DEVELOPING A LIGHT CURVE SIMULATION TOOL FOR GROUND AND SPACE-BASED OBSERVATIONS OF SPACECRAFT AND DEBRIS

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Aerospace Engineering

by

Andrew Tiger Ochoa

December 2021
COMMITTEE MEMBERSHIP

TITLE: Developing a Light Curve Simulation Tool for Ground and Space-Based Observations of Spacecraft and Debris

AUTHOR: Andrew Tiger Ochoa

DATE SUBMITTED: December 2021

COMMITTEE CHAIR: Kira Abercromby, Ph.D.
Professor of Aerospace Engineering

COMMITTEE MEMBER: Eric Mehrel, Ph.D.
Professor of Aerospace Engineering

COMMITTEE MEMBER: Pauline Faure, Ph.D.
Assistant Professor of Aerospace Engineering

COMMITTEE MEMBER: David Mitchell, Ph.D.
Professor of Physics
ABSTRACT

Developing a Light Curve Simulation Tool for Ground and Space-Based Observations of Spacecraft and Debris

Andrew Tiger Ochoa

A light curve is a plot of brightness versus time of an object. Light curves are dependent on orbit, attitude, surface area, size, and shape of the observed object. Using light curve data, several analysis methods have been developed to derive these parameters. These parameters can be used for tracking orbital debris, monitoring satellite health, and determining the mission of an unknown spacecraft.

This paper discusses the development, verification, and utilization of a tool that simulates light curve data. This tool models ground-based observations, space-based observations, self-shadowing geometry, tumbling debris, and controlled spacecraft. The main output from the tool is the pass prediction plot and the light curve plot. The author intends to publish the tool and supporting documents for future researchers to utilize. This will save researchers time developing their own models and the tool can act as a baseline for comparisons between analysis methods. For clarity, this paper does not develop nor implement a light curve analysis method, but rather creates a tool to simulate light curve observations and data.

Each section of the tool was verified independently to ensure that the simulated light curves were correct. The tool was verified with STK, matlab, and simulink. It predicts the start and end times of passes, eclipses, and ground-site night cycles within 1% of the total event duration, when compared to STK. The attitude propagator predicts the attitude of the target with offsets less than 0.06° on average and a maximum offset less than 0.6° when compared to provided attitude code.
ACKNOWLEDGMENTS

Thanks to:

• My Parents, for always supporting and loving me. I couldn’t have done it without you. I hope I make you guys proud.

• Lexy, for reading my papers, listening to my presentations, and always giving me an excuse to eat ice cream.

• Papa and Nana, for always making me laugh and buying me that belt. You have truly supported me in more than one way.

• Lauren for reminding me why I chose to do this to myself.

• Dr. A for being an amazing professor, thesis advisor, and person. I had a great time in your six classes and I’m honored to be one of your many grad students.

• Roman, Gabriel, and Alex for keeping me laughing, giving me a reason to come back to Hemet, and throwing burritos with me.

• Dipper for being a good boy and helping me distract Lexy while she did big boy work from home.
# TABLE OF CONTENTS

| LIST OF TABLES | xi |
| LIST OF FIGURES | xiii |

## CHAPTER

1. Introduction ............................................. 1
2. Background ............................................. 3
   2.1 Light Curve Observations ......................... 3
      2.1.1 Ground-Based Observations .................. 4
      2.1.2 Space-Based Observations .................. 7
      2.1.3 Space-Based Observation Missions .......... 9
      2.1.4 Observation Summary ....................... 11
2.2 Light Curve Utilization .............................. 12
2.3 Reflection Models ..................................... 13
2.4 Existing Tools ........................................ 14
2.5 Coordinate Frames .................................... 17
2.6 Modified Rodriguez Parameters .................... 19
2.7 Project Definition .................................... 20
3. Methodology ............................................ 21
   3.1 Assumptions ........................................ 21
   3.2 Model Overview ..................................... 22
   3.3 User Definitions ................................... 23
      3.3.1 Simulation Requirements .................... 26
      3.3.2 Target Definition ............................. 27
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3.3</td>
<td>Observer Definition</td>
<td>28</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Attitude Profile</td>
<td>29</td>
</tr>
<tr>
<td>3.3.4.1</td>
<td>Random Tumbling</td>
<td>29</td>
</tr>
<tr>
<td>3.3.4.2</td>
<td>Defined Tumbling</td>
<td>30</td>
</tr>
<tr>
<td>3.3.4.3</td>
<td>Controlled</td>
<td>31</td>
</tr>
<tr>
<td>3.3.5</td>
<td>User Definition Summary</td>
<td>31</td>
</tr>
<tr>
<td>3.4</td>
<td>Pass Prediction</td>
<td>34</td>
</tr>
<tr>
<td>3.5</td>
<td>Attitude Propagation</td>
<td>38</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Tumbling</td>
<td>39</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Controlled</td>
<td>40</td>
</tr>
<tr>
<td>3.6</td>
<td>Reflection Calculations</td>
<td>42</td>
</tr>
<tr>
<td>3.6.1</td>
<td>Phong BRDF</td>
<td>43</td>
</tr>
<tr>
<td>3.6.2</td>
<td>Ray Tracing</td>
<td>45</td>
</tr>
<tr>
<td>3.7</td>
<td>Output</td>
<td>46</td>
</tr>
<tr>
<td>4</td>
<td>Verification</td>
<td>50</td>
</tr>
<tr>
<td>4.1</td>
<td>Pass Prediction</td>
<td>50</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Ground-Based</td>
<td>51</td>
</tr>
<tr>
<td>4.1.1.1</td>
<td>Verification Plan</td>
<td>51</td>
</tr>
<tr>
<td>4.1.1.2</td>
<td>Results</td>
<td>52</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Space-Based</td>
<td>56</td>
</tr>
<tr>
<td>4.1.2.1</td>
<td>Verification Plan</td>
<td>56</td>
</tr>
<tr>
<td>4.1.2.2</td>
<td>Results</td>
<td>57</td>
</tr>
<tr>
<td>4.1.3</td>
<td>Illumination Requirements</td>
<td>61</td>
</tr>
<tr>
<td>4.1.3.1</td>
<td>Verification Plan</td>
<td>61</td>
</tr>
<tr>
<td>4.1.3.2</td>
<td>Eclipse Results</td>
<td>63</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Summary of space-based observation missions.</td>
<td>11</td>
</tr>
<tr>
<td>2.2</td>
<td>Summary of existing tools.</td>
<td>17</td>
</tr>
<tr>
<td>3.1</td>
<td>Summary of assumptions.</td>
<td>22</td>
</tr>
<tr>
<td>4.1</td>
<td>Summary of test objects for ground-based pass prediction.</td>
<td>51</td>
</tr>
<tr>
<td>4.2</td>
<td>Pass prediction verification requirements.</td>
<td>52</td>
</tr>
<tr>
<td>4.3</td>
<td>Ground-based pass prediction average percent offsets, with a requirement of 1% or less.</td>
<td>56</td>
</tr>
<tr>
<td>4.4</td>
<td>Summary of test objects for space-based pass prediction.</td>
<td>57</td>
</tr>
<tr>
<td>4.5</td>
<td>Space-based pass prediction average percent offsets, with a requirement of 1% or less.</td>
<td>61</td>
</tr>
<tr>
<td>4.6</td>
<td>LEO Cosmos eclipse average percent offset.</td>
<td>63</td>
</tr>
<tr>
<td>4.7</td>
<td>Illumination requirements success criteria.</td>
<td>63</td>
</tr>
<tr>
<td>4.8</td>
<td>LEO Cosmos eclipse average percent offsets, with a requirement of 1% or less.</td>
<td>64</td>
</tr>
<tr>
<td>4.9</td>
<td>Ground site darkness average percent offsets, with a requirement of 1% or less.</td>
<td>65</td>
</tr>
<tr>
<td>4.10</td>
<td>Average Sun pointing offset with a requirement of 0.5° or less.</td>
<td>66</td>
</tr>
<tr>
<td>4.11</td>
<td>Summary of test cases for tumbling attitude verification.</td>
<td>68</td>
</tr>
<tr>
<td>4.12</td>
<td>Attitude verification requirements.</td>
<td>69</td>
</tr>
<tr>
<td>4.13</td>
<td>Average offsets for a target with tumbling attitude, with a requirement of 1e-3 or less.</td>
<td>71</td>
</tr>
<tr>
<td>4.14</td>
<td>Maximum offsets for a target with tumbling attitude, with a requirement of 1e-2 or less.</td>
<td>72</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>4.15</td>
<td>Summary of test cases for controlled attitude motion verification.</td>
<td>72</td>
</tr>
<tr>
<td>4.16</td>
<td>Average controlled attitude offsets for all cases, with a requirement of 1e-3 or less.</td>
<td>76</td>
</tr>
<tr>
<td>4.17</td>
<td>Maximum controlled attitude offsets for all cases, with a requirement of 1e-2 or less.</td>
<td>77</td>
</tr>
<tr>
<td>4.18</td>
<td>Summary of test cases for reflection coefficients.</td>
<td>78</td>
</tr>
<tr>
<td>C.1</td>
<td>Space-based average tumbling attitude offsets for all cases, with a requirement of 1e-3 or less.</td>
<td>109</td>
</tr>
<tr>
<td>C.2</td>
<td>Space-based maximum tumbling attitude offsets for all cases, with a requirement of 1e-2 or less.</td>
<td>109</td>
</tr>
<tr>
<td>C.3</td>
<td>Average controlled attitude offsets for all space-based cases with a requirement of 1e-3 or less.</td>
<td>111</td>
</tr>
<tr>
<td>C.4</td>
<td>Maximum controlled attitude offsets for all space-based cases with a requirement of 1e-2 or less.</td>
<td>111</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Example of a light curve observation.</td>
<td>3</td>
</tr>
<tr>
<td>2.2</td>
<td>Example of a ground-based pass of the ISS [29]</td>
<td>5</td>
</tr>
<tr>
<td>2.4</td>
<td>Light curve observed by Andersen [30].</td>
<td>7</td>
</tr>
<tr>
<td>2.5</td>
<td>Example of a space-based pass [19].</td>
<td>8</td>
</tr>
<tr>
<td>2.6</td>
<td>Artist rendition of NEOSSat [15]</td>
<td>10</td>
</tr>
<tr>
<td>2.7</td>
<td>Orbit and observation concept for SBSS. The direction of illumination from the Sun is shown in orange [14].</td>
<td>11</td>
</tr>
<tr>
<td>2.8</td>
<td>Example of specular reflection on the left and diffuse reflection on the right [12].</td>
<td>14</td>
</tr>
<tr>
<td>2.9</td>
<td>Example of the body frame for the box wing.</td>
<td>18</td>
</tr>
<tr>
<td>3.1</td>
<td>Overview of the model steps.</td>
<td>23</td>
</tr>
<tr>
<td>3.2</td>
<td>Example of a .json configuration file for a ground-based observation of a tumbling cylinder.</td>
<td>25</td>
</tr>
<tr>
<td>3.3</td>
<td>Example of the simulation requirements for a space-based pass defined in a .json configuration file.</td>
<td>26</td>
</tr>
<tr>
<td>3.4</td>
<td>Example of target definitions for a MEO box wing satellite.</td>
<td>27</td>
</tr>
<tr>
<td>3.5</td>
<td>Example user input for a ground-based observer.</td>
<td>28</td>
</tr>
<tr>
<td>3.6</td>
<td>Example user input for a space-based observer.</td>
<td>29</td>
</tr>
<tr>
<td>3.7</td>
<td>Example input for tumbling target attitude motion with random starting attitude and angular velocity.</td>
<td>30</td>
</tr>
<tr>
<td>3.8</td>
<td>Example input to define tumbling attitude motion for the target starting at a defined quaternion.</td>
<td>30</td>
</tr>
</tbody>
</table>
4.4 MEO Breeze to Cal Poly pass time offsets. .......................... 55
4.5 STK plot of space-based verification orbits, with NEOSsat in white,
Minotaur in red, Breeze in purple, and AMC 12 in green. ............ 57
4.6 Leo Cosmos to Leo NEOSSat pass time offsets. ................. 58
4.7 MEO Breeze to LEO NEOSSat pass time offsets. ............... 59
4.8 GEO AMC to LEO NEOSSat pass time offsets. .................. 60
4.9 Light curves resulted from rotating the Sun pointing vector 0.5° in
all three axes. ..................................................... 62
4.10 Eclipse start and end offsets for LEO Cosmos. ................. 64
4.11 Ground site darkness time offsets. .............................. 65
4.12 Sun pointing offsets in degrees. ................................. 66
4.13 Tumbling attitude offsets for Case 2. ............................ 70
4.14 Tumbling attitude offsets for Case 1. ............................ 71
4.15 Controlled attitude offsets for Case 1. .......................... 74
4.16 Controlled attitude offsets for Case 2. .......................... 75
4.17 Controlled attitude offsets for Case 3. .......................... 76
4.18 Specular Coefficient comparison with total counts on the top and
normalized counts on the bottom. ................................ 79
4.19 Diffuse Coefficient comparison with total counts on the top and nor-
malized counts on the bottom. ..................................... 80

