University of Science and Technology of Hanoi
Department of Space and Application

SOLAR SYSTEM AND CELESTIAL MECHANICS
Lecturer: *DR. PHAN Thanh Hien*

MID-TERM REPORT

Authors:	*Student ID:*
DUONG Thu Phuong	22BI13362
HA Bui Khoi Nguyen	22BI13337
TRAN Quang Anh	22BI13041
DO Duc Vinh	22BI13471
NGUYEN Thi Hai Chi	BA12-029
NGUYEN Thuy Linh	BA11-063

May 25, 2024

Contents

1 Overview **3**

2 Equipments **4**
2.1 Telescope: Celestron NexStar Evolution 9.25 . . . 4
2.2 Eyepieces: 56mm, 25mm, 9mm . . . 4
2.3 Software: Stellarium, ASIStudio . . . 5
2.4 ZWO ASI 178MC Camera . . . 5
2.5 Focal reducer telescope . . . 6

3 Jupiter Observation **7**
3.1 Setting up the telescope . . . 7
3.2 Results . . . 9

4 Processing data from Quy Nhon Observatory **10**
4.1 Velocity . . . 10
4.2 Position vector . . . 16

5 Conclusion and Acknowledgement **21**
5.1 Conclusion . . . 21
5.2 Acknowledgement . . . 21
5.3 References . . . 21

List of Figures

1 Celestron NexStar Evolution 9.25 . . . 4
2 56mm, 25mm and 9mm eyepieces . . . 4
3 ZWO ASI 178MC Camera . . . 5
4 Focal Reducer Telescope . . . 6
5 Jupiter - 18:32 - 1st April . . . 9
6 Jupiter - 19:02 - 1st April . . . 9
7 Code part to read the fit file . . . 10
8 The fit file information . . . 10
9 The code to find angular distance . . . 11
10 Angular distance output at 13h43 . . . 12
11 Angular distance output at 13h48 . . . 13
12 Angular distances output of the second period . . . 14
13 Angular distances output of the third period . . . 14
14 Angular distances output of the foruth period . . . 15
15 Angular distances output of the last period . . . 15
16 Position vector coding part . . . 19
17 Position vector coding part . . . 19
18 Position vector results . . . 20

1 Overview

During the theory sessions of this subject, we explored various aspects of the solar system, including planetary science, planetology, extrasolar planets, giant planets, and more. Additionally, there were sessions where students presented simulations on many solar system topics in a sequential manner. This report is the result of a practical observation session conducted in Hanoi, utilizing data provided by the Quy Nhon Observatory.

In this report, we aim to give you (or any non-experimented observers) the means of achieving observations. This includes knowing what kind of object is observable with a given telescope when it is possible to observe it, and how to do it as efficiently as possible.

The question to address when planning astronomical observations is of course the aim of the observation. The observations are usually part of a scientific or pedagogical project. For a scientific project, the observer will choose a list of precise and similar targets. For a pedagogical project, on the other hand, it is more interesting to observe a wide sample of astronomical objects.

In our practical observation session, we will focus on observing Jupiter and its satellites from Hanoi, under the expert guidance of Dr. PHAN Thanh Hien. Utilizing the data provided by Dr. LE Quang Thuy from the Quy Nhon Observatory, we will perform calculations and analyses to enhance our observational accuracy and understanding.

This report will feature photographs of Jupiter and its satellites taken during our observation session in Hanoi. Furthermore, it will provide a detailed outline of our observational methods, the theoretical concepts underlying, the step-by-step procedures we followed, and the programming techniques we employed to process the data. Specifically, we will calculate the velocity and position vectors of the observed objects based on the data provided by the Quy Nhon Observatory.

The process of astronomical observation involves several critical steps. Initially, it is essential to select the appropriate equipment, including telescopes and cameras, to capture high-quality images of the celestial objects. Understanding the capabilities and limitations of the equipment is crucial for optimizing the observation process.

Once the equipment is set up and the optimal timing is determined, the actual observation begins. This involves pointing the telescope at the target object, adjusting the focus, and capturing images or data. Throughout this process, it is important to document the settings and conditions to ensure reproducibility and accuracy in the analysis.

After the observation in Hanoi, we will use programming techniques to process the data from the Quy Nhon Observatory and calculate the velocity and position vectors of the given object in files.

2 Equipments

2.1 Telescope: Celestron NexStar Evolution 9.25

The Celestron NexStar Evolution 9.25 is a telescope designed for amateur astronomers. Its primary function is to observe celestial objects, including planets, stars, and deep-sky objects. It features a 9.25-inch aperture, computerized GoTo mount, and wireless control capabilities, allowing users to locate and track celestial objects easily. This telescope is suitable for both visual observation and astrophotography, making it versatile for different aspects of amateur astronomy.

Figure 1: Celestron NexStar Evolution 9.25

2.2 Eyepieces: 56mm, 25mm, 9mm

Eyepieces in astronomy serve to magnify the image produced by a telescope, allowing astronomers to observe celestial objects in greater detail. They are interchangeable lenses that you place at the focal point of the telescope. Different eyepieces provide varying levels of magnification, field of view, and eye relief, giving observers flexibility in choosing the desired view based on the target and observing conditions. Eyepieces play a crucial role in shaping the visual experience and determining the level of detail visible during astronomical observations.

Figure 2: 56mm, 25mm and 9mm eyepieces

2.3 Software: Stellarium, ASIStudio

Stellarium: Stellarium is a planetarium software that simulates the night sky in 3D. It provides a realistic depiction of celestial objects, constellations, and other astronomical phenomena. Users can use Stellarium to plan observations, identify stars and planets, and simulate the night sky from different locations and times.

