walked past this long-dead exhibit recently I thought "I bet I could make that" and so I did!

This is a guide on how to make the Planet Finder (featuring the Moon) so you too can know where to look when you're feeling awed by space.

Step 1: What You Need

1 x Raspberry Pi (version 3 or higher for onboard wifi)

1 x LCD screen (16 x 2) (like this)

2 x Stepper motors with drivers (28-BYJ48) (like these)

3 x Push Buttons (like these)

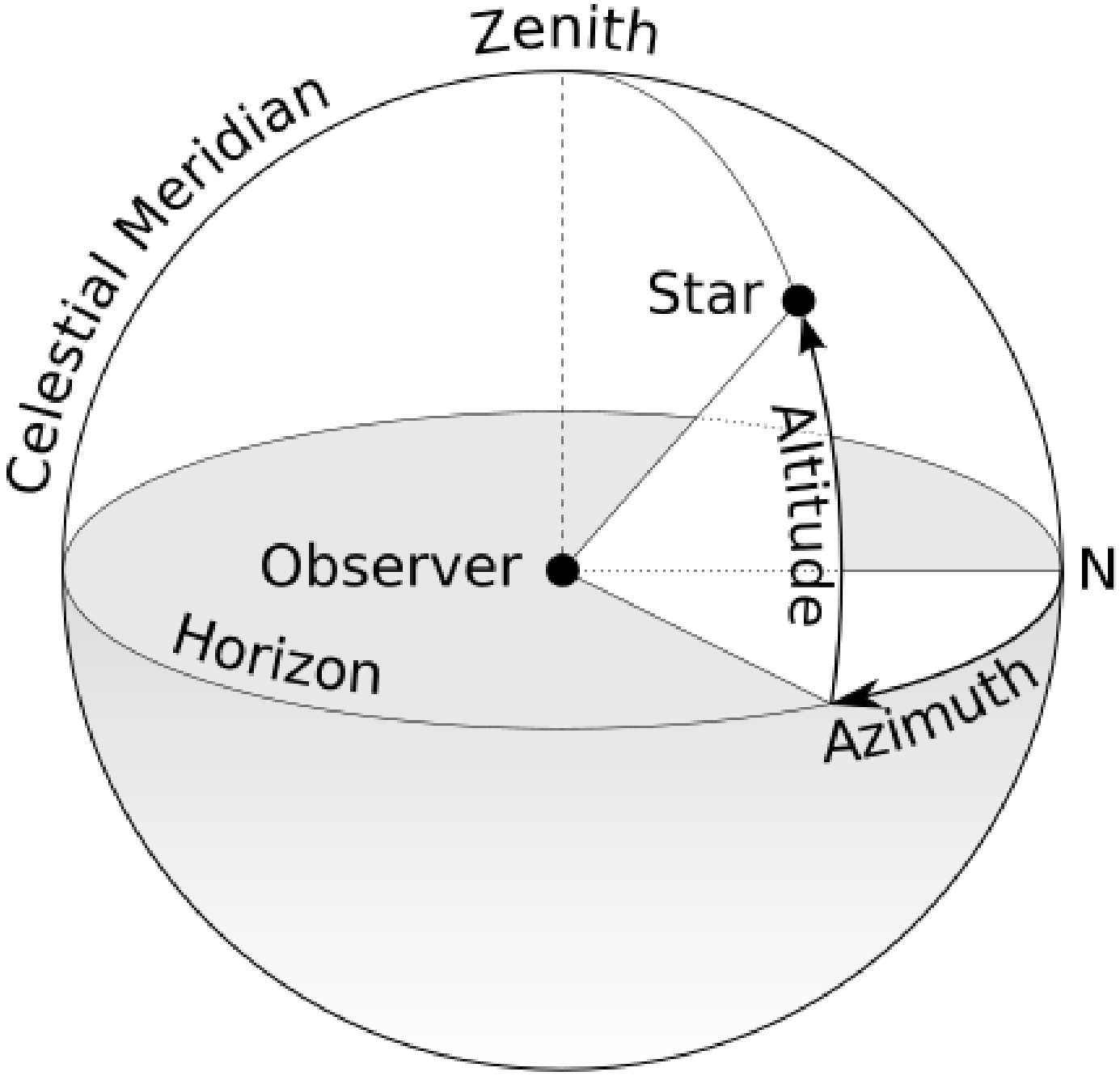
2 x Flange Couplers (like these)

1 x Button compass (like this)

8 x M3 bolts and nuts

3D printed parts for the case and telescope

Step 2: Planetary Coordinates



There are a few different ways of describing where astronomical objects are in the sky.

