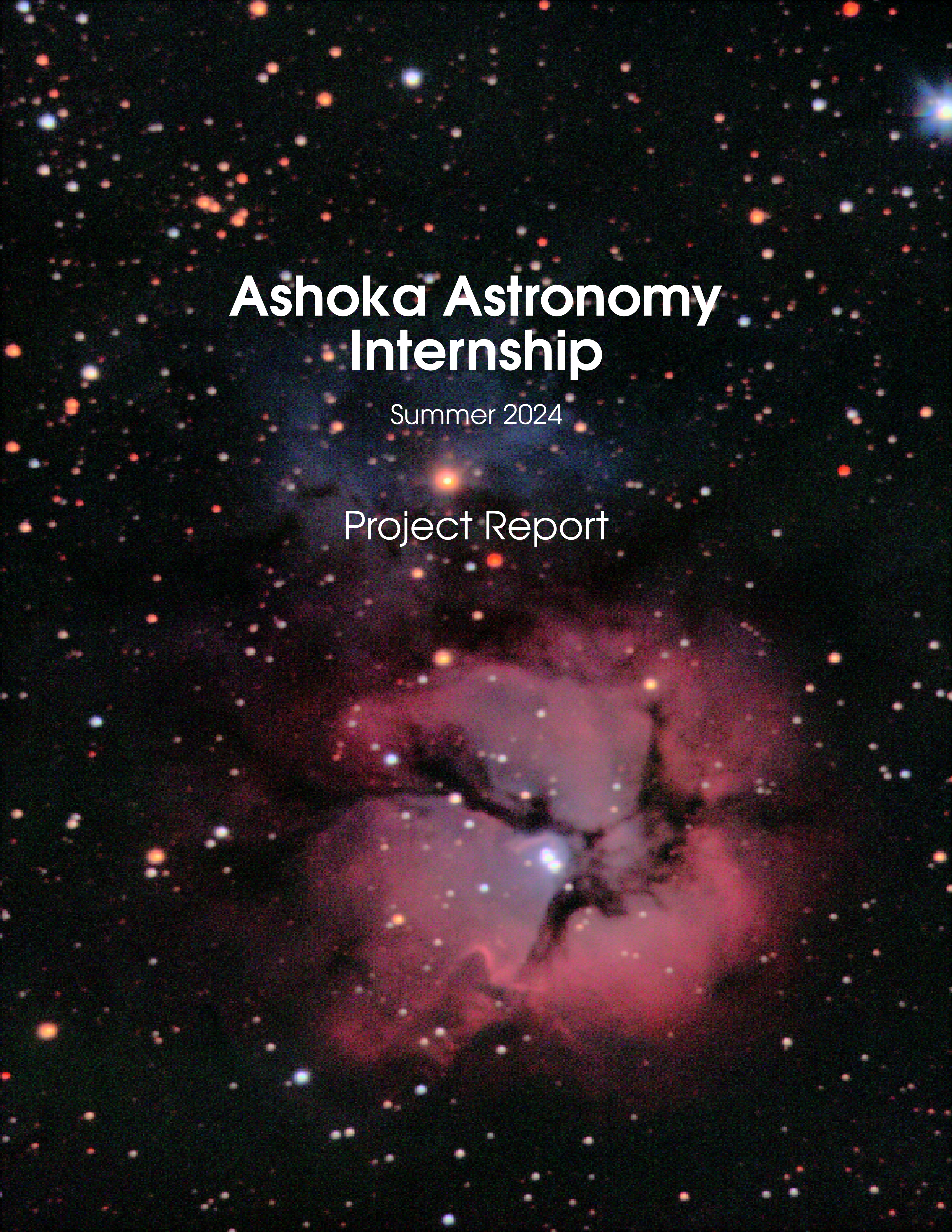
# Ashoka Astronomy Internship

Summer 2024

## Project Report



SUMMER RESEARCH INTERNSHIP, ASHOKA UNIVERSITY

GITHUB.COM/

This research was done under the supervision of Professor Dipankar Bhattacharya and Philip Cherian for a total of 10 weeks, from June 17th to August 26th, 2024.

*First release, September 2024*

# Contents

# Chapter 1

# Sun Spots

## 1.1 Introduction

A sunspot is an area on the surface of the sun with a lower temperature compared to the surrounding area. As the name suggests, they appear as ‘dark ’ spots on the sun’s surface as seen in Fig.1.1.

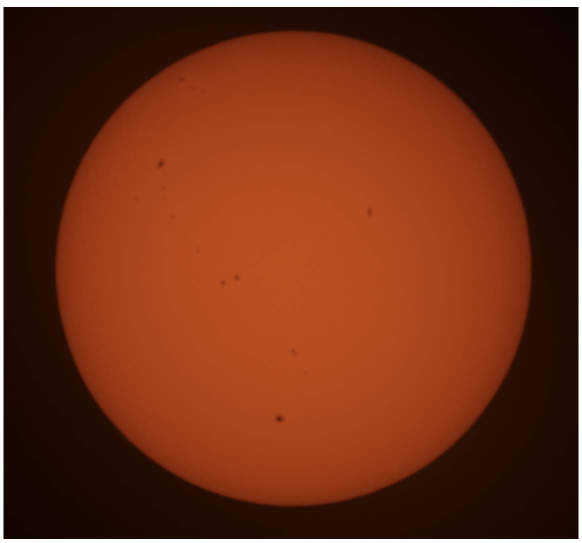


**Figure 1.1:** On the sun’s surface, multiple prominent dark spots are visible. These are sunspots. This image was taken using the Celestron Nexstar 8SE telescope with Mylar filter to reduce the intensity of incoming light, and the Nikon DS650 to photograph the the sun and the spots.