5.1 Light curve result for all six different premade geometries. .... 82
5.2 Tumbling vs controlled light curve. ............................... 84
5.3 Light curve observation of nadir pointing SXM-8 observed from Cal
Poly. ................................................................. 86
5.4 Light curve observation of nadir pointing SXM-8 observed from NEOSSat. 86
5.5 Ground and Space-based observation of SXM-8 on same time scale. 87
5.6 Example of simulation function call out. .......................... 88
Chapter 1

INTRODUCTION

A light curve is a plot of brightness versus time of an object as seen by an observer. The observed brightness of the object is dependent on the attitude, ephemeris, material properties, size, and shape of the object [1]. Because light curves are dependent on these factors, researchers have developed analysis methods to derive one or more of these properties from light curve observations of spacecraft and debris [4]. These analysis methods can be used for tracking orbital debris, satellite health monitoring, and to determine the mission of an unknown spacecraft.

There are two ways for researchers to obtain light curve data, either through simulations or visual observations of space objects. This is true for both ground-based observations and space-based observations. However, most authors focus on simulated light curve data rather than observed light curve data [3][5][6][7][13][14][15][16]. If they find a solution using observed data, it is almost impossible to verify because generally, they do not know the true state of the resident space object (RSO). The researchers can not prove that their analysis method is successful without knowing the true attitude, size, shape, and material properties of the RSO [7]. This information is so limited because they often observe debris and decommissioned satellites that do not have functioning attitude sensors and have materials that have been effected by the space environment. With simulated light curve data, on the other hand, the researchers can know the true state of the RSO because they defined that state in the simulation.
However, before a researcher can test their analysis method with simulated light curve data, they must first obtain a simulation tool [1][5][6][7][13][14][15][16]. While these simulation tools are necessary to develop an analysis method, the author has not found a similar simulation tool available online. That means every time a researcher aims to develop an analysis method they must first develop a simulation tool. The development of a tool takes time from the researchers and each model is different, which makes it hard to compare results.

This thesis explores the development, verification, and utilization of a light curve data simulation tool that models ground-to-space and space-to-space observations, tumbling debris, nadir pointing spacecraft, and self-shadowing geometry. With the intent of making the tool and supporting documents publicly available, this work will save researchers time developing their own simulations and the tool can act as a new baseline for comparing analysis methods. For clarity, this paper does not develop nor implement an analysis method, but rather creates a tool to simulate light curve observations and data.

This is a legacy project following Rush’s “An Application of the Unscented Kalman Filter for Spacecraft Attitude Estimation on Real and Simulated Light Curve Data” [1]. Their thesis and code provided a base for the author to build from.
Chapter 2

BACKGROUND

In this chapter methods to observe, utilize, and simulate light curves will be explored. Information in this section helped the author produce and verify the light curve simulation tool.

2.1 Light Curve Observations

An example of a light curve is shown in Figure 2.1. This figure has counts, which are the number of photons received by the sensor during a measurement, on the Y-axis with the time since the beginning of the observation in minutes on the X-axis. More counts means a brighter object.

![Figure 2.1: Example of a light curve observation.](image)
A count is measured by a charge coupled devices, CCD, in the sensor. Each count represents an amount of photons that are detected by a CCD pixel [1]. The number of photons per count is defined by the CCD gain [1]. If a sensor has a CCD gain of 5, for example, 1 count would translate to 5 photons detected.

Light curve data can be collected by one of two ways. The first method is achieved by tracking the object and taking several images, which makes the object a single point in the frame, while all other objects are streaks. In the second method, the observer is kept inertially pointed and takes a long exposure image of an object as it passes making a streak across the frame [20]. Both methods will produce the same light curve. Either method can be performed from a ground-based telescope or from sensors on a space-based satellite, both of which have their own requirements and benefits.

2.1.1 Ground-Based Observations

For a ground-based observation to be possible, three pass requirements must be met. The target must be illuminated by the Sun, the target must be above the horizon, and the ground site must be in darkness [1]. To predict a ground-based pass, the location of the target object, ground site, and the Sun are propagated, while checking these three requirements at each time step. Figure 2.2 shows an example of a ground-based pass of the ISS [29]. In this case the ISS rises from the southwest and sets in the northeast with a maximum elevation of 66°.
Figure 2.2: Example of a ground-based pass of the ISS [29]

In this case, the ISS is visible once it rises above 10°, when all three pass requirements are satisfied. It is no longer visible once it sets below 31°, when it enters eclipse and is no longer illuminated by the Sun. As the ISS passes over the observations site, its brightness will vary. The attitude, location, material properties, size and shape of the target, location of the Sun, location of the observer, and noise from the atmosphere all affect the apparent brightness of the target. LEO passes like this are sometimes visible by the naked eye, but telescopes and cameras are required to observe higher altitude objects and obtain accurate data.

In 2014, Andersen used the Cal Poly observatory to take light curve observations of tumbling rocket bodies in geostationary transfer orbits [30]. Their main instrumentation included a telescope, CCD camera, and TLE tracking software. Figure 2.3 shows the CCD camera mounted to the end of the telescope [30].
Figure 2.3: Observation equipment used by Andersen. Left: telescope. Top middle: looking down the telescope. Top right remote to orientate telescope. Bottom right: CCD camera. [30]

Over several nights Andersen imported TLEs to the tracking software, aligned the telescope, and took long exposure images of rocket body debris as it passed overhead. To produce accurate light curves, several calibrations were taken before each observation. This included taking bias images to calibrate the CCD camera. After the observations were complete, data processing and data reduction steps were taken to convert the long exposure images to light curves. These processes are explained in detail in their Data Reduction chapter. Figure 2.4 shows a light curve observed by Andersen.
Figure 2.4: Light curve observed by Andersen [30].

Ground-based light curves can be collected with existing astronomical observation architecture. However, observations are limited by the time of day and weather. Furthermore, it is difficult to get useful light curve data for GEO objects. Light curve analysis methods require fluctuation in the brightness of the target, but because GEO objects are stationary relative to ground sites, both transitionally and often rotationally, analysis methods often fail to find a solution [14].

2.1.2 Space-Based Observations

Space-based observations also have three requirements. The target must be illuminated by the Sun, outside the Earth exclusion angle, and outside the Sun exclusion angle [8]. An exclusion angle is the minimum angle between the vector from the observer to the target and the vector from the observer to the surface of the celestial body for the pass to be valid, this can be seen in Figure 2.5. This angle is important because the Earth and Sun are brighter than the target and will obstruct observations. The magnitude of the exclusion angles will depend on the sensitivity of the camera
and the relative brightness of the target [13]. Figure 2.5 shows an example of a valid pass, in this case the Earth exclusion angle is 35° [19]. To predict a space-based pass, the location of the target object, observation spacecraft, and the Sun are propagated, while checking these three requirements at every time step.

![Diagram showing a space-based pass](image)

**Figure 2.5: Example of a space-based pass [19].**

Unlike ground-based observation platforms, space-based platforms are not limited by time of day or weather, and provide more useful light curve data for GEO objects. A study by Silha et al. found that a single LEO satellite in a Sun synchronous orbit (SSO) can observe 98% of all the objects in GEO and nearby graveyard orbits in a 24 hour period [14]. Furthermore, Wallace et al. found that a quickly moving observer, such as a LEO spacecraft, benefited attitude estimation methods [16]. Because the LEO satellite is moving relative to the GEO target, the observer can see the target from more angles and sometimes for longer, when compared to ground-based observations. Although dedicated satellites may be more expensive than ground-based telescopes, it is a practical mission that has been flown several times.
2.1.3 Space-Based Observation Missions

The Midcourse Space Experiment or MSX was one of the first space-based light curve observation satellites [10]. It was launched by the Department of Defense in 1996 into a Sun synchronous low Earth orbit. It was tasked with surveillance of RSOs from a space-based platform and utilized the Space-Based Visible sensor (SBV) to record light curve data [10]. The MSX had an attitude accuracy of 4arcseconds. The SBV had a field of view (FOV) of about 6.6° × 1.4° with an Earth exclusion angle of 25° [10]. This data helped the author make assumptions about the space-based observer and served as a starting point for the Earth exclusion angle. With a high pointing accuracy and relatively large FOV for the sensor, the MSX could track objects in Earth orbits and deep space. Within its first eight months of operation the MSX successfully took over 1000 observations of 700 RSOs [26]. It was able to determine the orbit of objects within 50 meters of the predictions from the ground-based Space Surveillance Network, proving that a space-based platform can provide accurate orbit determination data. The SBV sensor ceased operation in 2008 after 12 years of operation [27].

The Department of Defense plans for the Space Based Surveillance System (SBSS) to be a constellation of satellites as a follow up mission after the MSX [27]. Again the SBSS will be placed in a Sun-synchronous orbit with an altitude of 700km [14]. Silha et al. provided analysis on the proposed SBSS, which will track objects in GEO, MEO, and LEO orbits [14]. Silha et al. found that in a 25 hour period about 110,000 LEO tracks could be obtained by the SBSS. However, not all of them could be used for orbit determination because many passes were too short or not bright enough.

NEOSSat is a more modern example of a space-based observation platform. Figure 2.6 shows a rendition of the microsatellite [15]. It was launched by Canada in 2013, into
a Sun synchronous low Earth orbit with the mission of surveillance of resident space objects. Its main sensor is an optical camera that collects light curve data [15]. It has a pointing accuracy of $2\text{arcseconds}$ and can track objects for about $1.5\text{mins}$ to $2\text{mins}$. NEOSSat’s camera has a field of view of $6^\circ \times 6^\circ$. Data from NEOSSat’s observations are used to monitor the orbits of debris to predict possible collisions [15]. After two tracks of a GEO target, NEOSSat can determine the orbit of the target with uncertainties as low as $5\text{km}$ [25]. Even after suffering a magnetometer failure in 2016, NEOSSat is still in operation today taking observations of RSOs.

