Virginia Tech

Project X: Verifying EclipseStudy.py is giving correct information

Jackson Andrew

MOSAIC

6/1/25

Contents

[Purpose 3](#_Toc168213955)

[Dependencies 3](#_Toc168213956)

[Test Case GOES 16 3](#_Toc168213957)

[Using EclipseStudyOneAngle.py 3](#_Toc168213958)

[Asynchronous test 4](#_Toc168213959)

[Possible sources of error and long periods of measurement considerations 4](#_Toc168213960)

# Purpose

The purpose of this document is to prove without a shadow of a doubt that EclipseStudy.py can provide accurate information about the angular separation of 2 celestial objects. This is part of the MOSAIC research project which aims to build a software package capable of different satellite tracking based operations. Most of this document will go over 1 test case that was done to verify EclipseStudy.py is working but throughout will have suggestions on how this test might be replicated. Throughout the document I use the terms elevation and altitude interchangeably.

# Dependencies

This study uses a child program to EclipseStudy.py called EclipseStudyOneAngle.py which is like EclipseStudy.py in every way except for only outputting one angle at a user defined time. There is another short program called EclipseStudyAsyncCalc.m which as the name would suggest is an asynchronous calculation done to verify that EclipseStudy.py is working. EclipseStudyAsyncCalc.m works by checking the azimuth and elevation of the satellite in question and the sun from a source *other* than the MOSAIC project. EclipseStudyOneAngle.py and EclipseStudyAsyncCalc.m are both on the MOSAIC GitHub in the Side project folder.

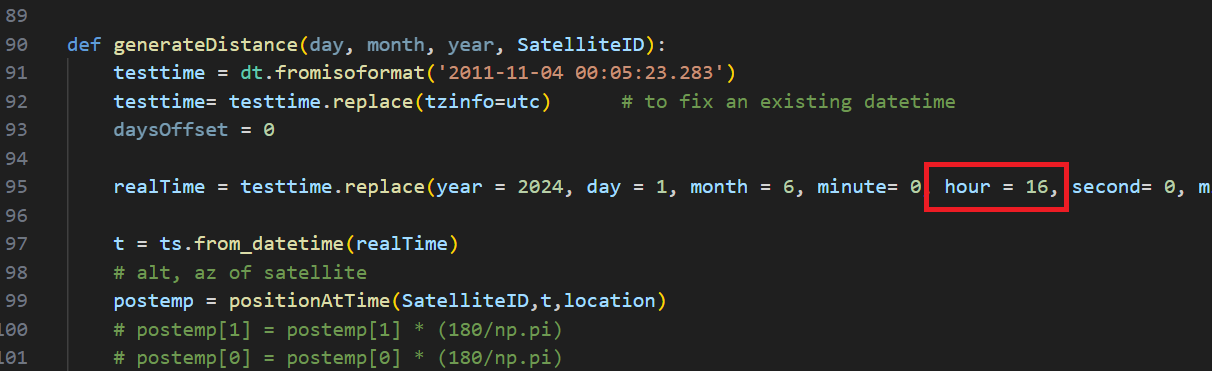
# Test Case GOES 16

To test that EclipseStudy.py was working a test case involving the weather satellite GOES16 was chosen. This test was done on June 1st at 12pm. All dates and times reflect the time that this test was taken and will be discussed in more detail throughout.

## Using EclipseStudyOneAngle.py

A screenshot of a computer

Description automatically generated EclipseStudyOneAngle.py wasn’t designed to be used in any situation other than to test that EclipseStudy.py is working and thus does not use a configuration file. To edit the date and time in EclipseStudyOneAngle.py change the variable declarations at the top of the file. Shown here are the variable declarations for June 1st, 2024, at Blacksburg VA.

 Next the user will want to edit the time for the angular distance calculation. As previously mentioned this test was taken at 12pm. Skyfield exclusively uses UTC so it is necessary to input the UTC value for the time in question into the function generateDistance(). In this case Blacksburg is in UTC-4 so we would add 4 to the time we are interested in. The program is all set and will output the correct angle to the terminal upon running but there is more that this program can check. The angle the program outputs in my case is **31.2 degrees**.

Using this program there are other to check that other aspects of the MOSAIC project are working with clever use of a debugger. I am using the VScode python debugger, but I trust most other debuggers will have similar capabilities. Put a pause on the return statement for generateDistance(). Open the local variable list and check for alt, az and postemp. Alt and az are the altitude and azimuth of the sun in that order. postemp is a short array where index value 0 is GOES16’s altitude and index value 1 is GOES16’s azimuth. Write these numbers down to later compare with other sites the altitude and azimuth of these objects with reputable sources.

## Asynchronous test

The next step to verifying that ElipseStudy.py is working is to do an asynchronous test. Navigate to N2YO.com. Track the same satellite that was chosen in the last step, in my case it is GOES 16. Note down the azimuth and elevation at the current time. If this time is largely different from the time used in the previous step complete the previous step again with the current time. During my test case I got an azimuth of 171.4 degrees and an elevation of 46.5 degrees. Next navigate to suncalc.org and set the time to the same time as the tracked satellite. Note the azimuth and elevation. In my test case I got an azimuth of 121.6 degrees and an elevation of 66.8 degrees. Use EclipseStudyAsyncCalc.m to calculate the angular distance. If a user wants to directly calculate the angular distance without EclipseStudyAsyncCalc.m the formula for angular distance can be found [here](https://en.wikipedia.org/wiki/Angular_distance) just make sure not to use any of the approximations. Using EclipseStudyAsyncCalc.m I get an angular distance of **32.7 degrees.**

## Possible sources of error and long periods of measurement considerations

While EclipseStudy.py is accurate to about 1 degree which should be useful as an estimate for distance the error is significant enough to warrant discussion. Possible sources of error include rounding by the user and tracking websites. Another source of error is the useful period of TLEs. Using all known digits in the asynchronous calculation gives an angular separation of 32.6758 degrees which is very slightly closer to the calculated angular separation. Another error source could be embedded within the fact that TLEs have epochs where measurements of satellite position are most accurate the closer the real time is the TLEs epoch. In this case the epoch for the GOES 16 was June 1st so the position of the satellite was very accurate. Estimating over long periods of time would mean that the TLE becomes more inaccurate the further away the observed time is from the satellite’s epoch.

# Conclusion

EclipseStudy.py can be expected to be accurate within about 1 degree when the TLE epoch is very close to the observed time. Without testing, it would be hard to make an estimate of how inaccurate angular separation becomes over time, but my hypothesis is that the angular distance would be within 3 degrees of the actual angular separation. This would mean that EclipseStudy.py is a valid estimate of angular distance between 2 celestial objects.