Since the temperature of the spots is lower than the rest of the sun, the temperature difference creates a contrast making these spots appear darker in our images. In this experiment, we primarily tracked the movement of these sunspots over several days. One of the advantages of conducting this experiment now is that the sun is approaching the end of its 11-year-long solar cycle and has an increased number of sunspots as shown in. Fig (1.2). An increased number of sunspots also gives us a higher number of 'data points' to track and observe. Tracking their movement can help us study the rotation of the sun on different latitudes.

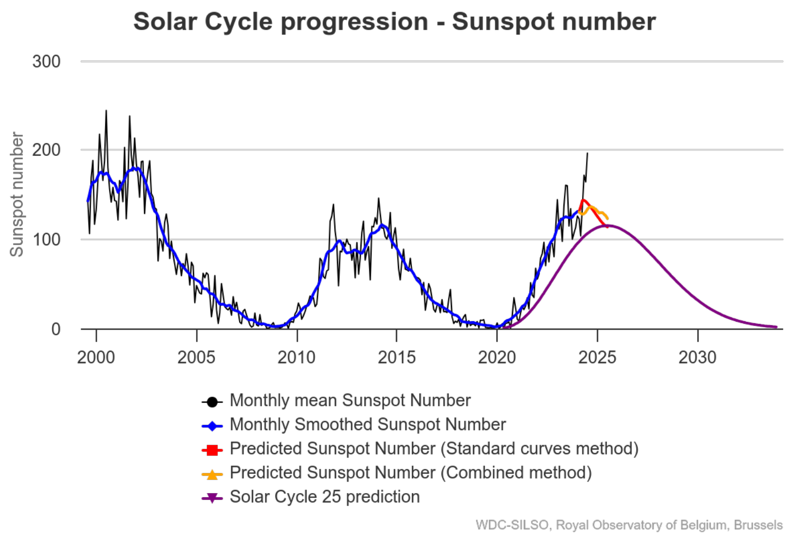



**Figure 1.2:** In this graph we can see an example where the number of sunspots increased from around 2010 (the start of the solar cycle) and peaked around 2015. Furthermore, the graph begins to dip again showing the reduction in the number of sunspots signifying the end of the cycle.

Another important point to note is Unlike the Earth which is a rigid body, the sun is made of gas and plasma making it a nonrigid body. A consequence of this is that different latitudes have different rotational periods.

This is something we further see in our Analysis (section 1.3) Sunspots move along latitudes and also rotate along the sun's rotational axis. Since these sunspots very rarely change latitudes, it helps us measure the rotational speed of different latitudes. As mentioned previously, the sun is a non-rigid body and has different rotational speeds at different points of its surface.

By taking regular data daily, we can track the movement of the sun over a large period.

## 1.2 Data Collection

Images of the Sun were taken using the Celestron Nexstar 8SE Telescope and a Celestron Alt/Az mount. The photographs were clicked using a Nikon DS5600. To protect the Telescope and the camera from the intense rays of the sun we used a Mylar filter on top of the telescope to block most of the incoming light, just enough that we could safely view the sun. We took images at different exposure times based on the weather conditions and time. Regular Data was taken daily whenever the sun was visible through our telescope, or whenever the skies were clear

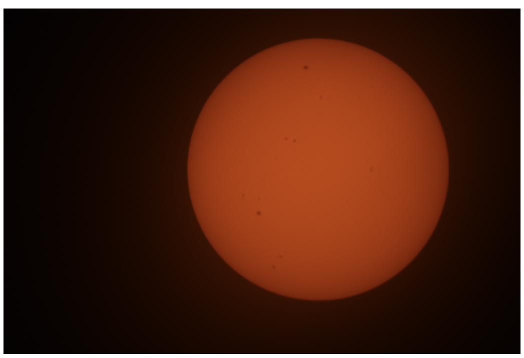
## 1.3 Analysis

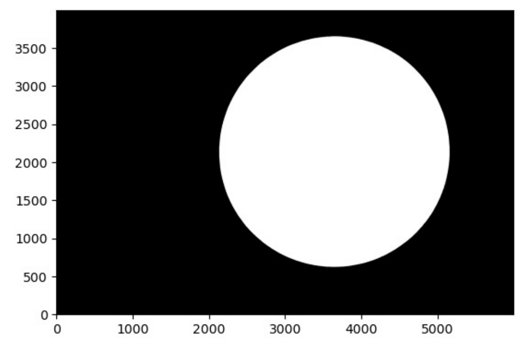
The primary Analysis was conducted using Python libraries- Numpy, Scipy, Matplotlib.pyplot and Astrolab. In our Analysis, we aimed towards being able to detect and see the spots with greater clarity as compared to the original image, and then using data over different days try to calculate the rotational period of the sunspots. We used an algorithm called Bradley Thresholding to improve the clarity of our sunspots. How we did this is further mentioned under 1.3.2. Then we focused on aligning the sun in the image such that the Solar North coincides with the Image North. This makes observing the rotation of the spots much easier. More on this has also been mentioned in 1.3.3

### 1.3.1 Thresholding

Thresholding aims to classify a certain pixel as either 'dark' or 'light'. For example, one can assign a pixel the value 'dark' if the pixel counts are lesser than a threshold value *'$p_t$'* (*simply put, the brightness*). If it were to be greater than *'$p_t$'* it could be assigned the value 'light'.