Figure 2.6: Artist rendition of NEOSSat [15]

All of these satellites are in an Sun-synchronous orbits because it provides the observation satellite with a constant view of illuminated targets. As shown in Figure 2.7, the observation satellite is pointed in the opposite direction of the Sun to reach the maximum illumination of the target [14].
The attitude, payload, and orbital information from all three satellites helped the author make assumptions for the model, which is explained in the Methodology chapter. A summary of each mission is shown in Table 2.1. Because NEOSSat is still operational, its orbital information will be used later in the verification section. NEOSSat will serve as this thesis space-based observer, because its orbit was specifically designed to take visible observations of RSOs.

<table>
<thead>
<tr>
<th>Table 2.1: Summary of space-based observation missions.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Launch Date</td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>NEOSSat</td>
</tr>
<tr>
<td>MSX</td>
</tr>
<tr>
<td>SBSS</td>
</tr>
</tbody>
</table>

### 2.1.4 Observation Summary

In summary, both ground and space-based observations are viable methods for collecting light curve data. Space-based observations are not limited by time of day or weather and collect data on GEO objects more effectively, but ground-based observations are more readily available. Although ground and space-based observations
are different, their requirements are effectively the same. For either observation to be possible, the target must be in the line of site of the observer, the target must be illuminated, and the observing platform can not be looking towards the Sun. These requirements are necessary to develop a pass prediction model.

2.2 Light Curve Utilization

Because the brightness of an object is highly dependent on the attitude, ephemeris, material properties, size, and shape of the object, analysis methods are applied to light curve data to estimate one or more of these parameters [1]. Understanding how light curves are utilized will aid the development of a useful simulation tool.

One of the most common analysis methods is applying a Kalman filter to the light curve data to estimate the target’s attitude [1]. The Kalman filter iterates its solution until it reaches a minimum error. Most authors only use simulated data because real data tends to be noisy, which makes it difficult for the filter to settle on a solution [1]. Even if the Kalman filter does find a solution, it is almost impossible to verify because the true state of an observed object is rarely known [3]. The true attitude, material properties, or even size and shape of the object being observed are unknown. The main benefit of using simulated data is that the true state of the target is always available to compare to because it was defined by the user in the simulation. Rather than using observation data, Jah et al. believes the best way to improve analysis methods is by “incrementally increasing the complexity of simulations until they are close to real data” [3]. Complexity in this sense means modeling articulating solar panels, or a constantly nadir pointing spacecraft, or even different materials on the target that have different reflective properties. Several authors start by testing relatively simple scenarios like a homogeneous flat plate or
cylinders tumbling in space [3][5][6][7]. Including more variables would increase the state space that the Kalman filter solves for and make the problem more indeterminate [1]. Therefore, a hyperrealistic simulation may not be the best tool in this case. A simulation tool that can be built upon until it is close to real data will aid researchers more.

Because there are little to no publicly available light curve simulation tools, often before the researcher can test their analysis method, they must first develop a model to produce test data. This was the case for Rush, Linares et al. and Wetterer et al., to name a few [1][5][6][7][13][14][15][16]. The development of a model takes time from the researchers and each one is different. Differences ranged from large details like orbit propagators and reflection models, to smaller details like geometry and ephemeris. A widely available tool would save researchers time and help set a baseline to make it easier for analysis methods to be compared.

In summary, a publicly available tool that can be built upon and provides options to increase the complexity of the simulation may be the most useful to researchers.

2.3 Reflection Models

Reflection models are used in simulation tools to calculate the brightness of an object. One of the most commonly used reflection models is the Phong Bi-directional Reflectance Distribution Function, or BRDF, developed by Ashikhmin et al. [2]. This model was chosen by several authors, including Linares et al. [5][6] and Rush [1]. The model obeys the law of energy conservation, accounts for Fresnel’s law, has a non-constant diffuse term, and allows for anisotropic reflection, which means that the reflection has different properties when measured in different directions [2]. The Phong BRDF calculates the amount of reflected light from the specular and the diffuse
components separately [2]. Diffuse light is scattered in all directions, while specular light is reflected in a certain direction [12]. Figure 2.8 shows an example of specular reflection on the left and diffuse reflection on the right [12].

![Figure 2.8: Example of specular reflection on the left and diffuse reflection on the right [12].](image)

The Phong BRDF calculates the brightness of a surface. The model has a method to check if the target is illuminated by the Sun and visible to the observer, but it assumes that the target is a convex shape [2]. This means that the target does not have self shadowing geometries, which is often untrue. For example, satellites can have solar panels that may cast shadows or occlude the observer’s line of sight. To account for this, a ray tracing algorithm was developed by Kaasalainen et al. [4] and utilized by Rush [1]. The Phong BRDF and Kaasalainen ray tracing algorithm will be used in this thesis as well. The algorithm breaks the surface into small flat facets and places points at the center of each facet. Then each point is tested to check if it is illuminated and visible. If both are true, then that facet is added into the reflection calculations.

2.4 Existing Tools

A few papers were found exploring current light curve simulation tools. Each tool will be summarized and compared to the tool developed in this thesis.
Singletary et al. discussed their modifications to existing ground-based observation code provided by the Air Force [8]. They included the capability of simulating a space-based observer and self-shadowing geometries. They used the Cook and Torrance BRDF for their reflection model, the Simplified General Perturbations propagator (SGP4) for an orbit propagator, and the Qhull algorithm, a ray tracing algorithm to account for self-shadowing geometries. They were successful in getting the model working however, they did not post their code. Although this thesis uses the same orbit propagator, a different reflection model and ray tracing algorithm are used, and this thesis is coded in python rather than matlab. Also this thesis includes the option to model actively controlled targets.

Willison et al. developed a ground-based observation model with a spectral bidirectional reflectance distribution function (sBRDF) [17]. This tool can import CAD files into the model and use that geometry for the reflection calculations. Singletary et al., Rush, and the author of this thesis all defined the geometry in code, which is significantly less intuitive than CAD [8][1]. Importing CAD files allows the user to define the geometry in detail without relying on premade geometries. The sBRDF is different from a BRDF because it calculates the amount of reflected light across a range of wavelengths. Using look up tables for different materials properties, it can report the intensity of reflected light in different colors. While this is impressive, researchers generally focus on the total intensity of reflected light rather than which colors are most intense. The sBRDF model opens the possibility for new analysis methods to be developed, but would require more information about the target and a larger state space to be solved for. Thus this tool may not be useful to researchers who do not aim to add color into their calculations. Willson et al. also did not publish their code. The tool developed in this thesis is different because it uses a BRDF rather than a sBRDF, includes space-based observations, and can model tumbling and controlled targets.
Rush [1], Jah et al. [3], Linares et al. [5][6], and Wetterer et al. [7] all developed light curve simulation models to produce test data for their light curve analysis methods. The first three used the Phong BRDF, while Wetterer et al. used the Cook and Torrance BRDF. Rush simulated self-shadowing geometry, while the others used convex shapes like flat plates and cylinders. All focused on ground-based observers, and none published their models. This thesis utilizes the reflection calculation code and makes modifications to the ground pass prediction code from Rush [1]. However this thesis is unique because it adds the capability to simulate space-based observations, actively controlled targets, and makes the code into a tool that other researchers can use.

A summary of the existing tools is shown in Table ??, Unfortunately, none of these reports provided enough detail about their simulations, so a meaningful comparison could not be made with the tool developed in this thesis. The reports were lacking details like the geometry of the target, material properties of the target, sensor parameters, and orbital information. Therefore, instead of using these models to verify the tool, each section of the code will be verified with other resources. If each section of the code passes verification, then the tool as a whole is verified. The author understands that it is possible for an interface error between each section of the code to exist, but the interfaces are only variable names for function inputs that were all checked extensively. With the available resources, this was the best verification method that could be achieved. This will be discussed in greater detail in the Verification chapter.
<table>
<thead>
<tr>
<th>Observer</th>
<th>Reflection Model</th>
<th>Ray Tracing</th>
<th>Public?</th>
</tr>
</thead>
<tbody>
<tr>
<td>This Thesis</td>
<td>Ground and Space</td>
<td>Phong BRDF</td>
<td>Kaasalainen</td>
</tr>
<tr>
<td>Singletary</td>
<td>Ground and Space</td>
<td>Cook and Torrance</td>
<td>Qhull</td>
</tr>
<tr>
<td>Willison</td>
<td>Ground</td>
<td>SBRDF</td>
<td>N/A</td>
</tr>
<tr>
<td>Rush</td>
<td>Ground</td>
<td>Phong BRDF</td>
<td>Kaasalainen</td>
</tr>
<tr>
<td>Jah</td>
<td>Ground</td>
<td>Phong BRDF</td>
<td>N/A</td>
</tr>
<tr>
<td>Linares</td>
<td>Ground</td>
<td>Phong BRDF</td>
<td>N/A</td>
</tr>
<tr>
<td>Wetterer</td>
<td>Ground</td>
<td>Cook and Torrance</td>
<td>N/A</td>
</tr>
</tbody>
</table>

2.5 Coordinate Frames

This thesis utilizes several reference frames that are important to define. The Earth Centered Inertial, or ECI, frame is defined as usual with the origin at the center of Earth with the positive X-axis pointed in the direction of the vernal equinox, the Z-axis is aligned with the North Pole, and the Y-axis completes.

Similarly, the Earth Centered Earth Fixed, ECEF, frame is defined as usual with the origin at the center of Earth with the positive X-axis pointed towards the intersection of the prime meridian and the equator, the Z-axis aligned with the North Pole, and the Y-axis completes.

The Body frame is attached to the object and has its origin at the centroid of the satellite. The Body frame is aligned with the principal axes of the satellite if its mass was evenly distributed. Figure 2.9 shows the body frame for the premade geometry, box wing, which represents a satellite with solar panels. The geometry and body frame for each of the six premade geometries can be found in the appendix.
Figure 2.9: Example of the body frame for the box wing.

In this thesis the Local Vertical Local Horizontal, LVLH, frame follows the definition usually associated with controls. The LVLH frame origin is at the centroid of the satellite with the positive Z-axis pointing towards the center of Earth, positive X-axis in the velocity direction, and the positive Y-axis completes. The following shows the mathematical description of LVLH frame.

In this thesis a bold variable indicates a vector or a matrix while a non-bold variable is a scalar. Equation (2.1) defines the LVLH Z-axis as a unit vector pointing towards the center of Earth, where $\mathbf{r}$ is the position of the satellite in the ECI frame and $r$ is the magnitude of the position vector.

$$\mathbf{Z} = \frac{-\mathbf{r}}{r} \quad (2.1)$$

Here, equation (2.2) defines the LVLH X-axis as a unit vector pointing in the velocity direction, where $\mathbf{v}$ is the velocity vector of the satellite in the ECI frame.
\[ X = \frac{v}{v} \quad (2.2) \]

The Y-axis completes the LVLH frame, shown in equation (2.3).

\[ Y = Z \times X \quad (2.3) \]

Finally, equation (2.4) combines the previous equations to create the rotation matrix from the ECI frame to the LVLH frame, represented by \texttt{C.LVLH.ECI}.

\[ C.LVLH.ECI = \begin{bmatrix} X & Y & Z \end{bmatrix} \quad (2.4) \]

2.6 Modified Rodriguez Parameters

This thesis uses three different attitude representations: Euler angles, quaternions, and Modified Rodriguez Parameters (MRPs). All three are calculated and reported to allow the user to use which ever representation they choose. MRPs have some benefits over the more commonly used Euler angles and quaternions. Euler angles suffer from singularities that can cause their values to vary drastically with small changes in attitude, while quaternions are redundant, meaning they have four values to represent an attitude in three dimensions. MRPs on the other hand, have three elements and only one singularity near the attitude represented by the quaternion \( 0i + 0j + 0k - 1 \) [22]. At this attitude the MRPs diverge towards infinity. However, this can be avoided by converting to the MRP’s shadow parameter, an equivalent MRP with an inverted magnitude. Equation 2.5 shows how to convert from a MRP
to a shadow MRP, with $p_{\text{shadow}}$ representing the shadow MRP and $p$ representing the diverging MRP.

$$p_{\text{shadow}} = -\frac{p}{p^2} \quad (2.5)$$

Then, if after propagating further the shadow MRP gets too large, equation 2.5 can be implemented again to invert the magnitude. This ensures that MRP never gets near the singularity and results in a consistent representation of the object’s attitude [22].