For us, the one which makes the most sense to use is the Horizontal Coordinate System as shown in the image above. This image is from The Wikipedia page linked here:

https://en.wikipedia.org/wiki/Horizontal_coordinat...

The Horizontal Coordinate system gives you an angle from North (the Azimuth) and upwards from the horizon (the Altitude), so it is different depending on where you are looking from in the world. So our planet finder needs to take location into account and have some way of finding North to be a reference.

Rather than try to calculate the Altitude and Azimuth which change with time and location, we will be using the wifi connection on board the Raspberry Pi to look up this data from NASA. They keep track of this sort of thing so we don't have to ;)

Step 3: Accessing Planet Data

We are getting our data from the NASA Jet Propulsion Laboratory (JPL) - https://ssd.jpl.nasa.gov/?horizons

To access this data, we use a library called AstroQuery which is a set of tools for querying astronomical web forms and databases. The documentation for this library is found here: https://astroquery.readthedocs.io/en/latest/jplhor...

If this is your first Raspberry Pi project, start by following this set up guide: https://projects.raspberrypi.org/en/projects/raspb...

If you're using Raspbian on your Raspberry Pi (you will be if you followed the guide above), then you already have python3 installed, make sure you have the most recent version installed (I'm using version 3.7.3). We need to use this to get pip. Open a terminal and type the following:

```
sudo apt install python3-pip
```

We can then use pip to install the upgraded version of astroquery.

```
pip3 install --pre --upgrade astroquery
```

Before continuing with the rest of of this project, try accessing this data with a simple Python script to make sure all the right dependencies have been installed correctly.

```
from astroquery.jplhorizons import Horizons

mars = Horizons(id=499, location='000', epochs=None, id_type='majorbody')

eph = mars.ephemerides()

print(eph)
```

This should show you the details of the location of Mars!

You can check to see if this data is right using this site to look up live planet positions: https://theskylive.com/planetarium

To break this query down a bit, the id is the number associated with Mars in JPL's data, epochs is the time we want the data from (None means right now) and id_type is asking for the major bodies of the solar system. The location is currently set to the UK as '000' is the location code for the observatory in Greenwich. Other locations can be found here: https://minorplanetcenter.net//iau/lists/ObsCodesF...

Troubleshooting:

If you get the error: No module named 'keyring.util.escape'

try the following command in the terminal:

```
pip3 install --upgrade keyrings.alt
```