Following is an example of a naively threshold-ed image






Original Image

Image after naive thresholding *'$p_t$'*=40

**Figure 1.3:** The thresholded image is shown on the right. Since the sun is brighter than our given value of '$p_t$' there appears to be a mask created. This helps us isolate the sun from the background

The image seems to have created a "mask," where the area representing the sun has turned white, while the rest has turned black. The pixels corresponding to the sun have been assigned the value "light," while the background has been assigned "dark." This is because our image's sun is much brighter than the background.

Following is a pseudocode for a function that creates a similar *mask* that accepts an image *i*, threshold value *t*

In this code, any pixel in the image ***'i'*** that is greater than the given ***'$p_t$'*** is assigned the number 1. If it's lesser, it's assigned the number 0. As mentioned before, thresholding is to classify a pixel as dark or light. The "ones" represent the light pixels, and the "zeros" represent the dark pixels. This results in an image similar to Fig. 3.1 on the right.

Furthermore, one can change the value of '$p_t$' and see how it affects the image.

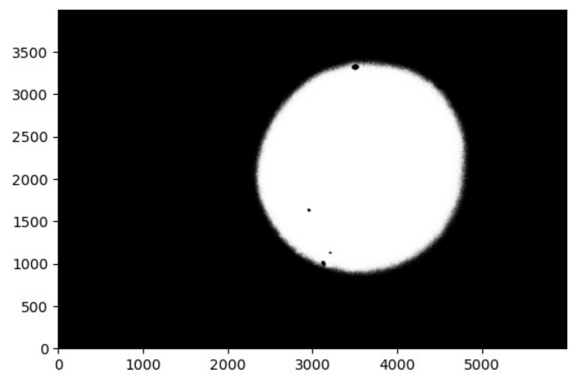
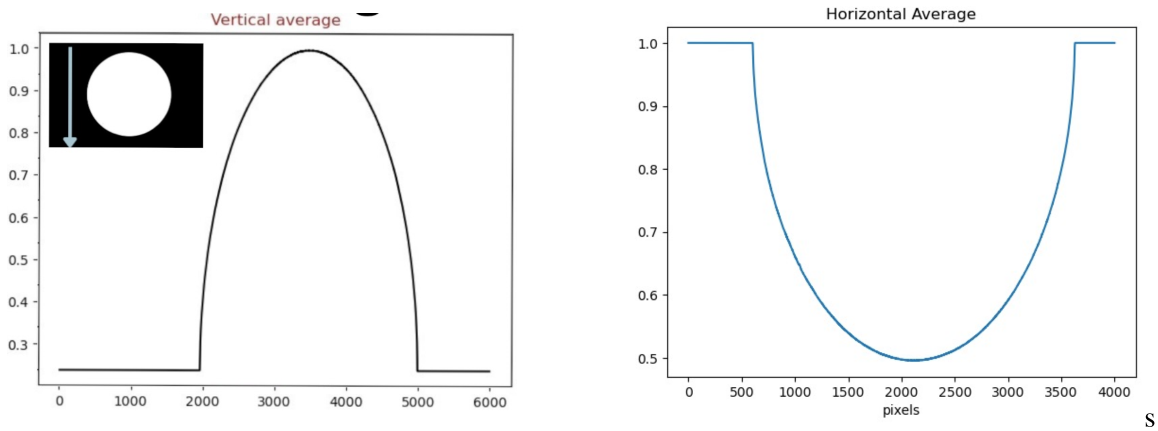



**Figure 1.4:** On changing the value of '$p_t$'to 70 spots appear on the white circle. These are the sunspots on the sun. A result of this higher $p_t$ also results in a deformation of the shape of the sun

When the threshold value is increased, a few white spots appear on the surface of the sun. These are sunspots (Fig.1.4). Since they are relatively cooler areas, the pixels encompassing these spots also have lower values than the rest of the sun. However, they are also relatively brighter than the background, which requires a higher value of '$p_t$' to be detected. While increasing the value of '$p_t$' makes these spots visible you can also see that the shape of the sun appears to be deformed.
This can make the analysis more challenging, however, it does help us essentially separate the sun from the background and, therefore find the diameter of the sun.

Since the image is primarily ones and zeros, we can find the location of the sun by doing a vertical and horizontal average. This tells us where the ones start appearing in the photo.




vertical average | horizontal average

**Figure 1.5:** Upon thresholding using a value where we can see the sun being clearly separated from the background we can take vertical and horizontal averages of the image. The peak is the center of the sun and the points where it rises from the axes and falls back down is the edge of the sun. This can help us crop the image and make further analysis easier

Since we can find the diameter of the sun in both the x and y axes (the values should be roughly equal) we can now take the same image and create a mask such that only the sun remains in the given diameter and the remaining background is given the value 0 or dark.