### 2.7 Project Definition

The purpose of this thesis is to produce a light curve simulation tool that is useful to light curve researchers. The tool will be able to model tumbling debris, controlled spacecraft, and self-shadowing geometry. It will also be able to simulate observations from both ground and space-based observers. With the intent to publish, this tool will save researchers time developing their own simulations and can act as a baseline for comparing analysis methods. For clarity, this paper does not develop nor implement an analysis method, but rather creates a tool to simulate light curve observations and data.
Chapter 3

METHODOLOGY

3.1 Assumptions

The author makes several assumptions about the observer. This thesis assumes that the observer, either on ground or in space, can track the target through the entire pass. Therefore, the angles for the ground-based telescope or the attitude dynamics of the observation satellite are not modeled. If the target is in view, then the observer is tracking it. This assumption may not hold true for space-based observations of MEO or GEO target where they can remain in view of the observation satellite for hours. However, by providing data through the entire pass, the user can parse the pass into the length they desire. It is also assumed that all ground-based observations happened during clear nights with no interference. Although this may not be realistic, a Gaussian distribution for noise is assumed for all reflection calculations to very roughly simulate atmospheric effects and other interference. A standard deviation of 30 counts was modeled by Rush because it reflected the real light curve observation data that they analyzed [1]. The same standard deviation was used in this thesis to model noise.

The target geometry is assumed to be made of small flat facets, and the premade targets are constructed of a homogeneous material. However, it is possible for the user to define the specular and diffuse coefficients of the material. The target is assumed to be fixed in the body frame, for example the solar panels are stationary with respect to the target. The target object is assumed to experience no disturbance
torques while tumbling, but the user can define a constant disturbance torque for controlled space objects.

Both the observation satellite and the target space object were propagated using the Simplified General Perturbation model, or SGP4, provided by the Python Package Index [31]. This model accounts for solar and lunar gravitational effects, Earth oblateness, and drag. At the epoch of the TLE, SGP4 has a positional error of about 1 km and this error grows about 1 km to 3 km per day [28]. All of these assumptions are summarized in Table 3.1.

<table>
<thead>
<tr>
<th>Observer</th>
<th>If the target is in view then the observer is tracking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>Attitude or angles not modeled</td>
</tr>
<tr>
<td></td>
<td>Small flat facets</td>
</tr>
<tr>
<td></td>
<td>Homogeneous material</td>
</tr>
<tr>
<td></td>
<td>Fixed in the body frame</td>
</tr>
<tr>
<td>Attitude</td>
<td>No attitude disturbances</td>
</tr>
<tr>
<td>Noise</td>
<td>Gaussian distribution with a standard deviation of 30 counts</td>
</tr>
<tr>
<td>Orbit Propagator</td>
<td>SGP4</td>
</tr>
</tbody>
</table>

3.2 Model Overview

The model is broken into a few steps. To start, the user must define several details about the simulation, the target, the observer, and the attitude profile. Then the pass prediction step propagates the target, observer, and Sun while checking the illumination requirements to find valid passes. Once the required number of passes are found, the tool takes the start and end time of each pass and begins the pass specific loop. Starting with the first pass, the tool again propagates the target, observer and Sun, but at the data rate defined by the user. Then the tool propagates the attitude of the target, either tumbling or controlled depending on the user inputs. Finally the Phong BRDF and Kaaselinin ray tracing algorithm are used to calculate
the brightness of the object at each time step. Then the tool moves on to the next pass and repeats the pass specific loop. This process is summarized in Figure 3.1 and each step will be explored in more detail next.

![Diagram](image)

**Figure 3.1:** Overview of the model steps.

### 3.3 User Definitions

To configure the simulation, the user defines several parameters in a .json file, which is basically a formatted text file. Required user definitions include simulation requirements, target, observer, and attitude profile. Each definition will be explored in detail in the following sections.
Some important .json syntax rules to remember are quotation marks around all strings, commas after every line except the last line, and curly brackets at the beginning and the end of the file. It is important to note that if any parameter is missing or misspelled, or if the .json format is not followed, then the simulation will give an error and not run. An example .json file for a ground-based observation of a tumbling cylinder in space is shown in Figure 3.2.
Figure 3.2: Example of a .json configuration file for a ground-based observation of a tumbling cylinder.
3.3.1 Simulation Requirements

The parameters required to define the simulation requirements are “Directory”, “Date”, “Number of Passes”, “Data Rate”, and pass requirements. An example of a space-based simulation requirements is shown in Figure 3.3.

![Example JSON configuration file](example_json_space.png)

**Figure 3.3:** Example of the simulation requirements for a space-based pass defined in a .json configuration file.

“Directory” is the folder where the data and plots for each pass will be saved.

“Date” is represented by a vector in YYYY-MM-DD-hh-mm-ss format and defines where to begin the pass prediction calculations. “Date” can be any valid day, but for ease the user can type “TLE” after “Date” to start the pass prediction at the epoch of the TLE.

“Number of Passes” sets the required amount of valid passes for the pass prediction code to find. Each pass will create a a folder within “Directory” named for the time at the beginning of the pass in YYYY-MM-DD-hh-mm-ss format.

The pass requirements are different for ground-based observations and space-based observations. For a ground-based observer, the only definition required is “Min Pass
Elevation [Deg]” and can be seen in Figure 3.2. For a space-based observer, “Min Pass Length [Sec]”, “Earth Exclusion Angle [Deg]”, and “Sun Exclusion Angle [Deg]” must be defined, as in Figure 3.3.

3.3.2 Target Definition

The target definition section is the same for both observers. This section consists of “Spacecraft Geometry”, “Specular Coeff”, “Diffuse Coeff”, “Inertia matrix [kg m^2]”, “TLE Line1”, and “TLE Line2” for the target. An example of a target definition is shown in Figure 3.4.

![Example target definition for a MEO box wing satellite.](image)

**Figure 3.4: Example of target definitions for a MEO box wing satellite.**

“Spacecraft Geometry” is chosen from a predefined list of geometries. The current list consists of BOX.WING, BOX, PLATE, CYLINDER, RECTANGLE, and LONG.RECTANGLE. For the purpose of this thesis the rectangle has dimensions of 1m, 2m, and 3m, while the long rectangle has dimensions of 1m, 2m, 5m. The box is a 1m cube. Drawings for each geometry are included in the appendix. It is possible to create new geometries or change the existing geometries by editing the class “Premade Spacecraft” in the “Reflection Funct” code file. The user would have to define each facet’s dimensions in the code.
The premade geometries are assumed to be made of homogeneous materials, but the user can define the specular and diffuse coefficients of that material in this section. The “Inertia matrix \([\text{kg m}^2]\)” of the target object is defined by a \(3 \times 3\) matrix.

The target TLE is broken into “TLE Line1” and “TLE Line2”, but is in standard TLE format that can be copy and pasted from a TLE recording website or STK.

### 3.3.3 Observer Definition

To define a ground-based observer enter “Ground” for “Observer Based”. Then define the observer with “Observation Site Latitude [Deg]”, “Observation Site Longitude [Deg]”, and “Observation Site Altitude [km]” represented in degrees north as positive, degrees east as positive, and kilometers above sea level respectively. An example is shown in Figure 3.5.

![Example user input for a ground-based observer.](image)

**Figure 3.5:** Example user input for a ground-based observer.

Similarly for a space-based observer enter “Space” for “Observer Based”. Then enter the TLE lines for “Obs TLE Line1” and “Obs TLE Line2”, shown in Figure 3.6.
For both observers parameters about the sensor must be defined. These parameters include “Sensor Diameter [m]”, “Exposure Time [s]”, and “CCD Gain”. The values in Figure 3.6 refer to the Basler avA2300-25gm telescope, which was also used by Rush [1].

3.3.4 Attitude Profile

Finally, the attitude profile for the target attitude dynamics has several different input options and formats. There are three main categories: random tumbling, defined tumbling, and controlled. A tumbling object could be used to model debris, while a controlled object can be used to model active satellites.

3.3.4.1 Random Tumbling

For random tumbling, input “Random” for “Attitude Profile”, like in Figure 3.7. The tool will produce a random starting attitude, defined in MRPs, and a random angular velocity, both of which are unit vectors. This ensures that the starting attitude is a valid MRP. Each pass will begin with this random state.
3.3.4.2 Defined Tumbling

Defined tumbling starts each pass at a defined attitude and angular velocity. “Attitude Profile” must be defined as “Euler”, “Quaternion”, or “MRP”. The input required for Euler angles is “ea_b.eci0 [rad]”, the input for quaternions is “q_b.eci0 [scalar last]”, and the input for MRPs is “MRP_b.eci0”. All tumbling defined inputs will be defined as the body relative to ECI frame. The angular velocity in the ECI frame, or “w_b.eci0 [rad/s]”, is defined in all three cases. An example of a quaternion definition is shown in Figure 3.8.

Figure 3.7: Example input for tumbling target attitude motion with random starting attitude and angular velocity.

Figure 3.8: Example input to define tumbling attitude motion for the target starting at a defined quaternion.
3.3.4.3 Controlled

The controlled attitude profile keeps the body frame aligned with a control quaternion relating the body frame to the LVLH frame. Usually, this control quaternion is used to keep the spacecraft pointed nadir, but any valid quaternion can be used. “qc.b_lvlh [scalar last]” is the control quaternion, “q_b_lvlh0 [scalar last]” is the initial attitude of the body frame relative to the LVLH frame, “w_b_eci0 [rad/s]” is the initial angular velocity, and “Td.b_eci [N/m]” is the disturbance torque in the ECI frame. User input to define a nadir pointing spacecraft is shown in Figure 3.9. It is important to note that every pass will start with these parameters. Therefore, if the pass starts with the spacecraft not pointing nadir, it will start each pass slewing to nadir. If the user wishes not to see these transients, they can define “q_b_lvlh0” and the control quaternion as the same thing.

Figure 3.9: Example user input for a controlled nadir pointing target.

3.3.5 User Definition Summary

To summarize the user definitions section, the following figures show the necessary user inputs for each case with the proper format. Figure 3.10 shows the simulation parameter user inputs. Figure 3.11 shows the target definition inputs. Figure 3.12
shows an example of a ground-based and space-based observer. Finally, Figure 3.13 shows the possible attitude configurations. These inputs are displayed with the proper syntax so that they can be copied and pasted into a .json file.

Figure 3.10: Example simulation parameters for a ground-based simulation on the left and a space-based simulation on the right.

Figure 3.11: Example target definition. "Spacecraft Geometry" can be any of the six options in the red outlined box.
Figure 3.12: Example user inputs for a ground-based observer on the left and a space-based observer on the right.

Figure 3.13: Example user inputs for a random tumbling object on the left, defined tumbling object in the middle, and a controlled object on the right. The tumbling defined object can be defined in either quaternions, Euler angles, or MRPs.
3.4 Pass Prediction

After unpacking all the user data, the tool begins to calculate valid passes. The process is slightly different for ground-based and space-based observers, but the idea is the same.