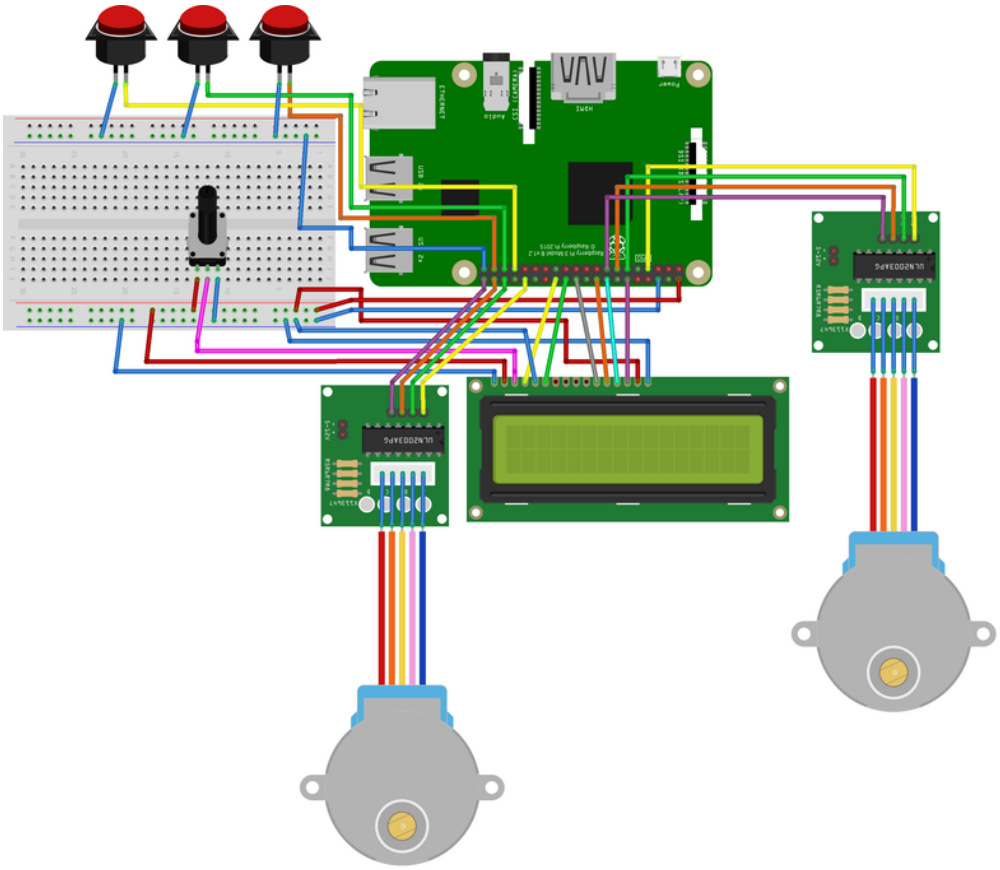
Step 4: Code

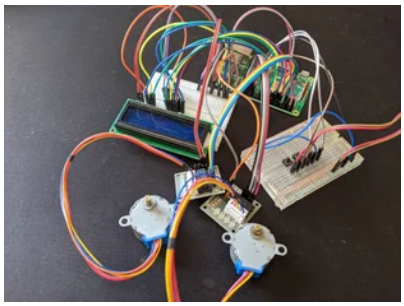
Attached to this step is the full python script used in this project.

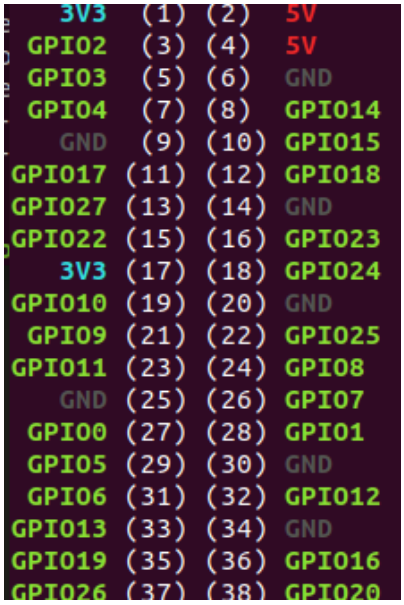
To find the correct data for your location, go to the function getPlanetInfo and change the location using the list of observatories in the previous step.

```
def getPlanetInfo(planet):
        obj = Horizons(id=planet, location='000', epochs=None, id_type='majorbody')
        eph = obj.ephemerides()
        return eph
```

Step 5: Connecting Hardware







Using breadboards and jumper wires, connect up two stepper motors, the LCD screen and three buttons as shown in the circuit diagram above.

To find out what number the pins are on your Raspberry Pi, go to the terminal and type

```
pinout
```

This should show you the image above complete with GPIO numbers and board numbers. We are using board numbers to define which pins are used in the code, so I will be referencing the numbers in brackets.