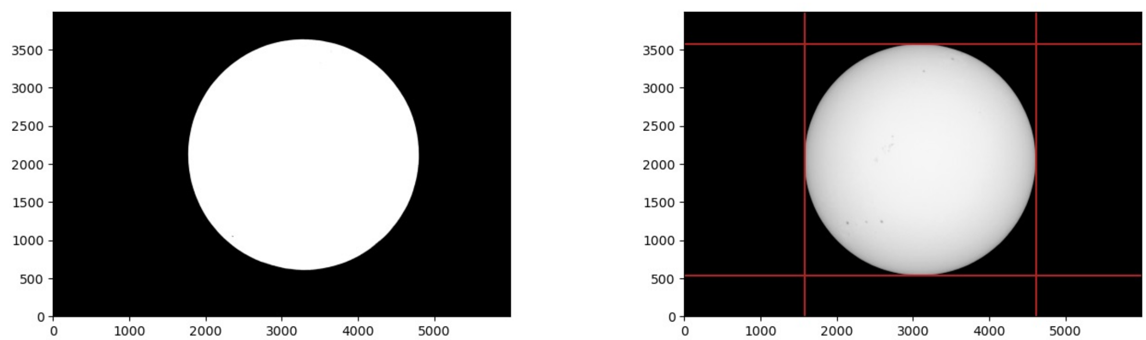


Image with mask | Image of the sun in the mask

**Figure 1.6:** The image on the left is the mask created for the image of the sun. The image on the right is the sun placed in the diameter (highlighted by red lines) we found using the horizontal and vertical averages. Note: our display function turns colored images into binary images (greyscale)

The idea and reason behind such a tedious task is primarily to reduce background noise. Since all the background values are now ‘0’, they are essentially equal. This method and its reasoning are further clarified in section- 1.3.2. Now that we know what the diameter of the sun is, we can even crop the image to exactly that value to get a better view of the sun Furthermore, even though we managed to reduce background noise, the spots themselves are not visible. Being quite faint, they would be hard to track and pinpoint. We now need to find a way to make these spots more visible, one way of doing this is by using a method known as Bradley Roth Thresholding.

### 1.3.2 Bradley Roth Thresholding

Bradley-Roth imaging is a method for processing live videos and photos, designed to operate efficiently on most computer systems. This algorithm works by analyzing the pixel counts in an image and setting it to dark if the pixel counts are lower by a certain percentage, denoted as '$p_t$', than the average brightness of its surrounding pixels and vice versa

We employed Bradley-Roth image processing to enhance the visibility of sunspots in our images. Similar to our simple thresholding, this technique assigns binary values (0 and 1) to image pixels based on their deviation from the $p_t$ value. However, unlike the fixed threshold value used for the entire image, Bradley-Roth breaks the image into smaller windows, calculating the average pixel counts for each window. The threshold provided represents the percentage deviation from the $p_t$ value.

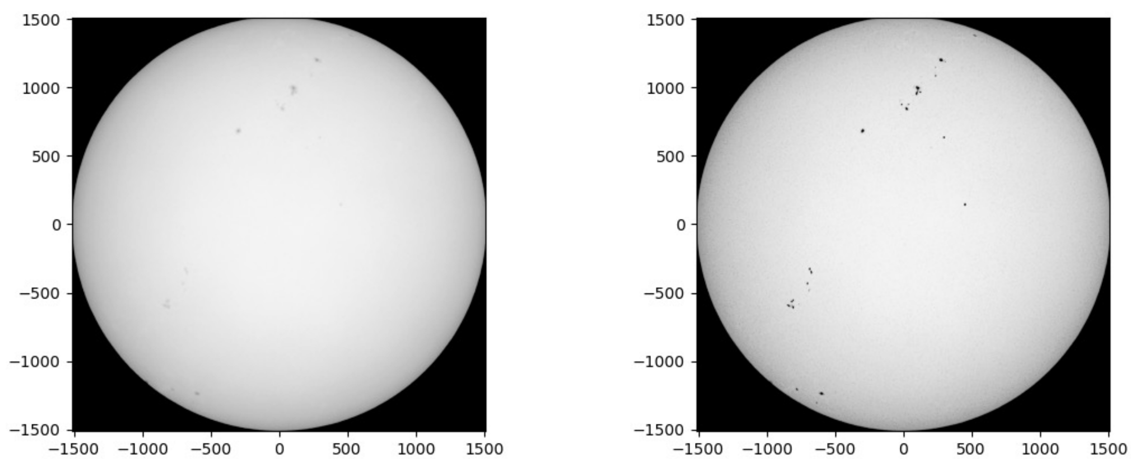



Image before Bradley Roth Thresholding

Image after Bradley Roth Thresholding

**Figure 1.7:** Using Bradley Roth thresholding improves the visibility of the sunspots since they have from being some 'pixel counts' to now being assigned the value 0 or 'dark'(essentially the color black) making them stand out from the rest of the sun

This helps us identify and track the sunspots even better.