At one second increments, starting at the user defined date, the tool finds the location of the observer, target, and Sun in ECI coordinates. One second increments were chosen because it allowed the pass prediction code to pass verification, which will be explored in chapter 4, while only taking a few minutes to run each simulation. The location of the space-based observer and the target are calculated using the SGP4 propagator. The ground-based observer and the Sun are propagated using equation 5.56 and algorithm 8.1 from Curtis respectively [21]. The ground-based observer location is calculated by converting the longitude, latitude, and altitude to a position vector in ECI.

Once the location of the target, observer, and Sun are calculated, the three pass requirements are checked. For ground-based observers, the code checks if the target is illuminated, above the horizon, and if the observation site is in darkness. If all statements are true, then the object is illuminated and visible and the code records the time to indicate that a potential pass is occurring.

Equation (3.1) checks if the target is illuminated [21]. In this thesis a bold variable indicates a vector or a matrix while a non-bold variable is a scalar. Variables $\mathbf{r}_{\text{Sun}}$ and $\mathbf{r}_{\text{targ}}$ are the ECI vectors describing the location of the Sun and target respectively. The radius of the Earth is assumed to be 6378km.

$$\arccos\left(\frac{\mathbf{r}_{\text{Sun}} \cdot \mathbf{r}_{\text{targ}}}{\mathbf{r}_{\text{Sun}} \cdot \mathbf{r}_{\text{targ}}}\right) > (\arccos\left(\frac{6378}{\mathbf{r}_{\text{targ}}}\right) + \arccos\left(\frac{6378}{\mathbf{r}_{\text{Sun}}}\right))$$  \hspace{1cm} (3.1)
Equation (3.2) calculates if the target is above the horizon [21]. Here range is the vector from the observer to the target and r_site is the location of the ground site. Both vectors are in ECI coordinates.

\[ range \cdot r_{\text{site}} > 0 \]  

(3.2)

Equation (3.3) checks if the ground site is in darkness [21].

\[ r_{\text{Sun}} \cdot r_{\text{site}} < 0 \]  

(3.3)

For space-based observers the code checks if the target is illuminated, outside the Earth exclusion angle, and outside the Sun exclusion angle. Like ground-based observers, equation (3.1) is used to check if the target is in eclipse or not. The field of view, or FOV, that is obstructed by both the Earth and Sun must be calculated to check if the target is outside the Earth and Sun exclusion angles. This is important because for an observation satellite in LEO, the Earth can take up about 40° of the observer’s FOV. Therefore, to obtain an appropriate exclusion angle, the FOV of the celestial bodies must be added to the exclusion angles defined by the user.

Equation (3.4) and equation (3.5) calculate the FOV and exclusion angle for the Earth respectively. Here r_obs is the location of the observation satellite in ECI coordinates.

\[ Earth_{\text{FOV}} = \arctan \left( \frac{6378}{r_{\text{obs}}} \right) \]  

(3.4)
\[
\arccos\left(\frac{\text{range}}{\text{range}} \cdot \frac{\text{r}_\text{obs}}{\text{r}_\text{obs}}\right) > (\text{Earth exclusion req} + \text{Earth FOV}) \tag{3.5}
\]

Equation (3.6) and equation (3.7) calculate the FOV and exclusion angle for the Sun. The radius of the Sun is assumed to be 696347 km. Although \(\text{r}_\text{Sun}\) is the location of the Sun relative to the Earth, it is assumed that the difference between \(\text{r}_\text{Sun}\) and the location of the Sun relative to the observation satellite is trivial.

\[
\text{Sun FOV} = \arctan\left(\frac{696347}{\text{r}_\text{Sun}}\right) \tag{3.6}
\]

\[
\arccos\left(\frac{\text{range}}{\text{range}} \cdot \frac{\text{r}_\text{Sun}}{\text{r}_\text{Sun}}\right) > (\text{Sun exclusion req} + \text{Sun FOV}) \tag{3.7}
\]

When at least one of the requirements is no longer true, the code checks if the pass meets the minimum requirements defined by the user. If the pass meets the requirements, the start time, end time, and duration of the pass are saved. The process is repeated until the user defined amount of passes are found.

Next the data is saved as both .npy and .csv files and each pass is plotted. For ground-based observers a polar plot with azimuth and elevation is created, like in Figure 3.14.
Figure 3.14: Example pass prediction ground track.

For space-based, both the target and observer satellite are plotted in 3D, like in Figure 3.15. In this plot, the target is shown in red and the observer is in blue. The portion of the orbit where the observation happens is displayed by a thicker line and deeper color. The range vector is in green off of the observer, while the Sun pointing vector is yellow off of the target.
Figure 3.15: Example of a space pass prediction plot. The target is shown in red, the observer is in blue, observation vector in green, and the Sun pointing vector in yellow. The portion of the orbit where the observation happens is displayed by a thicker line and darker color for both the observer and the target.

3.5 Attitude Propagation

After the desired amount of passes are found, the code begins to calculate data for each pass at the user defined data rate. Starting with the first pass, the code takes the start and end time of the pass and propagates the position of the Sun, target, and the observer in the ECI frame. By splitting the pass prediction into two sections, it allows the tool to take larger steps to find the valid passes without sacrificing accuracy within each pass. The code will not spend as much time calculating undesired data in between each valid pass. Ground passes of LEO satellites, for example, have about 90 minutes of data in between each pass where the pass requirements are not met.
Once the locations are calculated at the data rate defined by the user, the attitude of the target is propagated. Depending on the user inputs, the target is propagated as either a tumbling or controlled objected.

### 3.5.1 Tumbling

For both defined tumbling and random tumbling the process is the same. The code takes the user input in Euler angles, quaternions, MRPs, or random MRPs and the tool converts the input into all three attitude formats. Then all three attitudes are propagated separately using the equations of motion below.

Equation (3.8) defines the change in angular velocity [23]. Here $\mathbf{\omega}$ is angular velocity, $\dot{\mathbf{\omega}}$ is the derivative of angular velocity, $\mathbf{I}$ is the inertia matrix of the target, $\mathbf{Tc}$ is control torque, and $\mathbf{Td}$ is disturbance torque. If the object is tumbling, both $\mathbf{Tc}$ and $\mathbf{Td}$ are zero.

\[
\dot{\mathbf{\omega}} = \mathbf{I}^{-1}(\mathbf{Tc} + \mathbf{Td} - \mathbf{\omega}^2 \mathbf{I}\mathbf{\omega})
\]  

Equation (3.9) and equation (3.10) are used to find the change in Euler angles or $\mathbf{\dot{e}a}$ [23]. The matrix $\mathbf{LBI}$ is a transformation matrix where $\phi$ refers to the angle around the positive $x$ axis and $\theta$ refers to the positive $y$ axis.

\[
\mathbf{LBI} = \frac{1}{\cos(\theta)} \begin{bmatrix}
\cos(\theta) & \sin(\phi) \times \sin(\theta) & \cos(\phi) \times \sin(\theta) \\
0 & \cos(\phi) \times \cos(\theta) & -\sin(\phi) \times \cos(\theta) \\
0 & \sin(\phi) & \cos(\phi) \times \cos(\phi)
\end{bmatrix}
\]  

\[
\mathbf{\dot{e}a} = \mathbf{LBI} \times \mathbf{\omega}
\]
Equation (3.11) calculates $\dot{q}$, which refers to the change in quaternions [23]. Where $\eta$ is the scalar of the quaternion, $\varepsilon$ is the vector part, and $I_3$ is a $3 \times 3$ identity matrix.

$$
\dot{q} = \begin{bmatrix}
\frac{1}{2}(\eta I_3 + \varepsilon \times)\omega \\
-\frac{1}{2}(\varepsilon \cdot \omega)
\end{bmatrix}
$$

Finally, equation (3.12) finds the change in MRPs, or $\dot{p}$, where $p$ is the MRP [22].

$$
\dot{p} = \frac{1}{4}((1 - p^T p)I_3 + 2p\dot{r} + 2pp^T)\omega
$$

Originally all propagation was done in MRPs, and the output was converted to quaternions and Euler angles. However, after analyzing initial results from verification, it was found to be more accurate to convert it once using the initials, rather than converting it at each time step. All attitude information is saved in .npy and .csv files to be used in the reflection calculations section.

### 3.5.2 Controlled

For controlled targets, the code imports the user defined initial quaternion, control quaternion, initial angular velocity, and disturbance torque. Then it calculates the Euler angles, quaternions, MRPs, and angular velocity in both the ECI frame and the LVLH frame. All of these values are then propagated in the ODE loop.

Controlled objects use the same equations of motion as tumbling objects. Equations (3.9), (3.10), (3.11), and (3.12) are used to find the change in attitude in both the ECI and the LVLH frame. Equation (3.8) is used to find the angular acceleration of the body frame relative to the ECI frame, but Equation (3.13) and Equation (3.14)
are used to calculate the angular velocity of the body frame relative to the LVLH frame.

Here $\omega_{lvlheci}$ is the angular velocity of the LVLH frame with respect to the ECI frame [23].

$$\omega_{lvlheci} = \frac{r \times v}{r^2}$$ \hspace{1cm} (3.13)

In Equation (3.14), $C_{beci}$ is the rotation matrix from the ECI frame to the body frame and $\omega_{blvlh}$ is the angular velocity of the body relative to the LVLH frame [23].

$$\omega_{blvlh} = \omega_{beci} - (C_{beci} \times \omega_{lvlheci})$$ \hspace{1cm} (3.14)

After all the attitude information is calculated, the difference between the current LVLH quaternion and the control quaternion is found. Equation (3.15) calculates this error represented by $q_{err}$ [23]. In this equation $\varepsilon_{b lvlh}$ and $\eta_{b lvlh}$ are from the current quaternion, while $\varepsilon_{cont}$ and $\eta_{cont}$ are from the control quaternion.

$$q_{err} = \begin{bmatrix}
\eta_{cont} \varepsilon_{b lvlh} + \eta_{b lvlh} \varepsilon_{cont} + (-\varepsilon_{cont} \times \varepsilon_{b lvlh}) \\
\eta_{cont} \eta_{b lvlh} - (-\varepsilon_{cont} \cdot \varepsilon_{b lvlh})
\end{bmatrix}$$ \hspace{1cm} (3.15)

Along with the angular velocity relative to the LVLH frame and the PD gains, the quaternion error is used in Equation (3.16) to find the control torque, or $T_c$ [23]. The vector part of the quaternion error is represented by $\varepsilon_{err}$, $kp$ is the proportional gain, and $kd$ is the derivative gain.
\[ Tc = -kp \ast \varepsilon_{err} - kd \ast \omega_{b.lvh} \] (3.16)

Because \( \omega_{b.lvh} \) and \( Tc \) are calculated at each time step rather than integrated, they are not included in the state vector. Because they are not part of the state vector, but still change every loop, another method must be used to record their values. One solution is to set up global variables inside the ODE loop and append the vectors each time they are calculated. In this case, \( Tc.s \) and \( \omega_{b.lvh}.s \) are the global variables for \( Tc \) and \( \omega_{b.lvh} \). However, the time steps inside the ODE loop are independent of the reported times outside of the loop. The ODE reports values of the state vector every increment of the data rate defined by the user, but inside the ODE loop the time step is as large or as small as it needs to be to meet the defined absolute and relative tolerance, each set to 1e-10. To remedy this, \( date.s \) is also defined as a global variable to record the time of each ODE loop. This way the time for each calculation of \( Tc \) and \( \omega_{b.lvh} \) is known. This means that some interpolation is needed during verification to align the times, which will introduce some error.