As an aid to the circuit diagram, here are the pins which are connected to each part:

1st Stepper motor - 7, 11, 13, 15

2nd Stepper motor - 40, 38, 36, 32

Button1 - 33

Button2 - 37

Button3 - 35

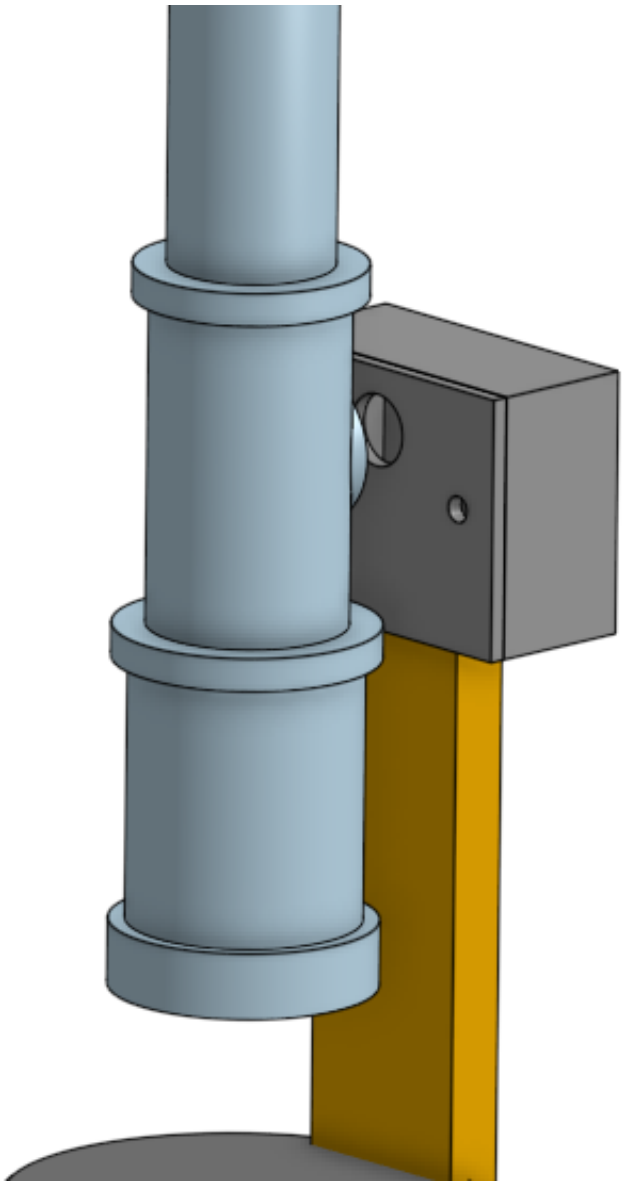
LCD screen - 26, 24, 22, 18, 16, 12

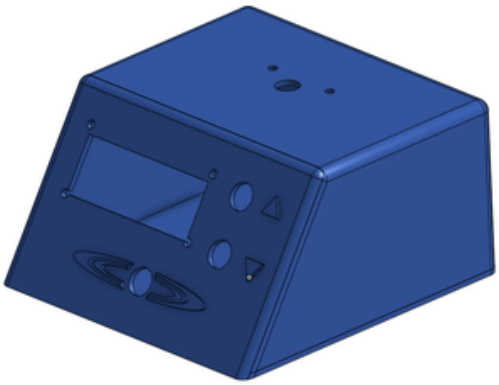
When this is all connected, run the python script

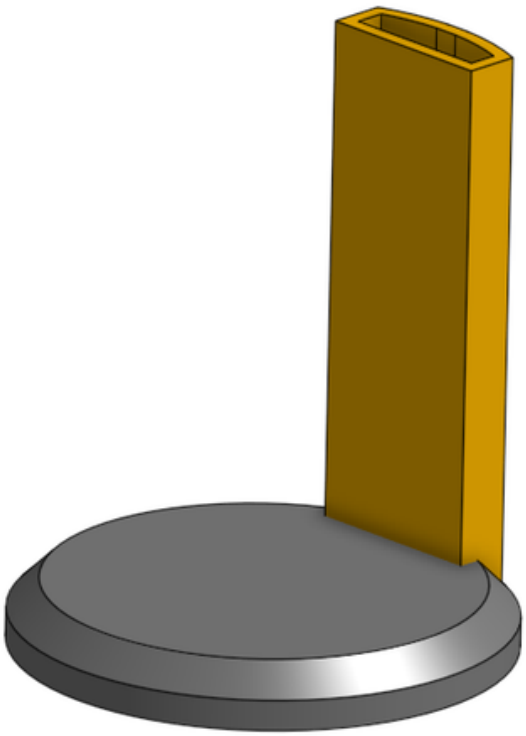
```
python3 planetFinder.py
```

and you should see the screen show setup text and the buttons should move the stepper motors.