ASIStudio: ASIStudio is software associated with ZWO ASI (Astro Imaging) cameras. It's designed for astrophotography and camera control. ASIStudio allows users to capture and process astronomical images using ZWO cameras. It often includes features such as image capture, stacking, and processing tools tailored for astrophotography.

2.4 ZWO ASI 178MC Camera

A special camera allows images to be displayed on a computer through the ASIStudio. It will be placed in place of the eyepiece once we have determined the location of the object we are looking for in the sky.

The ZWO ASI 178MC is a popular camera choice among astrophotographers, especially for planetary imaging and lunar imaging. Here are some key features of the ZWO ASI 178MC camera

Figure 3: ZWO ASI 178MC Camera

- Sensor: The ASI 178MC features a 1/1.8" CMOS sensor with a resolution of 6.4 megapixels (3096x2080). This sensor size strikes a good balance between field of view and resolution, making it suitable for both planetary and deep-sky imaging.
- Pixel Size: The camera has relatively small pixels, typically around 2.4 microns, which allows for high-resolution imaging of planetary details and lunar features.
- Color Imaging: The "MC" in the camera's name indicates that it's a color camera (RGB). This makes it convenient for capturing color images without the need for additional filters or image processing steps.
- High Frame Rates: The ASI 178MC is capable of high-speed imaging, with frame rates of up to several hundred frames per second (fps) depending on the settings and the capabilities of your computer.
- Low Noise: ZWO cameras are known for their low noise characteristics, which is crucial for capturing clear and detailed images of celestial objects, especially in low-light conditions.
- Versatility: While primarily designed for planetary and lunar imaging, the ASI 178MC can also be used for capturing brighter deep-sky objects and for guiding purposes.

2.5 Focal reducer telescope

Also known as a focal length reducer or a focal reducer lens, is an optical device used in telescopes and camera systems to decrease the focal length of a telescope's optical system. We use it when do observe with Sky-Watcher Evostar 72ED telescope. The focal reducer can help us

- Wider Field of View: By reducing the focal length, a focal reducer widens the field of view, allowing you to capture more of the sky in a single image.
- Increased Brightness: A shorter focal length results in a brighter image, which can be particularly useful for faint deep-sky objects.
- Reduced Exposure Times: With a wider field of view and increased brightness, exposure times for astrophotography can be shorter, reducing the chances of blurring due to Earth's rotation.
- Reduced Image Scale: Objects appear larger in the field of view, making them easier to observe and photograph.

Focal reducers are commonly used in astrophotography to enhance imaging capabilities, especially for capturing wide-field views of the night sky and faint deep-sky objects. They are often paired with cameras to optimize the imaging system for different types of astronomical targets and to achieve specific imaging goals.

Figure 4: Focal Reducer Telescope

3 Jupiter Observation

3.1 Setting up the telescope

Process:

1. Setup the tripod:

First, We spread the tripod legs and stand the tripod upright and remove the Tripod Support Nut and Washer from the central column attached to the top of the tripod.Then place the accessory tray over the central column so that each of the three arms of the tray is touching a tripod leg. Thread the nut and washer back onto the threaded column and firmly tighten it into place. The accessory tray should not be able to move against the tripod. Next we adjust the height of the tripod by loosening the lock knobs on the end of each tripod leg. Then adjust the leg height as needed and retighten the lock knobs, one leg at a time. Confirm the tripod is level using the built-in bubble level on the tripod base.

2. Mount the telescope :

We place the fork arm mount on the tripod, carefully centering the mount over the center post on the tripod head. Do not let go of the mount until it has registered with the center post. With the mount resting on the flat top surface of the tripod head, rotate the mount until the three mounting sockets align. The sockets will click into place, indicating they are aligned. Then thread the three attached mounting bolts from underneath the tripod head into the bottom of the telescope base. Tighten all three bolts.

3. Setup the telescope Optical Tube:

Now we need to unlock the altitude clutch by loosening the orange altitude clutch lock knob. Rotate the altitude axis until the quick release knob faces downward. Tighten the altitude clutch lock knob. Next, loosen the quick release knob a couple of turns to allow room for the dovetail on the telescope optical tube. Slide the telescope optical tube into the quick release slot from the back side of the telescope. The fork arm should be on the left side of the optical tube. Keep hold of the optical tube and secure it into place by tightening the quick release knob.

4. Setup Visual Accessories:

Important thing to do now it is insert the mirror star diagonal into the visual back of the telescope and secure it into place by tightening the two set screws on the visual back, then insert the 25 mm eyepiece into the mirror star diagonal and secure it into place by tightening the two set screws on the diagonal. Plug the remote control into the AUX port. Star Pointer Red Dot

Finder:

Loosen the two Phillips head screws on the Star Pointer's dovetail clamp slightly using a screwdriver. Slide the Star Pointer over the dovetail rail preinstalled on the telescope. Tighten the two Phillips head screws to secure the finder in place

Align the telescope to an arbitrary object:

1. Direct the laser pointer to the location of a light bulb on a distant building.

2. Align the telescope to that position

3. Adjust the finder until the red dot point exactly on that position.

4. Adjust the focus of the telescope until we can see the object clearly

Adjust the telescope to Jupiter

1. Turn on the power switch on the telescope.

2. Wait until the hand control is ready.

3. Press the ENTER button to begin the alignment process.

4. A menu will appear showing the many different methods of alignment. Roll and select Solar System Align, press ENTER to continue.