3.6 Reflection Calculations

Once the attitude propagation is complete, the code uses the attitude and location of the target, location of the observer, and location of the Sun to find the observed brightness of the target at each time increment.

The reflection calculation code was developed by Rush and is used in this thesis [1]. The algorithm and model are documented and verified in Rush’s thesis, but an brief explanation of the process follows [1].
To calculate the observed brightness of an object, Rush uses the Phong BRDF developed by Ashikhmin et al. and the ray tracing algorithm explained by Kaasalainen et al. [2][4]. The Phong BRDF calculates the brightness of a surface by accounting for the direction of the illumination vector, the observation vector, and the area that is illuminated. The ray tracing algorithm checks if each facet is illuminated and visible, if both are true the area of the facet is passed to the Phong BRDF and the brightness is calculated. Each facet is tested and the total brightness is summed.

3.6.1 Phong BRDF

Although the Phong BRDF can account for anisotropic reflection, Rush did not include this component because it would require extra information about the reflection properties of the target object. Instead Rush assumes “that specular reflection is evenly distributed along all directions” [1]. Generally, the researchers do not know the anisotropic properties of the target, so not including this information is a valid assumption.

The important vectors off of each facet are shown in Figure 3.16 [1]. Here \( \mathbf{u}_n \) is the unit vector normal to the surface, \( \mathbf{u}_x \) and \( \mathbf{u}_y \) are in the plane of the facet, \( \mathbf{u}_{Sun} \) points in the direction of the Sun, \( \mathbf{u}_{obs} \) points towards the observer, and \( \theta_h \) is the half angle between \( \mathbf{u}_{Sun} \) and \( \mathbf{u}_{obs} \).
These vectors are then used to calculate the total reflected intensity off of each facet. Equation (3.17) calculates $F_{obs}$ or the total visible power reflected by the spacecraft in Watts [5]. The Phong BRDF sums the reflected intensity from the diffuse and specular components to calculate $p_{total}$ [2]. $C_{Sun,vis}$ is the total flux of visible light from the Sun, assumed to be 445W/m², $d$ is the distance from the target to the observer in meters, $A_i$ is the illuminated area of the facet in m², and $N$ is the number of facets that make up the geometry [5].

$$F_{obs} = \frac{C_{Sun,vis}}{d^2} \sum_{i=0}^{N} A_i p_{total,i}(u_{ni} \cdot u_{Sun})(u_{ni} \cdot u_{obs})$$  (3.17)

The intensity in Watts is then converted to counts received by the CCD sensor. This is calculated using Equation (3.18), where $\alpha$ is the area of the sensor in m², $\Delta t$ is the exposure time in seconds, $E_{e^-}$ is the energy of a photon in the visible light spectrum in Joules, and $K$ is the CCD gain in $\frac{E_{e^-}}{count}$ [1].

$$counts = \frac{F_{obs} \alpha \Delta t}{E_{e^-} K}$$  (3.18)
3.6.2 Ray Tracing

The ray tracing algorithm by Kaasalainen et al. was utilized by Rush and will be used in this thesis as well. The algorithm breaks each facet into an array of sample points [4]. The user must first define a list of which facets could cast a shadow on other facets. If only rectangles are used this involves checking if any vertices from one facet is out-of-plane from another facet [1]. Figure 3.17 shows a case where facet A could cast a shadow on facet B [1]. Here, $V_p$ is the position of one vertex on facet A, and $B_c$ is the center of facet B. In this image it is clear that $V_p$ is out-of-plane of facet B, therefore, facet A it will be included in the list of possible facets that could shadow facet B.

![Diagram showing ray tracing](image)

Figure 3.17: Example of facet A shadowing facet B [1].

The algorithm checks if each facet is visible and illuminated by verifying the following conditions [1]:

1. $u_{obs} \cdot u_n > 0$
2. \( \mathbf{u}_{\text{Sun}} \cdot \mathbf{u}_n > 0 \)

3. There is no facet between the point and the Sun. (Shadow)

4. There is no facet between the point and the observer. (Hidden)

If all four conditions are true then the facet is added to the total reflected intensity. Figure 3.18 shows an example of a point that is visible to the observer but is shadowed by another facet and therefore not illuminated [1].

![Figure 3.18: Example of a point that is visible but not illuminated [1].](image)

### 3.7 Output

The main output from the tool is the pass prediction plot and the light curve plot, both of which are shown on the same figure. Figure 3.19 shows an example output of a ground-based simulation, while Figure 3.20 shows an example from a space-based simulation.
Figure 3.19: Example output for a ground pass.

Figure 3.20: Example output for a space pass.

The ECI attitude of the target is plotted as well. Figure 3.21 shows an example of the ECI attitude of a tumbling target.
Figure 3.21: Example output for the ECI attitude of a target.

If the object is controlled, the LVLH attitude and the control torque are plotted too. Figure 3.22 shows the LVLH attitude of a controlled target. Figure 3.23 shows the required control torque to keep the target pointed nadir.
Figure 3.22: Example output for the LVLH attitude of a target.

Figure 3.23: Example output for the control torque.
Chapter 4

VERIFICATION

To verify that the tool correctly models light curve observations, each section of the code will be tested against reliable sources. If all sections of the code pass verification, then the tool as a whole passes verification. To verify pass prediction for both space-based and ground-based observations, the code will be compared to results from STK. The attitude propagation section will be checked against matlab and Simulink code provided by, committee member, Dr. Mehiel. The Phong BRDF reflection model and the ray tracing algorithm were verified by Rush [1] in his thesis, but edits were made to include the specular and diffuse coefficients in the user inputs. Therefore, qualitative comparisons will be made to verify that these changes were successful.

4.1 Pass Prediction

To verify the pass prediction section of the code, several objects were modeled in both STK and the tool. It is important to note that the illumination and Sun requirements for the pass prediction portion were turned off. This helps focus on the line of sight requirements and the orbit propagator. To test the rest of the pass requirements, more STK reports were generated and compared. All data processing and comparison math were calculated in Matlab.
4.1.1 Ground-Based

4.1.1.1 Verification Plan

Ground pass prediction verification consists of three different targets viewed from Cal Poly, modeled at 37.30° north latitude, 122.66° west longitude, and 0km in altitude. Two LEO and one MEO objects were selected. The LEO objects were Cosmos 2414 and a Minotaur Rocket Body. The MEO object was a Breeze Rocket Body. The TLE for each object can be found in Appendix A. Figure 4.1 shows the STK ground track and Table 4.1 shows a summary of each object.

![STK ground track](image)

Figure 4.1: STK ground track of ground-based verification objects. With Cosmos 2414 in blue, Minotaur in red, Breeze in purple. In this instance MEO Breeze is visible from Cal Poly.

<table>
<thead>
<tr>
<th>Target</th>
<th>Orbit</th>
<th>Inclination</th>
<th>Eccentricity</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cosmos 2414</td>
<td>LEO</td>
<td>82.95°</td>
<td>0.0039</td>
<td>Active</td>
</tr>
<tr>
<td>Minotaur RB</td>
<td>LEO</td>
<td>98.75°</td>
<td>0.0012</td>
<td>Debris</td>
</tr>
<tr>
<td>Breeze RB</td>
<td>MEO</td>
<td>18.53°</td>
<td>0.5341</td>
<td>Debris</td>
</tr>
</tbody>
</table>

Table 4.1: Summary of test objects for ground-based pass prediction.
The goal was to test the code with a variety of orbits, inclinations and operational status. These specific objects were chosen semi-randomly from a space tracking website to get a range of these parameters. The observation window was 10 days in April 2021 and 10 passes for each satellite were modeled. Rise times, set times, and duration of passes were compared with STK. If these parameters are similar, it indicates that STK and the code are predicting similar locations of the target, locations of the ground site observer, and times that the target rises above and sets below the horizon. The test is successful if the average pass start time and end time offsets are below 1% of the total duration of the pass, these requirements are shown in Table 4.2. This ensures that the difference in orbit propagation and line of site requirements between STK and the tool are small. Also it will allow for some error for the fact that STK models the Earth as an oblate spheroid, where this thesis models a sphere. So the location of the horizon or the Earth’s FOV may be slightly different, causing some time offsets for the start and end of each pass. It is important to include offsets as a percentage because pass durations range from a few minutes to several hours. Comparing the offsets as a percent of the duration allows all test cases to be evaluated evenly.

<table>
<thead>
<tr>
<th>Table 4.2: Pass prediction verification requirements.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>Pass Start Time</td>
</tr>
<tr>
<td>Pass End Time</td>
</tr>
</tbody>
</table>

4.1.1.2 Results

In this thesis all time offsets will be calculated by subtracting the time predicted by STK from the time predicted by the code, or code – STK. This means that if a time is negative, then the code predicted the event to happen before the STK time. If a time is positive, then the code predicted the event to happen after the STK
time. These offsets are caused by differing locations of the target and the observer, so these offsets will cause errors in the light curve because the observation vector will be different. But if the differences in pass times are small, then the difference in light curve is small.

Both LEO satellites had pass start and end times within 5 seconds of STK times on average. The average pass durations was about 12\textit{mins} for LEO Minoutaur and 14\textit{mins} for LEO Cosmos. Figure 4.2 shows the offsets for each pass of the LEO Cosmos satellite, and Figure 4.3 shows the offsets for the LEO Minotaur rocket body.

![Figure 4.2: LEO Cosmos to Cal Poly pass time offsets.](image-url)
Pass 1 of the Minotaur satellite has a considerable deviation of 10s for the start time and 14s for the end time. However, this was the shortest pass at only 124s, with a max elevation of 0.3°, barely rising above the horizon. For comparison Pass 5 was the longest, with a duration of 942s and a max elevation of 65°. Because the minimum pass elevation was 0°, theoretically the observer could see the target, but in reality the landscape would often make this observation impossible. Every other pass had a start and end offset of 5s or less.

The MEO Breeze passes have larger start and end offsets, as shown in Figure 4.4, but the duration of the passes are significantly longer. The shortest pass, Pass 10, is about 1.5hrs long, while the longest pass, Pass 1, is about 11hrs long. The first six passes follow a clear trend of starting the pass about 15s earlier than STK and ending the pass about 10s later than STK. As the passes went on, the error grew slightly then the trend stopped, but the offsets were always less than 1min. The
largest offset was the end time for Pass 7 which was about 57s or 0.30% of the total pass duration. Even though the start and end offsets were larger than the offsets for the LEO satellites, the pass durations for MEO Breeze were drastically longer, resulting in a lower average percent offset.

![Graph showing pass time offsets for MEO Breeze](image)

**Figure 4.4: MEO Breeze to Cal Poly pass time offsets.**

Table 4.3 shows the average start and end offsets in percentage of the duration of the pass for each target. As mentioned earlier, the requirement is that the offsets must be less than 1%. Both the LEO Cosmos and MEO Breeze objects meet that requirement, but the LEO Minotaur object did not. However, if Pass 1 of the Minotaur object is ignored, the percent offsets drop to the value in parentheses, with is less than half of the original percentage. The average pass duration also increases by 1 min. This acceptable considering that very few ground sites are capable of taking observations at an elevation of 0.3° and that all other passes consistently had shorter offsets. With this result, the author is confident that the ground-based pass prediction code passes
verification for passes with maximum elevations higher than 10°. The author also recommends a “Min Pass Elevation [Deg]” of at least 10° to make sure that these low passes do not show up.

Table 4.3: Ground-based pass prediction average percent offsets, with a requirement of 1% or less.