Step 6: Designing the Case







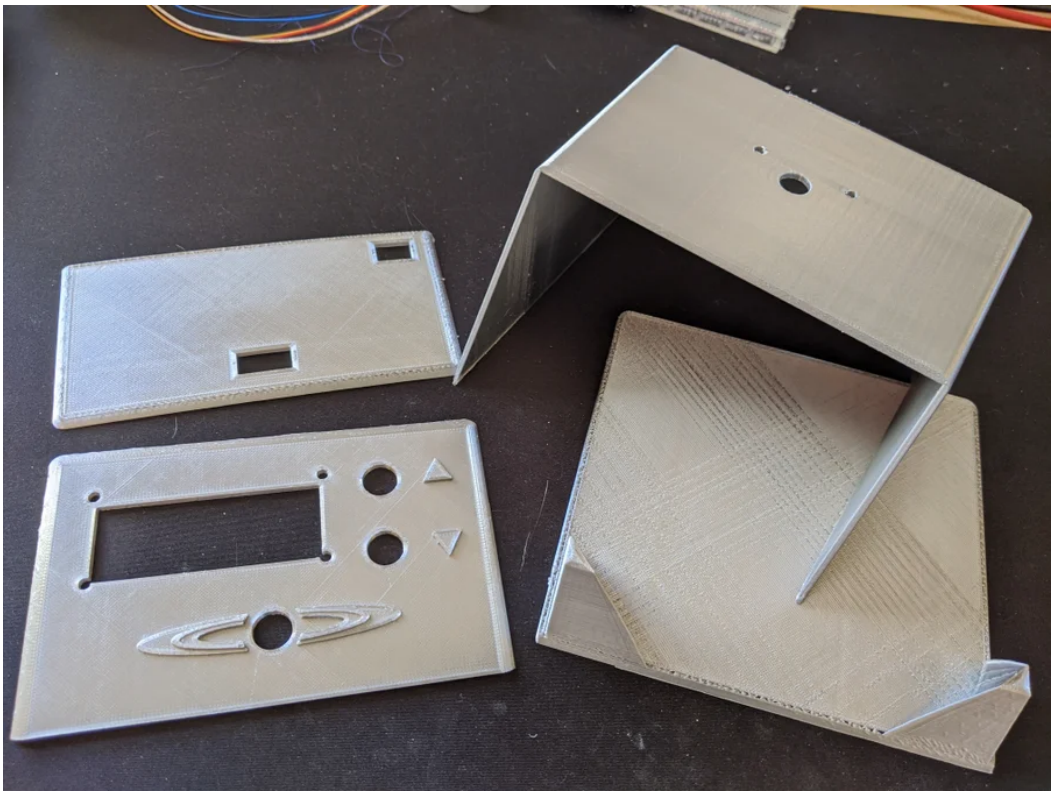
The case was designed to be 3D printed easily. It breaks down into separate parts which are then glued together once the electronics have been secured in place.

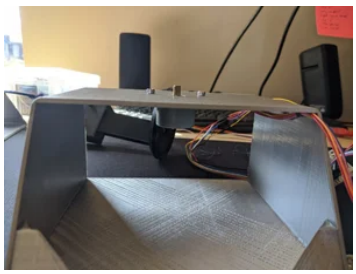
Holes are sized for the buttons I used and M3 bolts.

I printed the telescope in parts and glued them together later to avoid too much support structure.

STL files are attached to this step.