5. Select Standard Time.

6. Select zone 7 for time zone, then enter the date.

7. Press ENTER, the display will now prompt you to "Select Object" from the displayed list on the hand control. Roll and select Jupiter, press ENTER.

8. When the view of the telescope gets close to Jupiter, look through the Red Dot Finder and move the telescope to center the red dot directly to Jupiter.

9. Finally, center the star in the eyepiece and press ALIGN.

10. Wait for the hand control to calculate the alignment. After a few seconds, the telescope is now aligned.

3.2 Results

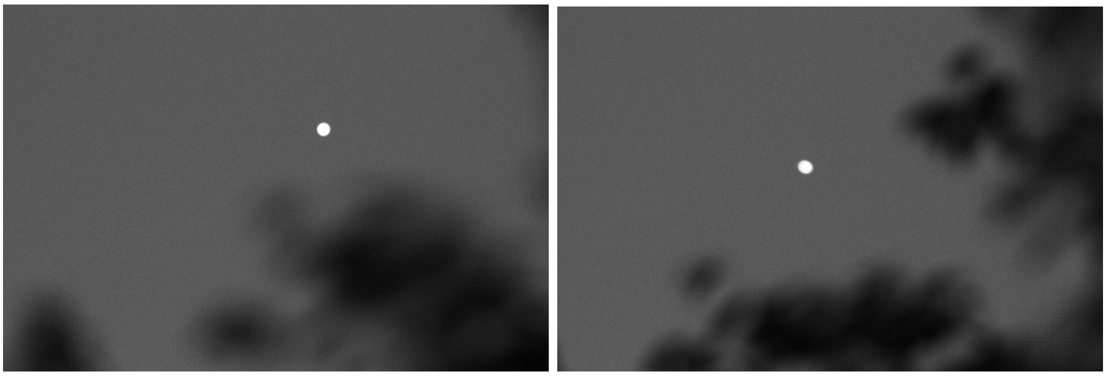

Figure 5: Jupiter - 18:32 - 1st April

There images of Jupiter we took from 18h32 to 19h02 on 1st April.

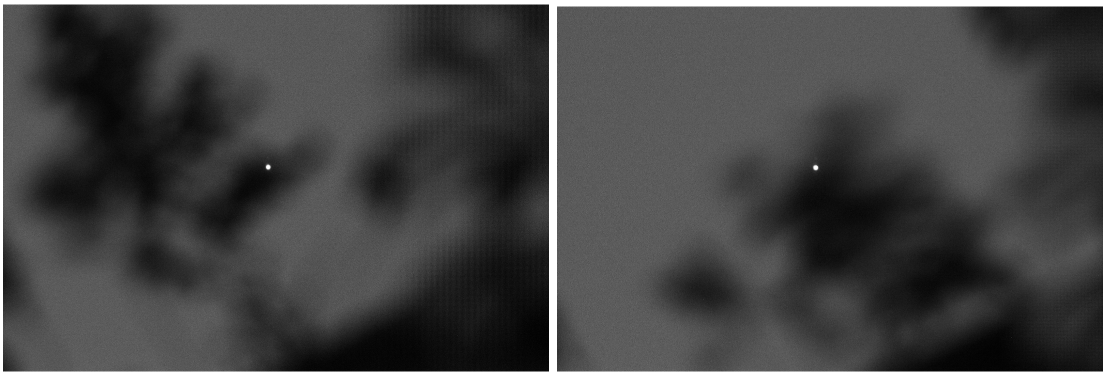

Figure 6: Jupiter - 19:02 - 1st April

At the angle of our telescope, Jupiter was hidden behind trees (as can be seen in the photos) but we were still able to capture the bright spot that is Jupiter.

Few first months of the year, it is difficult to have observations with satisfactory products at this time, with the influence of light pollution, cosmic dust and Jupiter setting very quickly (we only can observe for about 30 minute) it was possible to take these pictures of Jupiter without its moons.

4 Processing data from Quy Nhon Observatory

In this section, we using data from Quy Nhon Observatory to estimate the velocity and position in own orbit of the object. There are total 44 fit files took from 13h43 to 18h33 on 7th May with 6-7 minutes between two consecutive photos, each photo has the main object we need to analysis and other background stars.

4.1 Velocity

First, we need to estimate the velocity of the main object. The velocity of our object can change due time and position in its orbit so we will find the velocity of the orbit over many periods of time from 13h to 18h and see the large and small changes in velocity as it moves around the orbit.

```python
from astropy.io import fits
import matplotlib.pyplot as plt

with fits.open("AppData\\6 Hebe-Bin2-0001L.fit") as hdul:
    hdul.info()

    header = hdul[0].header
    data = hdul[0].data

    print(header)

    plt.imshow(data, cmap='gray', origin='lower')
    plt.colorbar()
    plt.title('Image from FITS file')
    plt.xlabel('X Pixel')
    plt.ylabel('Y Pixel')
    plt.show()
```

Figure 7: Code part to read the fit file

This is the code part to help us read the fit file from Quy Nhon Observatory

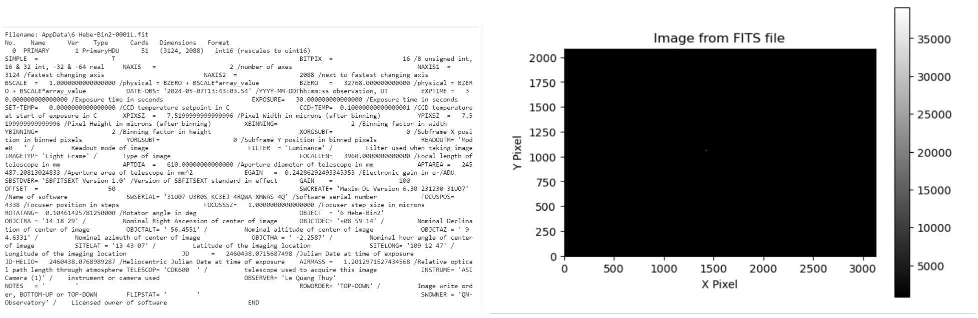

Figure 8: The fit file information

The results with have many information of the image, it is the first file in the list took at 13h43 7th May. We read other files with the same method.