As mentioned in the previous section 1.3.1, why do we need to create a mask and then fix the value of the background to zero? Why not just use Bradley Roth's shareholding directly? The answer to these questions is that Bradley Roth conducts thresholding by breaking down the image into smaller windows or boxes. Since the noise in the background is unequal, it will get thresholded unequally. Making the image look like the following

### 1.3.3 Tracking the spot

Now that we can see the sunspots, we can begin tracking them. One of the first methods we used was simply assuming that the spot moves across the solar disc, assuming the solar disc is a 2D projection

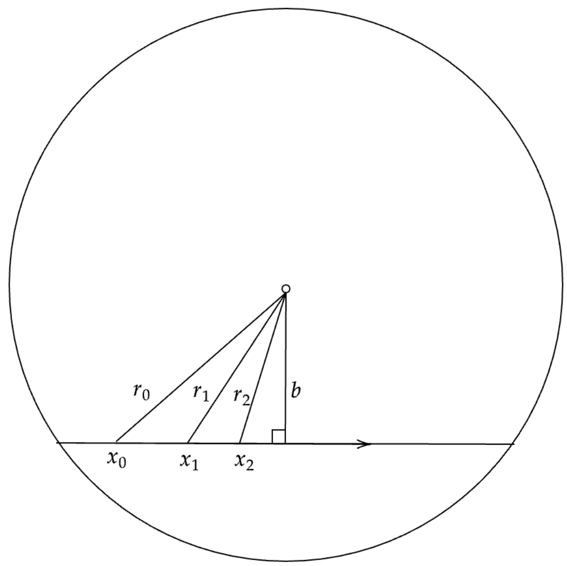



**Figure 1.8:** The x values are the different sunspots, and the distance that the spots move to. The r values are the distance from the spot to the center.

Looking at the image, we can deduce the following equations.

$$r_0^2 = b^2 + x_0^2$$

$$r_1^2 = b^2 + x_1^2$$

$$r_2^2 = b^2 + x_2^2$$

Assuming vx is the velocity with which the spot moves over some time t, we can further continue with our analysis using the following equations

$$r_1^2 - r_0^2 = vxt_1(vxt_1 + 2x_0)$$

$$r_2^2 - r_0^2 = vxt_2(vxt_2 + 2x_0)$$

Using these five equations we can find the values of $x_0$ and vx. Furthermore, these values of x that we get are where the spots should be on the x axis, and 'b' is the co oridnate on the y axis (b is the same since the spots should ideally move in a straight line along the latitudes.) One of the errors we faced with this model was that x-axis co ordinates appeared to be out of place for whent the spot was near the edge, this could possibly be due to the curvature of the sun as we move closer to the edge in the image. We assume this to be a 2D shape and do not account for such curvature.

On rotating the images such that the spot aligns with new x and y co-oridnates we can see while it does reach the desired y co-oridnate, the predicted x co-oridnate is quite off.

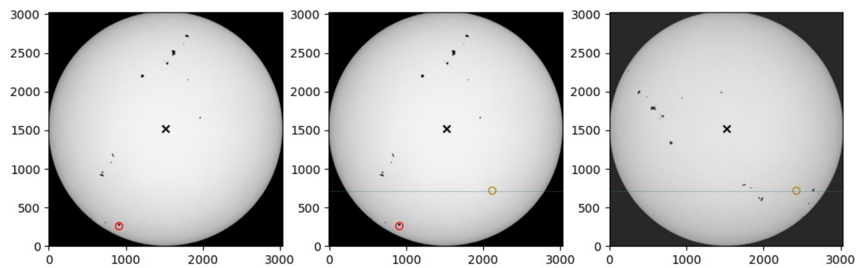



**Figure 1.9:** The image on the left is the original image, the one in the center has a gold circle which is the spot we wish to analyse. In the image on the right, the blue line is where the line along which the spot should move, the spot does appear on this line. The gold circle in the image on shows the predicted x co-ordinate, which appears to be off

### 3D projection

To begin analysis using a 3D model of the sun, we must first start by introducing new coordinates for the same.

