**SSA\_SST Next Steps 17.11.2022**

Tracker and Allsky:

1. Tracker (electronics etc.)

I didn't have time to complete the tracker, I got it running for testing using a raspberry pi but I had to remove the parts again because they were just borrowed. So the tracker is missing two DRV8825, and an Arduino or Raspberry Pi for control, as well as hall-effect sensors and limit switches. You can find the project and parts list on this thingiverse project: <https://www.thingiverse.com/thing:4316563>

I would prefer a Raspberry pi, because in the end both cameras will be controlled by the raspberry pi and the calculations can be done on it.

The limit switches are very important because otherwise you can damage the cameras because the lenses run into the tracker itself or the cables tangle.

Very important is that all the moving parts of the tracker need to be lubricated either by oil or graphite, otherwise the plastic parts will damage quickly.

1. Cameras

The Tracker has no camera: There are two options I mentioned in my report. Either a more expensive Camera like the Sony Alpha 7 III or something like the raspberry pi camera. And either camera would need a lense with magnification and best would be to fit the camera with an electrically controlled magnification. So the magnification can be adjusted to the angular speed of the satellite.

For the Sony you need a camera mounting screw, as well mentioned in the thingiverse projects parts list. And a Dummy battery.

For the Raspberry pi camera you will need to print a mount that can hold the sensor with the lense.

After installing you need to connect the camera to the raspberry pi, it is necessary that you can control the camera from the pi and be able to adjust the iso from the code. The Iso is important for the filter system.

1. Allsky

Generally in its current configuration the allsky camera isn't perfect for satellite detection, either; it needs to be reconfigured so that the whole sensor is used instead of cropping it to a round image and I would reduce the camera's viewing angle if possible.

The battery of the allsky camera is replaced but a dummy battery would still be the best option.

The Allsky cameras laptop needs to be connected to the camera.

The sun blocker (ELCIM) has an axle that needs to be replaced because it has some play that causes the sun blocker to be at the wrong height.

1. Allsky and Tracker

In the end both cameras will need to work together so both cameras and the tracker need to be controlled by the raspberry pi.

1. Testing of the Tracker

After installing all the electronics and

before working on the software of the tracker I would recommend testing its accuracy over a longer period of time, to see at what speeds the motors can run. If the mounted camera is heavy the stepper motors may miss steps leading to inaccuracy.

Software and Code:

The Software is currently still in a development state and I will upload explanations to each code to the Teams folder “SSA\_SST Detection , Niels”, mainly explaining why certain things are the way they are. There is a file, in which I have documented every upgrade I have done to the code.

1. Calibration

For calibration I used a system that would transform image coordinates to sky coordinates using the distance from the object in the image to the position of the zenith in the image for calculating the elevation. And using the angle between the vector (zenith-object) and the north vector (the vector between the zenith and a point that is exactly at 0° Azimuth) to calculate the Azimuth of the object.

This system is really simple and only needs a few input stars, but it has some misalignment in the corners of the image. This error is not caused by the calculation of the distortion but by something else. You can read about this alignment method and the error in the development script in “2.c.4”. There is a different way of transforming the coordinates, this is by using a 2d-fit, mentioned in “2.c.3”.

For the Project I wrote a simple api that would allow me to quickly find the calibration parameters of images, it is based on the method mentioned in the beginning. Generally having an api that allows for fast calibration of the camera is good but it probably needs to be upgraded too.

1. Satellite Positioning

When positioning a satellite in the sky you can either use the background stars or use the coordinate transformation. Generally what I mean with positioning is calculating the sky coordinates of the satellite trace (Azimuth and elevation).

The first way is by simply using the method that you use to transform image coordinates to sky coordinates. If the calibration is very precise this method is definitely good enough to find the sky coordinates of the satellite.( I mean by calibration the distortion parameters, as well as the zenith position in the sky and the direction the camera is pointing). This method works for stationary cameras but not for the satellite tracker because it is always looking somewhere else (heading direction is changing). You can read about this in “4. 2” of the report.

Theoretically using the background stars for stationary cameras is very precise, because you have lots of reference positions around the satellite you can use to find its exact sky coordinates. I tried to implement this positioning but got quite high errors sometimes and I did not find the solution. You can read about this in “2.f.1”.

1. Filter System

I tested two filter systems (2.e.1 in the development script or 3.3 in the report), both of them have different applications. One can detect satellites or similar objects with very high certainty but cant detect weak satellites. The second filter system will detect very weak satellites but has a lot of false detections that need to be filtered in a second iteration.

The second filter is heavily affected by the overall brightness of the images, that's the reason why something needs to be implemented so that it will be adjusted accordingly to the brightness of the image. Generally this filter was just a test of what is possible with digital filters and I would not recommend implementing something with such sensitive detection.