We already know the difference of time due to results information in the files, now to estimate the velocity, we need to know the angular distance between the main object and star background. Here we using "contours" in code part, "contours" can show two brightest points in the image, from this we can easily to determine the angular distance with unit is degrees (we use degrees/minute is the unit of the velocity in this report)

```
from skimage import io , color , feature , measure
import numpy as np
import matplotlib.pyplot as plt
import cv2

object = io.imread(r"C:\Users\Admin\Pictures\Screenshots\Screenshot (436).png", as_gray = True)

threshold = object > np.max(object)*0.9

contours = measure.find_contours(threshold ,0.8)
contours = sorted(contours , key = len , reverse = True)

object_contours = contours [0]
star_contours = contours [1]

object_center = np.mean(object_contours , axis =0)
star_center = np.mean(star_contours , axis =0)

object_y , object_x = object_center
star_y , star_x = star_center

pixel_dist = np.sqrt((object_x - star_x )**2 + (object_y - star_y)**2)
print (" Apparent distance :", np . round (pixel_dist ,2) ,'pixel ')

fov_long = 1.01
fov_short = 0.68

img_width = object.shape[1]
img_height = object.shape[0]
ang_dist_long = pixel_dist*(fov_long/img_width)
ang_dist_short = pixel_dist*(fov_short/img_height)
ang_dist = np.sqrt(ang_dist_long**2 + ang_dist_short**2)
print("Angular distance: ", np.round(ang_dist, 2), "degrees")

plt.imshow(object, cmap ='gray')
plt.plot(object_contours [: , 1] , object_contours [: , 0] , 'y', linewidth=2 , alpha = 0.5)
plt.plot(star_contours [: , 1] , star_contours [: , 0] , 'b', linewidth=2 , alpha = 0.5)
plt.plot(object_x , object_y , 'yo ', markersize=10 , alpha = 0.5 , label = 'x')
plt.plot(star_x , star_y , 'bo ', markersize=5 , alpha = 0.5 , label = 'y')

plt.legend()
plt.show
```

Figure 9: The code to find angular distance

```
Angular distance:  0.79 degrees

<function matplotlib.pyplot.show(close=None, block=None)>
```

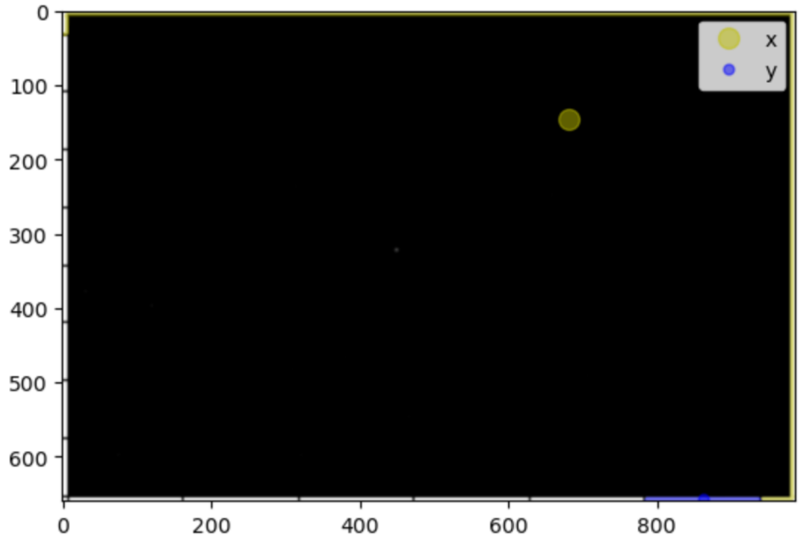

Figure 10: Angular distance output at 13h43

The output show us the main object and star background with the angular distance results at 13h43 7th May, the output below is at 13h48 7th May

```
Angular distance:  0.98 degrees

<function matplotlib.pyplot.show(close=None, block=None)>
```