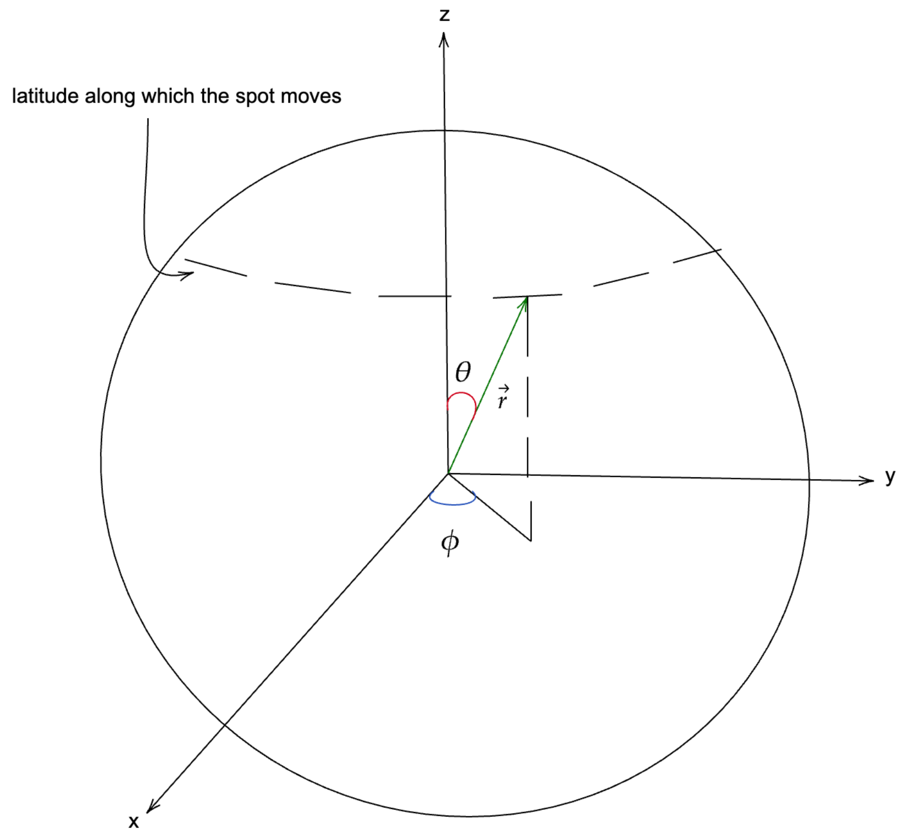



**Figure 1.10:** 3D representation of the sun. Here $\vec{r}$ is the vector along the radius- measuring from the center to a sunspot.

If we take the magnitude of the vector $\vec{r}$, we get the following equation .

Further conducting analysis in the yz plane we are left with the following equation.

$$\left(\frac{\vec{r}}{R}\right)_{\text{proj}} = sin\theta sin\phi + cos\theta \tag{1.1}$$

let us rewrite the projection as some variable $\rho$ Therefore the equation we get is

$$\rho = sin\theta sin\phi + cos\theta$$

If we rewrite cos$\theta$ as $\zeta$- which is the actual position of the spot on the z axis we get an equation

$$\rho = (1-\zeta^2)sin^2\left(\frac{2\pi(t-t_0)}{T}\right) + \zeta^2 \tag{1.2}$$

Where $t$ is the time between two images, $t_0$ is the time the spot to takes to reach the point of closest approach and $T$ is the rotation period of the spot around the sun. Now, if we were to plot a graph of this $\rho^2$ vs $t$, we should expect a graph as shown in Fig1.11

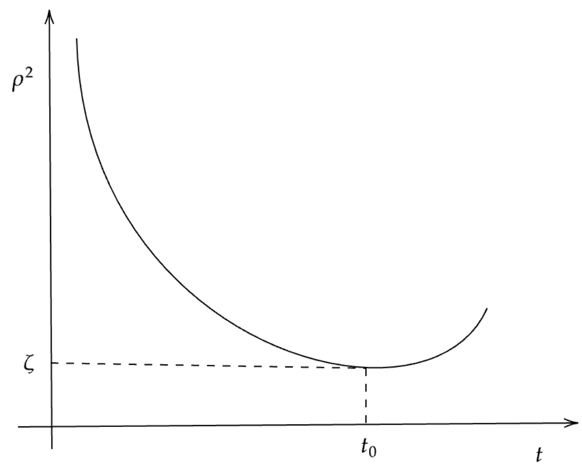



**Figure 1.11:** Plotting $\rho^2$ vs t should ideally gives us a similiar graph. The minima we see is the point of closest approach, or when the spot is in roughly on the center of the sun relative to the observers view . This minima gives us $\zeta$ and $t_0$

One of the important things to keep in mind while conducting the experiment is to pick out spots that would be easy to identify in multiple different images. We chose a spot that appears on one the lower latitudes on the suns surface.

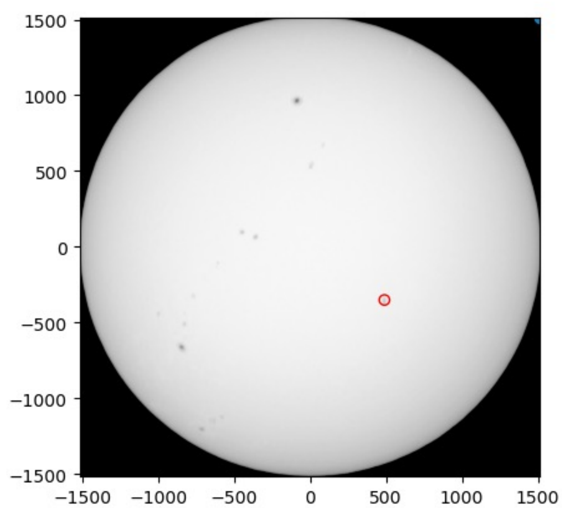
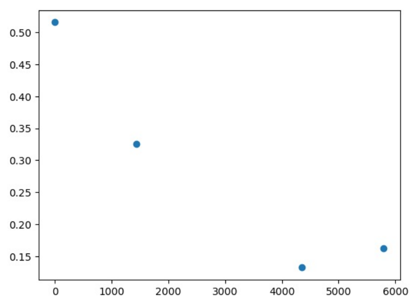
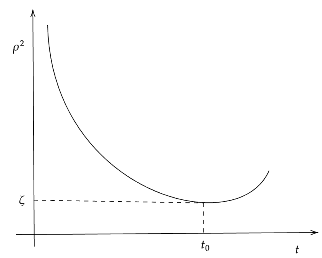