Step 7: Testing the Prints

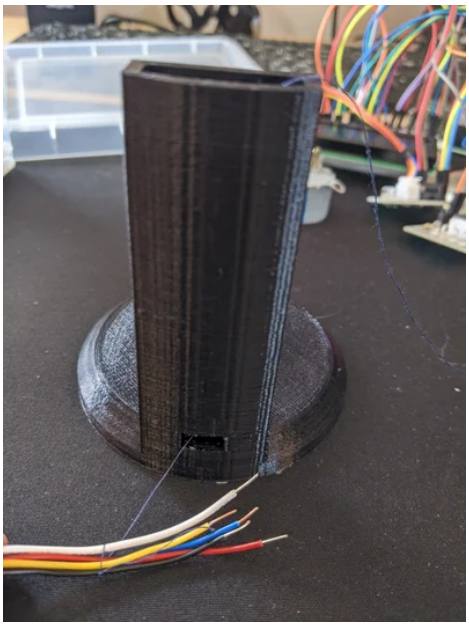
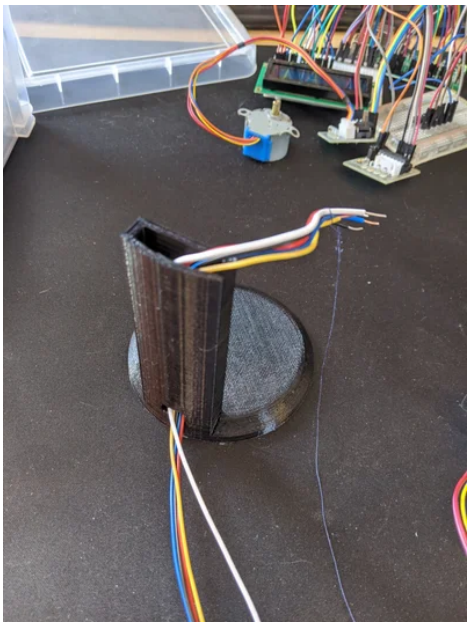
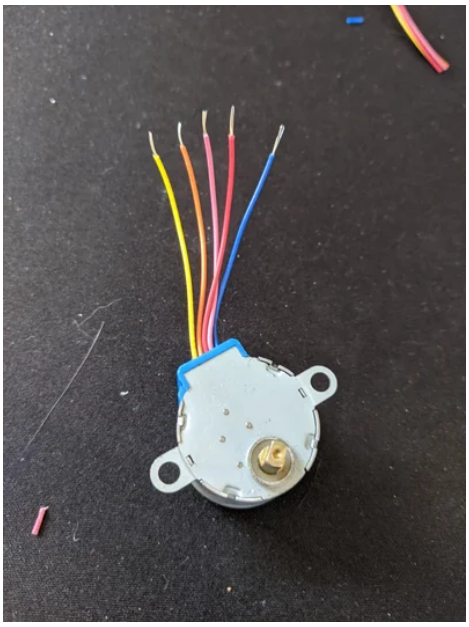




Once everything is printed, make sure everything fits snugly together before any glueing is done.

Fit the buttons in place and secure the screen and stepper motors with M3 bolts and give everything a good wiggle. File down any rough edges take everything apart again before the next step.