<table>
<thead>
<tr>
<th></th>
<th>Start Offset %</th>
<th>End Offset %</th>
<th>Average Duration</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minotaur RB</td>
<td>0.98 (0.42)</td>
<td>1.25 (0.45)</td>
<td>12 min (13 min)</td>
<td>No (Yes)</td>
</tr>
<tr>
<td>Cosmos 2414</td>
<td>0.34</td>
<td>0.46</td>
<td>14 min</td>
<td>Yes</td>
</tr>
<tr>
<td>Breeze RB</td>
<td>0.12</td>
<td>0.10</td>
<td>8 hr</td>
<td>Yes</td>
</tr>
</tbody>
</table>

4.1.2 Space-Based

4.1.2.1 Verification Plan

Three objects were also selected for the space-based pass prediction verification. In this case a LEO, a MEO, and a GEO target. These targets were a Minotaur Rocket Body, a Breeze Rocket Body, and AMC-12 satellite respectively. Again these specific objects were chosen semi-randomly to test the code with a variety of orbits, inclinations and operational status. The observation satellite chosen was NEOSSat which is a LEO satellite in a Sun-synchronous orbit. NEOSSat was chosen because it is designed to take light curve observations and its orbit was selected accordingly. Figure 4.5 shows the orbit of each object. In this instance all three targets are in view of NEOSSat at the same time. Table 4.1 shows a summary of each target.
Table 4.4: Summary of test objects for space-based pass prediction.

<table>
<thead>
<tr>
<th>Target</th>
<th>Orbit</th>
<th>Inclination</th>
<th>Eccentricity</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minotaur RB</td>
<td>LEO</td>
<td>98.75°</td>
<td>0.0012</td>
<td>Debris</td>
</tr>
<tr>
<td>Breeze RB</td>
<td>MEO</td>
<td>18.53°</td>
<td>0.5341</td>
<td>Debris</td>
</tr>
<tr>
<td>AMC 12</td>
<td>GEO</td>
<td>1.75°</td>
<td>0.0003</td>
<td>Active</td>
</tr>
</tbody>
</table>

The observation window was again 10 days in April 2021 and 10 passes for each target was modeled. The success criteria will be the same as the ground-based test, start time and end time offsets less than 1% of the duration of the pass on average. These requirements are reiterated in Table 4.2.

4.1.2.2 Results

Figure 4.6 shows the offsets for the LEO Cosmos object observed by NEOSsat. The first pass was about 21 min long with offsets of about 5 s on the start and 1 s at the end. The tool oscillated between predicting longer passes and shorter passes, but was always less than 10 s off, excluding Pass 10. In this case the tool added about 30 s on either side of the pass. STK predicted Pass 10 to only be about 3 min long, about
7 times shorter than Pass 1. Similar to the low elevation angle for the LEO Minotaur ground-based observation, the largest Earth exclusion angle recorded during Pass 10 of the space-based Cosmos observation was only 0.58° above the required angle. For comparison, Pass 1 had a max Earth exclusion angle 19° above the required angle. This problem pass points to the importance of having a larger ”Min Pass Duration [s]”, which could filter out short passes like Pass 10.

![Graph showing pass time offsets](image)

**Figure 4.6: Leo Cosmos to Leo NEOSSat pass time offsets.**

The MEO Breeze pass offsets were more varied, as seen in Figure 4.7. The author suspects this variance is caused by the highly elliptical orbit of the Breeze Rocket Body. All 10 observations were taken in about 18hrs. In this time the Breeze Rocket Body completed less than 2 full orbits. The first pass lasted about 3.5 hrs, while the shortest pass, Pass 4, only lasted 34mins. This means that each observation likely sees the Breeze Rocket Body at a different portion of its orbit and not at regular intervals, which causes the variance. This differs from observations of the
LEO Cosmos satellite, which had a similar orbit as the observing NEOSSat. In this case a pass was observed about every 50 mins. However, for the MEO Breeze Rocket Body, there was no clear pattern in pass times. Nevertheless, the largest offset was the start time for Pass 8 which was 26s or 0.27%, which is below the required 1%.

![Diagram](image.png)

**Figure 4.7:** MEO Breeze to LEO NEOSSat pass time offsets.

The GEO AMC pass offsets show an asymptotic pattern in Figure 4.8. All 10 passes happen within 17 hrs. Passes 1 through 5 both start and end before the STK predicted time, while passes 7 through 10 are predicted to happen before the STK time. Pass 6 differs as it predicts the pass to start 21s before STK and to end 20s after STK, or about 0.23% of the duration of the pass. Pass 6 is unique because it is 2.5 hrs long, while the other 9 Passes are about 1 hr long. That means for Pass 6, NEOSSat completes about 1.5 orbits before AMC-12 goes out of sight. That means during this time AMC-12 is passing either in front or behind the Earth, with respect to the Sun,
and coming out the other side. This could explain why the offsets go from before STK to after STK after Pass 6. Overall the offsets never get larger than 0.40%.

Figure 4.8: GEO AMC to LEO NEOSSat pass time offsets.

Table 4.5 shows the average percentage offset for each object. Here, the MEO Breeze and GEO AMC targets easily meet the requirement of less than 1% of the pass duration, while the LEO Cosmos object does not. However, like the LEO Minotaur object for the ground-based pass prediction, if Pass 10 is neglected, the percent offsets decrease to about a third of the original value, as shown in the parentheses. This is acceptable considering how short Pass 10 was and that every other pass was consistently less than 5s off from STK. The author is confident that the space-based pass prediction code is verified for passes longer than 5mins. The author also recommends a minimum pass duration of at least 5mins or 300s, to make sure that these short and low Earth exclusion angle passes are ignored.
Table 4.5: Space-based pass prediction average percent offsets, with a requirement of 1% or less.

<table>
<thead>
<tr>
<th></th>
<th>Start Offset %</th>
<th>End Offset %</th>
<th>Average Duration</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEO Cosmos</td>
<td>2.05 (0.69)</td>
<td>1.84 (0.51)</td>
<td>14 min (15 min)</td>
<td>No (Yes)</td>
</tr>
<tr>
<td>MEO Breeze</td>
<td>0.15</td>
<td>0.19</td>
<td>82 min</td>
<td>Yes</td>
</tr>
<tr>
<td>GEO AMC 12</td>
<td>0.10</td>
<td>0.14</td>
<td>69 min</td>
<td>Yes</td>
</tr>
</tbody>
</table>

4.1.3 Illumination Requirements

4.1.3.1 Verification Plan

The pass prediction verification accounts for the requirement that the target must be above the horizon, for ground observers, or outside the Earth exclusion angle, for space observers. To verify the rest of the pass requirements, STK reports for eclipses, site darkness times, and Sun location were compared to data generated by the tool.

The eclipse reports verify that STK and the tool predict the target to be illuminated at the same times. Site darkness reports verify that both predict the same night time for the ground site. The Sun location will be used to compare Sun pointing vectors to ensure that proper Sun exclusion angles and illumination vectors are calculated.

The start and end time offsets for eclipses and site darkness cycles are required to be less than 1% of the total duration of the event on average. The Sun pointing vector will be required to have an offset of less than 0.5°, which is explored in more detail next.

The Sun location affects the site darkness times, the Sun exclusion angle, and the illumination vector. The site darkness times will be compared to STK directly and the Sun exclusion angle only determines if the pass is valid or not. The more concerning effect of an error in Sun location is the change to the illumination vector and light
curve calculations. To ensure that an offset of 0.5° for the Sun pointing vector is acceptable, an error of 0.5° was added to the code. The Sun pointing vector was rotated 0.5° in the X, Y, and Z axes to see how the light curve was effected. Figure 4.9 shows the control and rotated light curve for each case plotted on top of each other.

![Graphs of light curves rotated around X, Y, and Z axes](image)

**Figure 4.9:** Light curves resulted from rotating the Sun pointing vector 0.5° in all three axes.

Table 4.6 shows the average percent difference for each case. The unit vector of the Sun at the beginning of the pass is [0.9325; 0.3313; 0.1436]. The percent difference in light curves for the rotation around the X-axis was the least, which makes sense because the majority of the unit vector is in the X-axis. Therefore, rotating about the X-axis does not change the direction of the vector that much. Furthermore, the rotation about the Z-axis, the smallest component of the Sun pointing unit vector, causes the largest difference at 1.13%. The Earth orbits the Sun at about 1° per day, therefore, a rotation of 0.5° around the Z-axis is about equal to the difference in Sun
pointing vectors in half a day. It is unlikely for the offset to be entirely in the Z-axis, so with this requirement a maximum difference of about 1% in light curves can be expected.

<table>
<thead>
<tr>
<th>Table 4.6: LEO Cosmos eclipse average percent offset.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation X Axis</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>Percent Difference</td>
</tr>
</tbody>
</table>

The success criteria for the illumination requirements are summarized in Table 4.7.

<table>
<thead>
<tr>
<th>Table 4.7: Illumination requirements success criteria.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>Eclipse Times</td>
</tr>
<tr>
<td>Site Darkness Times</td>
</tr>
<tr>
<td>Sun Pointing Vector</td>
</tr>
</tbody>
</table>

### 4.1.3.2 Eclipse Results

During the observation window, the only satellite to experience eclipses was the Cosmos 2414 satellite. The satellite experience 21 eclipses, all of which lasted about 34 mins, except the first pass where the test window started during the eclipse. Both STK and the code reported when the object entered penumbra. Figure 4.10 shows the offsets for each eclipse. Over all 21 eclipses the two never disagreed more than 2s. Table 4.8 shows that the offsets were on the order of a hundredth of a percent on average. This is well below the 1% requirement and an encouraging result. Furthermore, the tool did not report false eclipses for the other targets. However, this is small sample size and it would be beneficial to test more objects that experience eclipses.
Figure 4.10: Eclipse start and end offsets for LEO Cosmos.

Table 4.8: LEO Cosmos eclipse average percent offsets, with a requirement of 1% or less.

<table>
<thead>
<tr>
<th></th>
<th>Start Offset %</th>
<th>End Offset %</th>
<th>Average Duration</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEO Cosmos</td>
<td>0.03</td>
<td>0.05</td>
<td>33.5 min</td>
<td>Yes</td>
</tr>
</tbody>
</table>

4.1.3.3 Site Darkness Results

During the MEO Breeze ground-based pass test, the ground site experienced 8 night cycles. Figure 4.11 shows the offsets for each night, again reporting differences in penumbra times. The code consistently predicts the dark time to start about 70s after STK and to end about 70s before STK, with an average duration of about 11hrs. This means that the tool requires the site to be darker before deciding that it is dark enough to make an observation. The average offset is about 0.18%, so this verifies that the tool reliably predicts ground-site darkness times. Table 4.9 shows that the start and end offsets are well below the 1% limit.
**Figure 4.11:** Ground site darkness time offsets.

**Table 4.9:** Ground site darkness average percent offsets, with a requirement of 1% or less.

<table>
<thead>
<tr>
<th></th>
<th>Start Offset %</th>
<th>End Offset %</th>
<th>Average Duration</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEO Breeze</td>
<td>0.17</td>
<td>0.18</td>
<td>11 hrs</td>
<td>Yes</td>
</tr>
</tbody>
</table>

4.1.3.4 **Sun Pointing Vector Results**

In this section the offsets between the Sun pointing unit vectors calculated by STK and the tool will be reported in degrees. Because all simulations had the same start time and used the same method to propagate the Sun, the offsets were very similar for all targets. The ground-based MEO Breeze observations took the longest, about 8 days, so this data was plotted as shown in Figure 4.12. There is little variance over the 8 days, with the offset at about 0.29° the entire time. This is true for all satellites, Table 4.10 shows the average and maximum offset, which were all about 0.29°. This is below the required value of 0.5°. It is also important to note that the offset was
spread between all axes with the average difference between the two units vectors at [0.0019; 0.0044; 0.0019]. Although this does not translate exactly to rotations, it means that the offset is not entirely in the Z-axis and that the percent difference in light curves is below 1%.