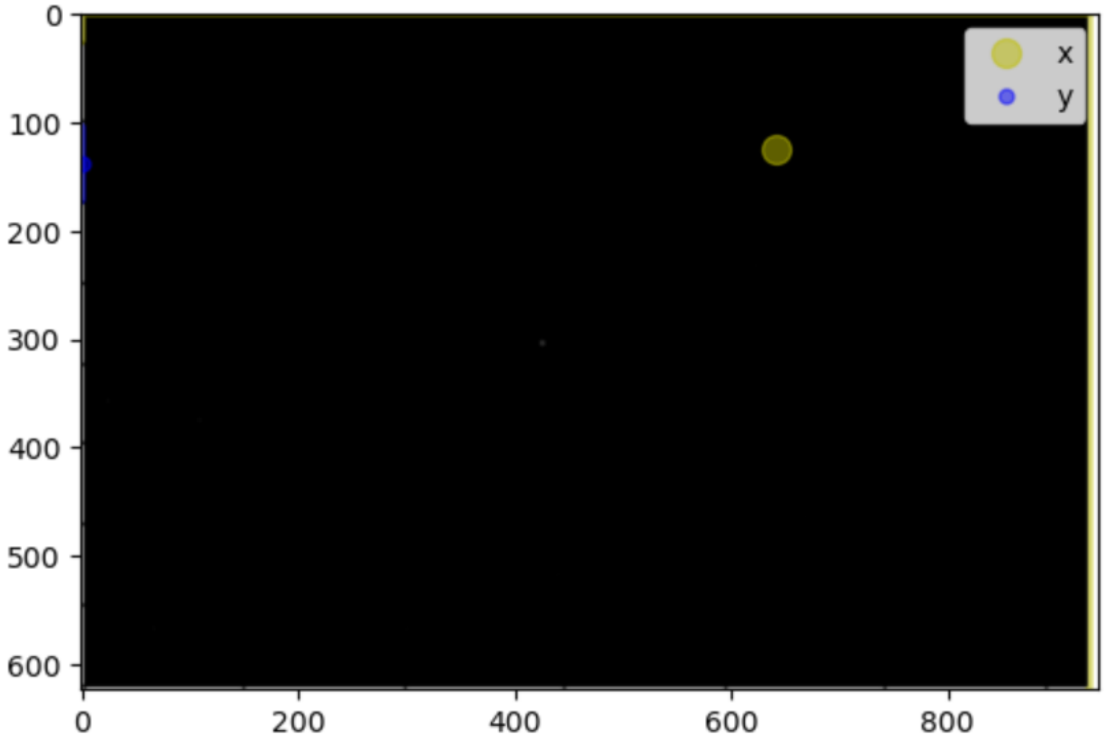

Figure 11: Angular distance output at 13h48

As we mentioned above, we will estimate many velocities value in many periods time, two first results above belong to period one (13h43 to 13h48 7th May), the difference time is 6 minutes and the difference of angular distance is , so the first velocity value (v1) is:

$$v_1 = \frac{\delta S}{\delta t} = \frac{0.98 - 0.79}{6} = 0.038(degrees/minute) \tag{1}$$

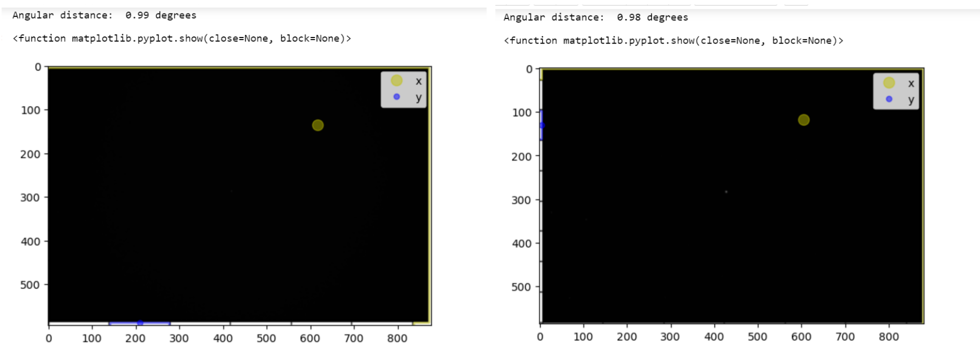

Figure 12: Angular distances output of the second period

The second period we consider is between 14h39 and 14h45 7th May. Two angular distances of them has showed above, the second velocity value is

$$v_2 = \frac{\delta S}{\delta t} = \frac{0.99 - 0.98}{6} = 0.00167(degrees/minute) \tag{2}$$

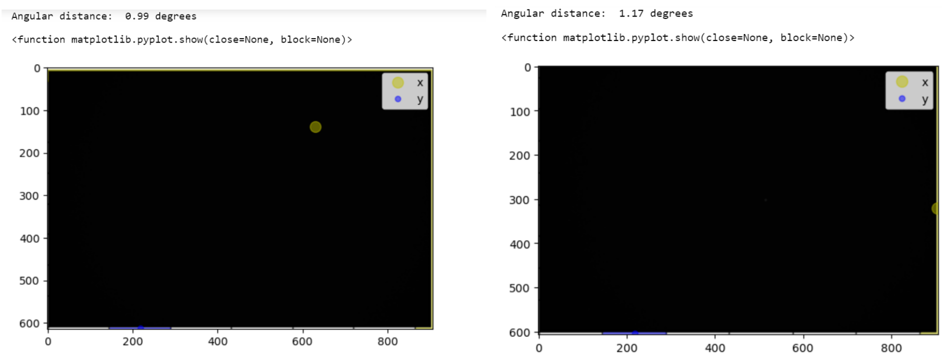

Figure 13: Angular distances output of the third period

The third period we consider is between 15h59 and 16h10 7th May. Two angular distances of them has showed above, the third velocity value is

$$v_3 = \frac{\delta S}{\delta t} = \frac{1.17 - 0.99}{11} = 0.0163(degrees/minute) \tag{3}$$