Step 8: Extending the Stepper Motor

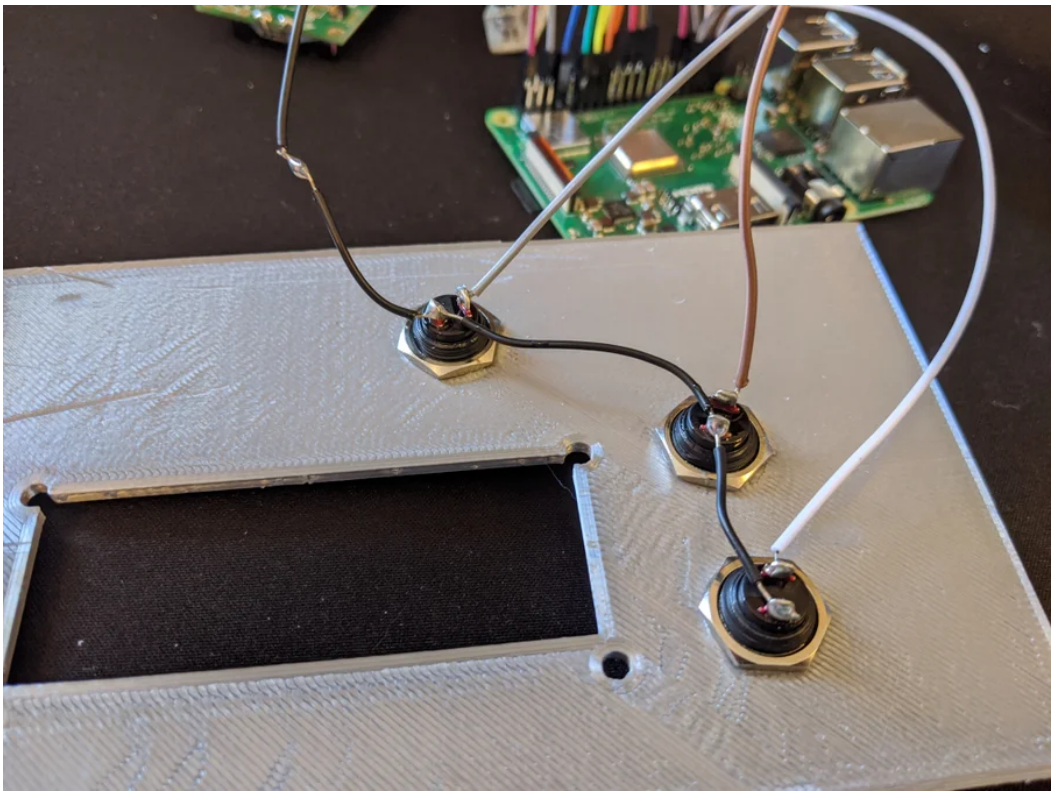




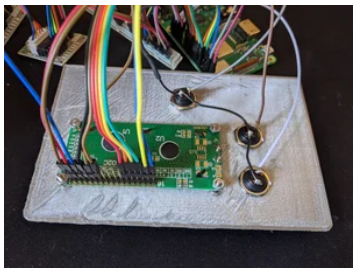
The stepper motor which will control the elevation angle of the telescope will sit above the main case and needs some slack in the wires in order to rotate. The wires need to be extended by cutting them between the stepper and it's driver board and soldering a new length of wire in between.

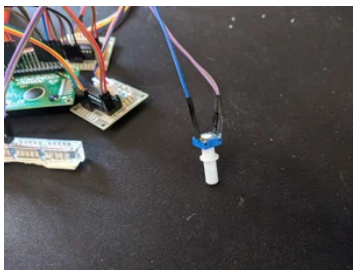
I inserted the new wire into the supporting tower using a piece of thread to help coax it through as the wire I'm using is quite stiff and kept getting stuck. Once through it can be soldered to the stepper motor, making sure to keep track of which colour is connected in order to reattach the right ones at the other end. Don't forget to add heat shrink to the wires!

Once soldered, run the python script to check everything is still working, then push the wires back down the tube until the stepper motor is in position. It can then be attached to the stepper motor housing with M3 bolts and nuts before the back of the housing is glued in place.

Step 9: Mount Buttons and LCD Screen





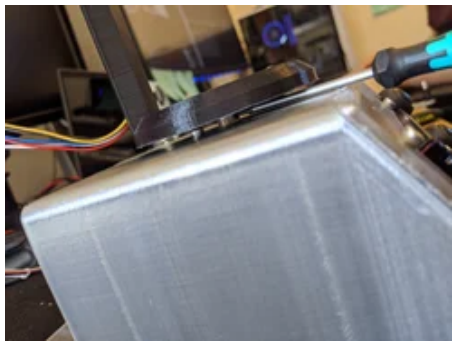


Insert the buttons and tighten the nuts to secure them in place before soldering. I like to use a common ground wire which runs between them for neatness.

Secure the LCD screen with M3 bolts and nuts. The LCD wants a potentiometer on one of it's pins which I also soldered in a this stage.

Test the code again! Make sure everything is still working before we glue everything together as it's much easier to fix at this stage.