**Figure 1.12:** Red circle indicates the spot being tracked. This spot stood out in the image and through multiple days of images and rough location we were able to ensure that it was the same spot that we were tracking

Following is a the analysis of the 4 different days worth of images we had with the same spot visible on the surface. The dates- 27th,28th, 30th June and 1st July. We begin by plotting a graph of




Scatter plot of our data




Expected function

**Figure 1.13:** One can see that our scatter plot (left) has a similar shape as the plot we expect (right) )

By plotting a curve fit of equation 1.2 we can find the values of zeta and $t_0$. The fit parameters are the $\zeta$, $t_0$ and $T$

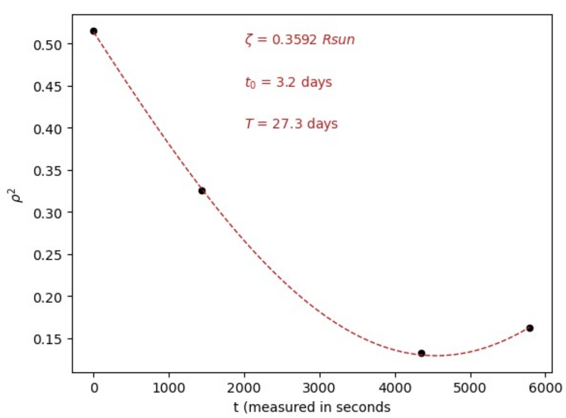



**Figure 1.14:** plot of $\rho^2$ vs t

Now that we have the values of the co-ordinates of $\rho$ and $\zeta$ for various images, we can move on further with our analysis. Since the aim is to align the consecutive images such that their solar axis aligns with the image North, we need to rotate the image by a certain degree $\alpha$ as shown in Fig

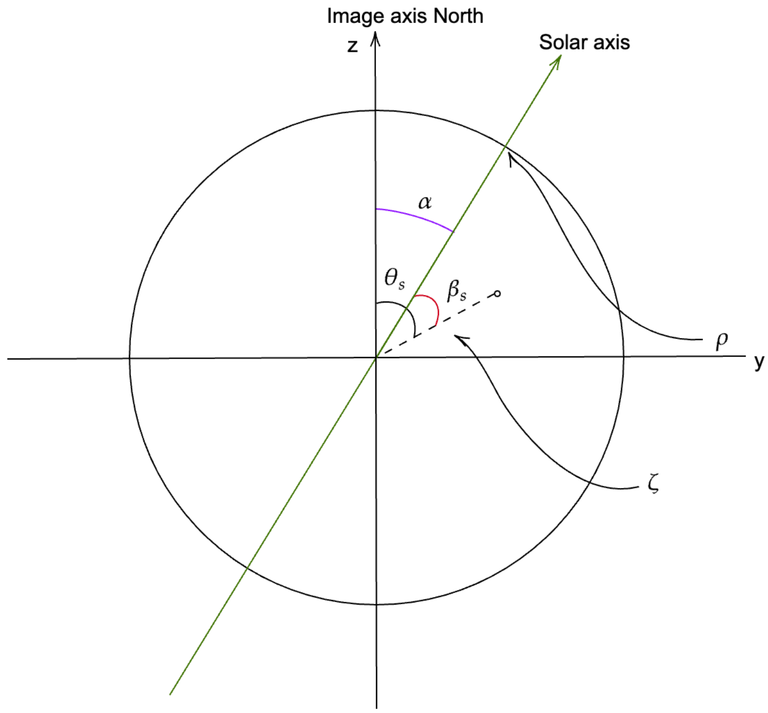



**Figure 1.15:** Alpha is the angle between the Image north and the solar axis. Since we know the values of $\zeta$ and $\rho$, using some simple trigonometry we can deduce the values of the $\theta_S$ and $\beta_s$. $\alpha$ then would simply be the difference of the two angles

Simply deducing from the image one can tell that alpha should simply be the difference between angles $\beta_S$ and $\theta_S$. $\beta_s$ is the angle between the the sun spot and the solar axis. Whereas $\theta_S$ is the angle between the spot and image north. Therefore, the equation for $\alpha$ should look like the following-

$$\alpha = \theta_S - \beta_S \tag{1.3}$$

In equation 1.3 we can write $\beta_S$ as $cos^{-1}\left(\frac{|\zeta|}{\rho}\right)$. However it is important to note that this formula does not always work due to quarant ambiguities, and different values of zeta. Therefore we have another set of equations for all the possible set of $\beta_S$ values. Let $\beta_S$= $\bar{\beta_S}$

$\bar{\beta_S} = cos^{-1}\left(\frac{|\zeta|}{\rho}\right)$
**1)** $\zeta > 0$

$$\beta_S = \bar{\beta_S}\ (t > t_0)$$

$$\beta_S = -\bar{\beta_S}\ (t < t_0)$$

3) $\zeta > 0$

$$\beta_S = \pi - \bar{\beta_S}\ (t > t_0)$$

$$\beta_S = \pi + \bar{\beta}_S \; (t < t_0)$$

### Rotating the images

Once we have the values of $\alpha$ for each individual image, we can go ahead and rotate them. We used four days of data- 27th, 28th , 30th and the 1st. Before rotating the images they looked as such