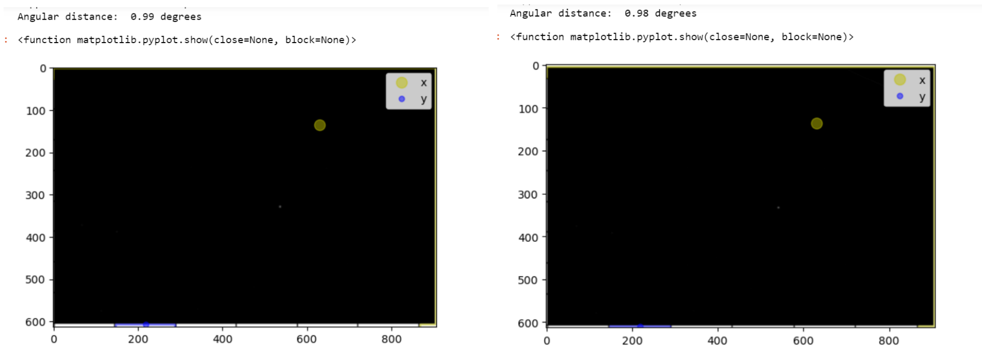

Figure 14: Angular distances output of the foruth period

The fourth period we consider is between 17h07 and 17h13 7th May. Two angular distances of them has shown above, the fourth velocity value is

$$v_4 = \frac{\delta S}{\delta t} = \frac{0.99 - 0.98}{6} = 0.00167(degrees/minute) \tag{4}$$

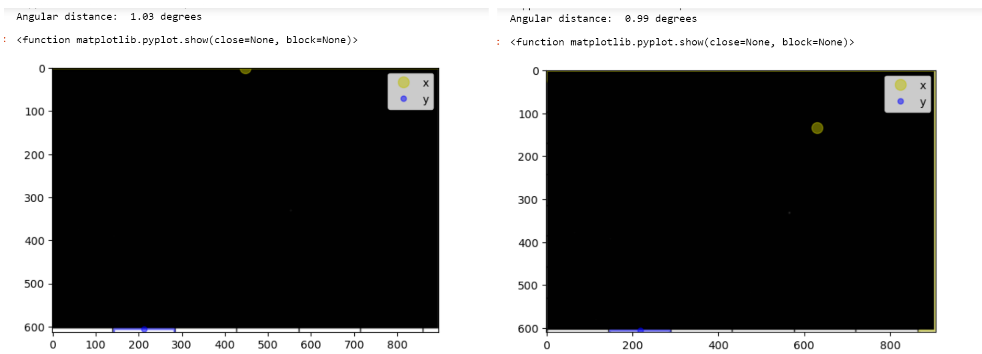

Figure 15: Angular distances output of the last period

The last period we consider is between 18h10 and 18h27 7th May. Two angular distances of them are shown above, the last velocity value is

$$v_5 = \frac{\delta S}{\delta t} = \frac{1.03 - 0.99}{17} = 0.00235(degrees/minute) \tag{5}$$

The average velocity of the object during the period from 13h43 to 18h33 on 7th May is calculated as follows

$$v_{arg} = \frac{v_1 + v_2 + v_3 + v_4 + v_5}{5} = 0.012(degrees/minute) \tag{6}$$

The object at position one, period one moved the fastest, and at position three, period three got the slowest velocity value. Where is the position of the object? We will find them in the next part.

4.2 Position vector

This is the most difficult part of this report, we need to apply theoretical knowledge about spherical coordinates in space and skills in working with python to process that pile of data most effectively. To determine position vector of this object, we would like to apply the Gauss's method.

In orbital mechanics (a subfield of celestial mechanics), Gauss's method is used for preliminary orbit determination from at least three observations (more observations increases the accuracy of the determined orbit) of the orbiting body of interest at three different times. The required information are the times of observations, the position vectors of the observation points (in Equatorial Coordinate System), the direction cosine vector of the orbiting body from the observation points (from Topocentric Equatorial Coordinate System) and general physical data.

We have the equation here

$$R_n = [\frac{R_e}{\sqrt{1-e^2\sin^2\phi_n}} + H_n]\cos\phi_n(\cos\phi_n\hat{I} + \sin\phi_n\hat{J}) + [\frac{R_e(1-e^2)}{\sqrt{1-e^2\sin^2\phi_n}} + H_n]\sin\phi_n\hat{K} \tag{7}$$

- R_n is the respective observer position vector (in Equatorial Coordinate System).
- R_e is the equatorial radius of the central body.
- r_e is the geocentric distance.
- f is the oblateness (or flattening) of the central body.
- e is the eccentricity of the central body.
- ϕ_n is the geodetic latitude (the angle between the normal line of the horizontal plane and the equatorial plane).
- H_n is the geodetic altitude.
- θ_n is the local sidereal time of the observation site.

In the algorithm, the initial process starts with adding vectors to calculate the position vector of the orbiting body. Then, leveraging the conservation of angular momentum and principles of Keplerian orbits (which assert that an orbit exists in a two-dimensional plane within three-dimensional space), a combination of these position vectors is established. Additionally, the connection between a body's position and velocity vectors via Lagrange coefficients is utilized, leading to the incorporation of these coefficients. Finally, through vector manipulation and algebraic operations, the subsequent equations are derived

Step 1 Calculate time intervals, subtract the times between observations

$$\tau_1 = \tau_1 - \tau_2 \tag{8}$$

$$\tau_3 = \tau_3 - \tau_2 \tag{9}$$

$$\tau = \tau_3 - \tau_1 \tag{10}$$

Step 2 Calculate cross products p1, p2 and p3, take the cross products of the observational unit direction