Step 10: Adding Flanges



To connect the 3D printed parts to the stepper motors, we are using a 5mm flange coupling which fits on top of the end of the stepper motor and is held in place by tiny screws.

One flange is glued to the base of the rotating tower and the other to the telescope.

Attaching the telescope to the motor on top of the rotating tower is simple as there is lots of space to access the small screws holding it in place. The other flange is harder to secure, but there is enough of a gap between the main case and the base of the rotating tower to fit a small allen key and tighten the screw.

Test again!

Now everything should be working as it will be in its final state. If it's not, now is the time to bug fix and make sure connections are all secure. Make sure exposed wires are not touching each other, go round with electrical tape and patch up any places which could cause a problem.

Step 11: Run on Startup

Rather than run the code manually every time we want to find a planet, we want this to run as a stand alone exhibit, so we're going to set it up to run our code whenever the Raspberry Pi turns on.

In the terminal, type

```
crontab -e
```

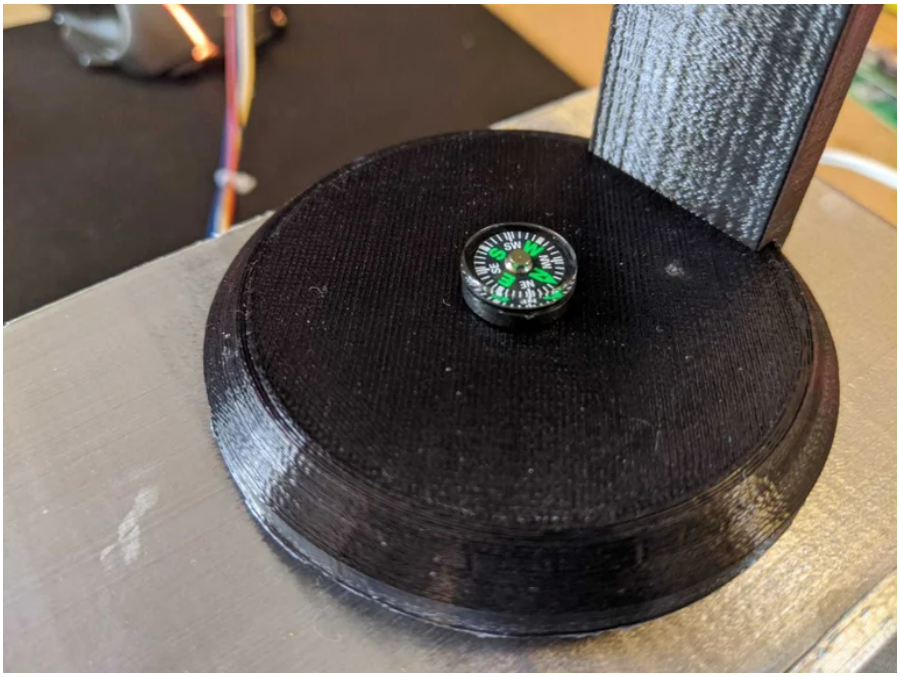
In the file which opens, add the following to the end of the file, followed by a new line.

```
@reboot python3 /home/pi/PlanetFinder/planetFinder.py &
```

I have my code saved in a folder called PlanetFinder, so /home/pi/PlanetFinder/planetFinder.py is the location of my file. If yours is saved somewhere else make sure to change it here.

The & at the end is important as it lets the code run in the background, so it doesn't hold up other processes which also happen in boot.

Step 12: Glue It All Together!



Everything that isn't already glued in place should now be fixed in.

Finally, add the tiny compass to the middle of the rotating base.

Step 13: Usage

When the Planet Finder turns on, it will prompt the user to adjust the vertical axis. Pressing the up and down buttons will move the telescope, try and get it to be level, pointing to the right, then press the ok button (at the bottom).

The user will then be asked to adjust the rotation, use the buttons to spin the telescope until it points North according to the small compass, then press ok.

You can now cycle through planets using the up/down buttons and select one you would like to find with the ok button. It will display the Altitude and Azimuth of the planet then go and point at it for a few seconds before turning back to face North.

Step 14: Finished

All done!

Enjoy knowing where all of the planets are :)