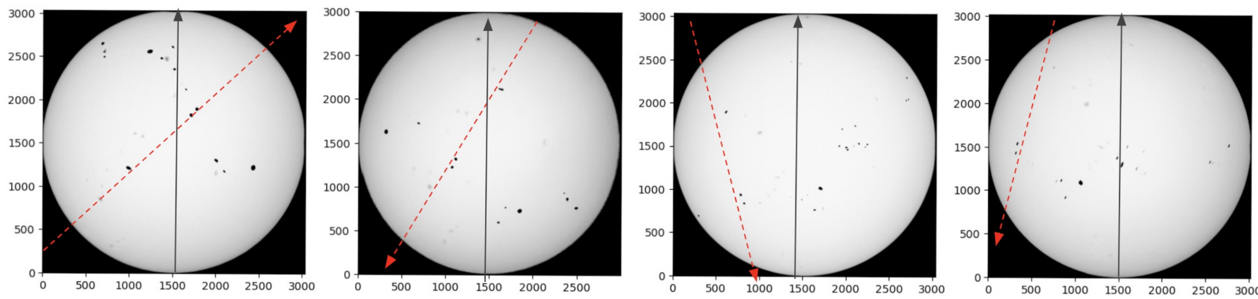



**Figure 1.16:** Images from 4 different days. As you can observe, the spots do not appear to be aligned in any way. the black line represents the image axis, and the red dotted line the apparent spin axis

The images after they've been rotated by some degree $\alpha$

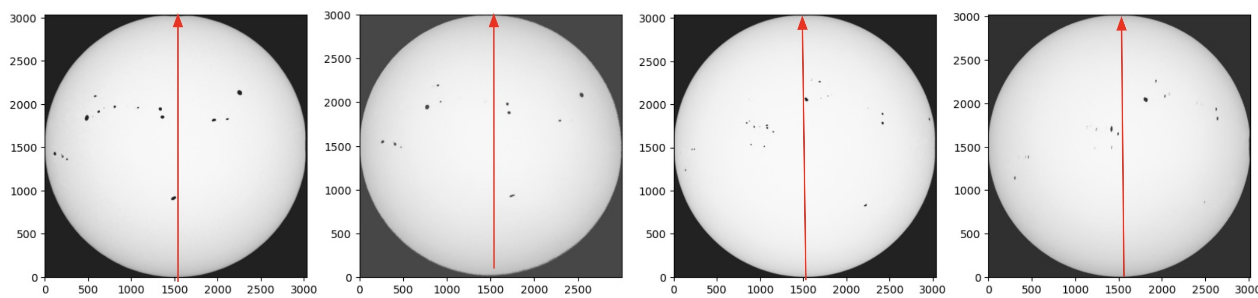
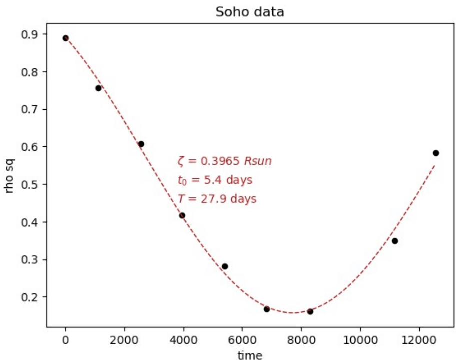



**Figure 1.17:** Images from the 4 days after rotated by an individual angle $\alpha$ unique to each image. The red line represents the image axis, one cans see that the spots appear to move in one line, following a latitude.

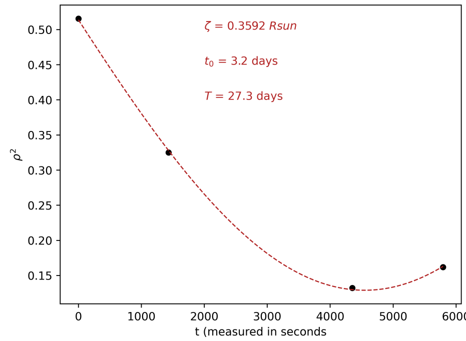
Using this method we have been able to align the image north along with the solar axis of the sun.

## 1.4 Results and Discussions

While we were able to align the Solar axis to the Image north, on conducting the same analysis with SIDC data which gaves us many more consecutive days with clear images. We missed quite a few days of images due to bad weather. However, plotting the graph of $\rho^2$ vs $t$ for the SIDC data, along the same spot gives us a graph as such.




Plot of $\rho^2$ vs t using SIDC data




Plot o $\rho^2$ vs t for our data

**Figure 1.18:** We can see that the plots for the same spot using SIDC data vs our data is quite similiar. However, there is also a difference in the$\zeta$, $t_0$,and T values )

| | $\zeta$ | $t_0$ | $T$ |
|---|---|---|---|
| SIDC | 0.3965 | 3.2 | 27.9 |
| Our Data | 0.3592 | 5.4 | 27.3 |

The important factor to take into consideration is $\zeta$. With a greater number of data points we ourselves could maybe get a more accurate and better value of $\zeta$ Therefore, with a lager number of images not only would be able to get a better look at the suns rotation but also have more accurate value for $\zeta$ which would help us rotate the images more accurately.