Step 3 Calculate common scalar quantity (scalar triple product), take the dot product of the first observational unit vector with the cross product of the second and third observational unit vector

$$D_0 = \hat{p_1}(\hat{p_2}x\hat{p_3}) \tag{11}$$

Step 4 Calculate nine scalar quantities with m and n run from 1 to 3

$$D_m n = R_m . p_n \tag{12}$$

Step 5 Calculate scalar position coefficients:

$$A = \frac{1}{D_0}(-D_{12}\frac{\tau_3}{\tau} + D_{22} + D_{32}\frac{\tau_1}{\tau}) \tag{13}$$

$$B = \frac{1}{6D_0}[D_{12}((\tau_3)^2 - \tau^2)\frac{\tau_3}{\tau} + D_{32}((\tau^2 - (\tau_1)^2)\frac{\tau_1}{\tau}] \tag{14}$$

$$E = R_2 . \hat{p_2} \tag{15}$$

Step 6 Calculate the squared scalar distance of the second observation, by taking the dot product of the position vector of the second observation

$$(R_2)^2 = R_2 . R_2 \tag{16}$$

Step 7 Calculate the coefficients of the scalar distance polynomial for the second observation of the orbiting body

$$a = -(A^2 + 2AE + (R_2)^2) \tag{17}$$

$$b = -2B\mu(A + E) \tag{18}$$

$$c = -\mu^2 B^2 \tag{19}$$

where μ is the gravitational parameter of the focal body of the orbiting body

Step 8 Find the root of the scalar distance polynomial for the second observation of the orbiting body:

$$(r_2)^8 + a(r_2)^6 + b(r_2)^3 + c = 0 \tag{20}$$

where r_2 is the scalar distance for the second observation of the orbiting body

Step 9 Calculate the slant range, the distance from the observer point to the orbiting body at their respective time

$$p_1 = \frac{1}{D_0}[\frac{6(D_{31}\frac{\tau_1}{\tau_3} + D_{21}\frac{\tau}{\tau_3}(r_2)^3 + D_{31}\mu(\tau^2 - (\tau_1)^2)\frac{\tau_1}{\tau_3}}{6(r_2)^3 + \mu(\tau^2 - (\tau_3)^2)} - D_{11}] \tag{21}$$

$$p_2 = A + \frac{B\mu}{(r_2)^3} \tag{22}$$

$$p_3 = \frac{1}{D_0}[\frac{6(D_{13}\frac{\tau_3}{\tau_1} - D_{23}\frac{\tau}{\tau_1}(r_2)^3 + D_{13}\mu(\tau^2 - (\tau_3)^2)\frac{\tau_3}{\tau_1}}{6(r_2)^3 + \mu(\tau^2 - (\tau_1)^2)} - D_{33}] \tag{23}$$

Step 10 Calculate the orbiting body position vectors, by adding the observer position vector to the slant direction vector (which is the slant distance multiplied by the slant direction vector) with n is run from 1 to 3

$$r_n = R_n + p_n . \hat{p_n} \tag{24}$$

Step 11 Calculate four Lagrange coefficients

$$f_1 \approx 1 - \frac{(\tau_1)^2 \mu}{2(r_2)^3} \tag{25}$$

$$f_3 \approx 1 - \frac{(\tau_3)^2 \mu}{2(r_2)^3} \tag{26}$$

$$g_1 \approx \tau_1 - \frac{(\tau_1)^3 \mu}{6(r_2)^3} \tag{27}$$

$$g_3 \approx \tau_3 \frac{(\tau_3)^3 \mu}{6(r_2)^3} \tag{28}$$

Step 12 Calculate the velocity vector for the second observation of the orbiting body

$$v_2 = \frac{f_1 r_3 - f_3 r_1}{f_1 g_3 - f_3 g_1} \tag{29}$$

The orbital state vectors have now been found, the position and velocity vector for the second observation of the orbiting body. With these two vectors, the orbital elements can be found and the orbit determined.

But they are just the theory part, now we will apply them to programming so that the computer will run the results

When we read the fit file, any information of the object printed. We focus on the DATE-OBS, SITELAT and OBJCTHA. DATE-OBS show us the time that object be took, SITELAT presen the latitude (13 43 07 in is case, we will convert it to decial degrees in the next step) while OBJCTHA let us know the altitude (56.4551 in this case, do not need to convert).

Let convert our latitude the decimal degrees

$$DecimalDegrees = Degrees + \frac{minute}{60} + \frac{second}{3600} = 13 + \frac{43}{60} + \frac{7}{3600} = 13.718611 \tag{30}$$

We call latitude is "phi" and altitude is "h" in the code part below

The "f" is the flattening factor of the Earth, a is the semi-major axis (equatorial radius) of the Earth and b is semi-minor axis (polar radius) of the Earth.

$$f = \frac{a-b}{b} = \frac{6378.137 - 6350.572}{6378.137} = 0.003353 \tag{31}$$

The last thing we need mention is three different times. Here we take 13h43, 15h59 and 18h27 as three timelines. There are 136 minutes between the first and second timelines and 284 minutes when compare the first and last timelines.

```python
import numpy as np

todeg = 180 / np.pi
torad = np.pi / 180

phi = 13.718611 * torad
f = 0.003353
h = 56.4551
re = 6378
mu = 398600

theta1 = (12 + 3/60 + 3.2/3600) * 15 * torad
theta2 = (12 + 3/60 + 56.7/3600) * 15 * torad
theta3 = (12 + 4/60 + 42.6/3600) * 15 * torad

ra1 = (7 + 16/60 + 55.04/3600) * 15 * torad
ra2 = (7 + 45/60 + 46.35/3600) * 15 * torad
ra3 = (8 + 17/60 + 38.14/3600) * 15 * torad

dec1 = (-1 - 30/60 - 51.2/3600) * torad
dec2 = (-23 - 11/60 - 35.5/3600) * torad
dec3 = (-43 - 45/60 - 43.4/3600) * torad

t1 = 0
t2 = 136
t3 = 284

def posroot(roots):
    posroots = roots[np.logical_and(roots > 0, np.isreal(roots))].real
    if len(posroots) == 0:
        raise ValueError("There are no positive real roots.")
    return posroots[0]
```

```python
def position_vector(theta):
    a = 1 - (2 * f - f**2) * (np.sin(phi))**2
    b = np.sqrt(a)
    c = re / b
    D = c + h
    e = c * (1 - f)**2
    F = e + h

    Rx = D * np.cos(phi) * np.cos(theta)
    Ry = D * np.cos(phi) * np.sin(theta)
    Rz = F * np.sin(phi)
    R = np.array([Rx, Ry, Rz])

    return R

R1 = position_vector(theta1)
R2 = position_vector(theta2)
R3 = position_vector(theta3)

def rhohat(ra, dec):
    rx = np.cos(dec) * np.cos(ra)
    ry = np.cos(dec) * np.sin(ra)
    rz = np.sin(dec)
    return np.array([rx, ry, rz])

rhohat1 = rhohat(ra1, dec1)
rhohat2 = rhohat(ra2, dec2)
rhohat3 = rhohat(ra3, dec3)

tau1 = t1 - t2
tau3 = t3 - t2
tau = tau3 - tau1

p1 = np.cross(rhohat2, rhohat3)
p2 = np.cross(rhohat1, rhohat3)
p3 = np.cross(rhohat1, rhohat2)
```

Figure 16: Position vector coding part

```python
D0 = np.dot(rhohat1, p1)
D11 = np.dot(R1, p1)
D21 = np.dot(R2, p1)
D31 = np.dot(R3, p1)
D12 = np.dot(R1, p2)
D22 = np.dot(R2, p2)
D32 = np.dot(R3, p2)
D13 = np.dot(R1, p3)
D23 = np.dot(R2, p3)
D33 = np.dot(R3, p3)

A = 1 / D0 * (-D12 * tau3/tau + D22 + D32 * tau1/tau)
B = 1 / (6 * D0) * (D12 * (tau3**2 - tau**2) * tau3/tau + D32 * (tau**2 - tau1**2) * tau1/tau)

E = np.dot(R2, rhohat2)
R2squared = np.dot(R2, R2)

a = -(A**2 + 2 * A * E + R2squared)
b = -2 * mu * B * (A + E)
c = -mu**2 * B**2

r2 = posroot(np.roots([1, 0, a, 0, 0, b, 0, 0, c]))

rho1 = 1 / D0 * ((6 * (D31 * tau1 / tau3 + D21 * tau / tau3) * r2**3
                  + mu * D31 * (tau**2 - tau1**2) * tau1 / tau3)/(6 * r2**3 + mu * (tau**2 - tau3**2)) - D11)
rho2 = A + mu * B / r2**3
rho3 = 1 / D0 * ((6 * (D13 * tau3 / tau1 - D23 * tau / tau1) * r2**3 +
                  mu * D13 * (tau**2 - tau3**2) * tau3 / tau1)/(6 * r2**3 + mu * (tau**2 - tau1**2)) - D33)

r1vec = R1 + rho1 * rhohat1
r2vec = R2 + rho2 * rhohat2
r3vec = R3 + rho3 * rhohat3

print("r1vec:", r1vec)
print("r2vec:", r2vec)
print("r3vec:", r3vec)
```

Figure 17: Position vector coding part

And the final results of position vector is printed out

```
r1vec: [-7478.60163592  3434.64727859  1417.6223589 ]
r2vec: [-7784.4961412   2975.49485656    40.76810242]
r3vec: [-8005.38860559  2433.26509357 -1457.57720859]
```

Figure 18: Position vector results

5 Conclusion and Acknowledgement

5.1 Conclusion

Although due to weather reasons, Jupiter's fast setting and other influencing factors mentioned before, we were not able to capture Jupiter's satellites but luckily we still have images of this planet.

The part about calculating the velocity of an object from the file provided from the Quy Nhon bservatory is quite familiar to us because it applies angular distance as in Dr. PHAN Thanh Hien's previous course (Introduction to astronomy). There may be certain errors when calculating because our code prints angular distance with only 2 digits after the decimal point and we only use hours and minutes without using seconds when calculating the time difference.

We only sampled a few values from the total 44 files sent to calculate the velocity, so the final value will only be close to reality. Note that we are not sure that the data from 13h43 to 18h33 on 7th May is the time the object completed its own orbit, so the final average velocity is just the final value during this time period.

There are many methods to calculate position vector, here we use the gauss method to calculate, the results between many different methods may produce certain errors. The method used in this report also introduces some small errors from the time as we only consider the difference in minutes and do not mention seconds. Other factors such as latitude converted to decimal degrees, altitude and flattening factor of the Earth we have tried to get as accurate as possible from the information in files of Quy Nhon Observatory.

5.2 Acknowledgement

We wish to register my profound gratitude to Dr. Phan Thanh Hien for the guidance and grace throughout the course. We would also like to extend special regards to Dr. Le Quang Thuy for his data from Quy Nhon Observatory. With your help, we were able to have a good grasp of what to do and complete the task assigned to us as a team. They all lead to certain errors when calculating the object's velocity

5.3 References

[1] R. SUSZYNSKI, K. WAWRYN, and M. DZIEBOWSK, "An efficient algorithm for determining positions of astronomical objects in the Deep Sky Object picture", Bulletin of the Polish Academy of Sciences, Technical Sciences 63, September 2015, DOI:10.1515/bpasts-2015-0079

[2] Gauss's method from Wikipedia

[3] Position And Displacement Vectors, BYJUS

`https://byjus.com/physics/position-and-displacement-vectors/`