![Graph showing sun pointing offsets over time](image)

**Figure 4.12: Sun pointing offsets in degrees.**

<table>
<thead>
<tr>
<th></th>
<th>Cosmos</th>
<th>Minotaur</th>
<th>Breeze</th>
<th>AMC 12</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Offset</td>
<td>0.2967°</td>
<td>0.2967°</td>
<td>0.2967°</td>
<td>0.2967°</td>
<td>Yes</td>
</tr>
<tr>
<td>Avg Offset</td>
<td>0.2966°</td>
<td>0.2964°</td>
<td>0.2966°</td>
<td>0.2966°</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Table 4.10: Average Sun pointing offset with a requirement of 0.5° or less.**

### 4.1.4 Pass Prediction Verification Summary

It has been shown that results from the ground-bases pass prediction, space-bases pass prediction, and illumination requirements agree closely with STK reports. This
verifies that the code accurately predicts passes, eclipses, site darkness cycles, and the location of the Sun.

4.2 Attitude Propagation

The attitude propagation section was verified with simulink and matlab code provided by committee member Dr. Mehiel. This code will be referenced as MVC for Dr. Mehiel’s Verification Code. The author slightly edited MVC to be able to turn on and off the control torque, accept initial conditions in the ECI and LVLH frames, and to output MRPs by converting the quaternion at each time step. All edits were approved by Dr. Mehiel.

It is important to note that MVC only accepts quaternions and angular velocity as the initial input and converts from there. Once converted the Euler angles and quaternions are propagated using the equation of motion from the Methodology section. At the end, the quaternions are converted to MRPs. This is a different method from the tool, where Euler angles, quaternions, or MRPs can be inputted and after converting the initial state, all three are propagated at the same time. This may lead to small discrepancies between the two, but the author aimed to make as few edits to MVC as possible.

4.2.1 Tumbling

4.2.1.1 Verification Plan

Although the equations of motion for tumbling objects are always the same, the code accepts initial conditions in three different formats, Euler angles, quaternions, or MRPs. To verify that there were no errors in converting initial conditions, all three
input formats were tested. Using matlab functions, one attitude was converted into all three formats and rounded to four decimals. Three cases were developed using this method. Case 1 starts with Euler angles and converts them to the other two formats. Then all three formats are ran as separate simulations. Data from these simulations are then compared to MVC. Case 2 does the same starting with quaternions, and Case 3 starts with MRPs. Table 4.11 shows the values tested for each case.

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>( e_{a.b.eci} )</td>
<td>([0.1;0.1;0.1])</td>
<td>([1.5708;0;1.5708])</td>
<td>([3.1416;0;0])</td>
</tr>
<tr>
<td>( q_{b.eci} )</td>
<td>([0.0474;0.0523;0.0474;0.9964])</td>
<td>([0.5;0.5;0.5;0.5])</td>
<td>([1;0;0])</td>
</tr>
<tr>
<td>( MRP_{b.eci} )</td>
<td>([0.0237;0.0262;0.0237])</td>
<td>([0.3333;0.3333;0.3333])</td>
<td>([1;0;0])</td>
</tr>
</tbody>
</table>

The parameters compared were Euler angles, quaternions, MRPs, and angular velocity, all in the ECI frame. Angular velocity offsets were calculated by subtracting the angular velocity calculated by the tool and MVC and finding the magnitude of the difference. Neither simulation included perturbations, so the angular velocity should remain constant. To compare Euler angles, quaternions, and MRPs on the same level, all attitudes were first converted into their respective direction cosine matrix (DCM). Then the conjugate transpose of the DCM from the tool and the DCM from MVC was calculated. The identity matrix, which represents perfectly aligned coordinate frames, was then subtracted from this matrix to get the difference. This process is shown in Equation (4.1). Here \( C_t \) is the DCM calculated from the tool, \( C_m \) is the DCM calculated by MVC, and \( err \) is the offset between the two.

\[
err = C_tC_m' - I_3
\]  

(4.1)

This process will be repeated for relating DCMs at every time step, thesis Euler DCM to MVC Euler DCM and so on. Finally the norm of each resultant matrix was
calculated and compared. This value is a unitless scalar that represents the pointing offset between the attitudes calculated by the tool and the attitudes calculated by MVC. For reference a pointing difference of 180° relates to an offset of 2 and an pointing difference of 1° relates to an offset of 0.0175. Comparing DCMs allows all three attitude coordinate frames to be analyzed at the same time.

The tool will be validated if the maximum offset is less than 1e-2 and the average offset is less than 1e-3 for a typical ground to LEO observation of about 14 mins. This relates to a max offset of about 0.6° and average offset of about 0.06°. Although the offsets will not be zero because of small differences in propagation methods and initial conditions, these required values are sufficiently small to ensure that the differences between the tool and MVC are minimal. Also these values were chosen because it was shown in the Sun pointing vector verification that angle offsets of less than 0.5° results in small light curve offsets. Therefore, with an average attitude offset of 0.06° or less, there will be minimal light curve differences. The success criteria is summarized in Table 4.12.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Offsets</td>
<td>Less than 1e-3</td>
</tr>
<tr>
<td>Maximum Offsets</td>
<td>Less than 1e-2</td>
</tr>
</tbody>
</table>

### 4.2.1.2 Results

All three tumbling cases show a similar trend. Each input starts at near zero offsets and linearly increases as time goes on. The largest error was the Euler angles in Case 2, shown in Figure 4.13. But even here, the linear increase is slow, only increasing about 1e-3 in 14 mins. This linear error is a result of differences in propagators and small offsets in initial conditions.
Figure 4.13: Tumbling attitude offsets for Case 2.

The MRPs for Cases 1 periodically spike for a short time, but again has a linear increase in amplitude of the spikes. This is displayed in Figure 4.14.
Figure 4.14: Tumbling attitude offsets for Case 1.

Table 4.13 shows the average offset for each case was below 1e-3 for all attitudes in all cases. The angular velocity offset was exactly 0 because there was no disturbance torque so it remained constant through each case.

Table 4.13: Average offsets for a target with tumbling attitude, with a requirement of 1e-3 or less.

<table>
<thead>
<tr>
<th></th>
<th>Euler Angle Input</th>
<th>Quaternion Input</th>
<th>MRP Input</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler Angles</td>
<td>4.02e-4</td>
<td>4.00e-4</td>
<td>4.40e-4</td>
<td>Yes</td>
</tr>
<tr>
<td>Quaternion</td>
<td>1.12e-4</td>
<td>7.76e-5</td>
<td>1.87e-4</td>
<td>Yes</td>
</tr>
<tr>
<td>MRP</td>
<td>8.55e-5</td>
<td>7.75e-5</td>
<td>1.51e-4</td>
<td>Yes</td>
</tr>
<tr>
<td>Angular Velocity</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The max offset for each case is shown in Table 4.14. All attitude formats for all cases had a maximum offset of less than 1e-2.
Table 4.14: Maximum offsets for a target with tumbling attitude, with a requirement of 1e-2 or less.

<table>
<thead>
<tr>
<th></th>
<th>Euler Angle Input</th>
<th>Quaternion Input</th>
<th>MRP Input</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler Angles</td>
<td>1.15e-3</td>
<td>1.15e-3</td>
<td>1.29e-3</td>
<td>Yes</td>
</tr>
<tr>
<td>Quaternion</td>
<td>2.55e-4</td>
<td>1.66e-4</td>
<td>3.39e-4</td>
<td>Yes</td>
</tr>
<tr>
<td>MRP</td>
<td>1.84e-3</td>
<td>1.87e-3</td>
<td>1.99e-3</td>
<td>Yes</td>
</tr>
<tr>
<td>Angular Velocity</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The average and maximum offset requirements were met and show that the tumbling attitude propagation code is verified.

4.2.2 Controlled

4.2.2.1 Verification Plan

The controlled attitude propagation also had three test cases, but only one input format. The three test cases were: target starting at nadir with an initial angular velocity, target not at nadir with no angular velocity, and target not pointing at nadir with angular velocity. The test cases and their values are shown in Table 4.15.

Table 4.15: Summary of test cases for controlled attitude motion verification.

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_{b.lvlh}$</td>
<td>[0;0;0;1]</td>
<td>[0.5;0.5;0.5;0.5]</td>
<td>[0.5;0.5;0.5;0.5]</td>
</tr>
<tr>
<td>$qc_{b.lvlh}$</td>
<td>[0;0;0;1]</td>
<td>[0;0;0;1]</td>
<td>[0;0;0;1]</td>
</tr>
<tr>
<td>$w_{b.eci}$</td>
<td>[0.1;0.2;0.3]</td>
<td>[0;0;0]</td>
<td>[0.1;0.2;0.3]</td>
</tr>
</tbody>
</table>

Euler angles, quaternions, MRPs, and angular velocity in both the ECI and the LVLH coordinate frames, as well as the control torque in the ECI frame, will be compared. The data will be analyzed in a similar manner to the tumbling verification, with the normalized conjugate transpose of the DCM matrices being compared. The control
torque will be compared similarly to the angular velocity, by finding the norm of the difference between the two vectors. A ground observation of the LEO Cosmos satellite was modeled for this test as well. Again a successful test is defined as a maximum offset less than 1e-2 and an average offset less than 1e-3 for each case, these requirements are shown in Table 4.12.

4.2.2.2 Results

All three cases for the controlled attitude profile showed similar trends. Figure 4.15 shows the offsets for Case 1. The ECI quaternions and MRPs have exactly the same offset, while the Euler angles follow the same trend. The ECI attitudes show an increasing offset ending below 2e-4. The LVLH attitudes have some transients before settling at a constant offset below 4e-7. The angular velocity in the ECI frame also shows these transients at the same magnitude as the LVLH attitudes. The IVLH angular velocity shows slightly larger discrepancies at the beginning of the pass, but then settles at offsets on the order of 1e-9. Similar trends are shown in Figure 4.16 for Case 2 and Figure 4.17 shows the offsets for Case 3.
Figure 4.15: Controlled attitude offsets for Case 1.
Figure 4.16: Controlled attitude offsets for Case 2.
Table 4.16 shows the average offsets for all passes. All averages are well below the 1e-3 requirement.

Table 4.16: Average controlled attitude offsets for all cases, with a requirement of 1e-3 or less.

<table>
<thead>
<tr>
<th></th>
<th>ECI Offset</th>
<th>LVLH Offset</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler Angles</td>
<td>6.62e-5</td>
<td>4.57e-7</td>
<td>Yes</td>
</tr>
<tr>
<td>Quaternion</td>
<td>6.20e-5</td>
<td>4.12e-7</td>
<td>Yes</td>
</tr>
<tr>
<td>MRP</td>
<td>6.18e-5</td>
<td>1.85e-8</td>
<td>Yes</td>
</tr>
<tr>
<td>Angular Velocity</td>
<td>2.21e-7</td>
<td>2.95e-7</td>
<td>Yes</td>
</tr>
<tr>
<td>Control Torque</td>
<td>6.75e-8</td>
<td>6.75e-8</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 4.17 shows the maximum offsets for all passes. Again all maximums are well below the 1e-2 requirement.
Table 4.17: Maximum controlled attitude offsets for all cases, with a requirement of 1e-2 or less.

<table>
<thead>
<tr>
<th></th>
<th>ECI Offset</th>
<th>LVLH Offset</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler Angles</td>
<td>1.90e-4</td>
<td>1.19e-6</td>
<td>Yes</td>
</tr>
<tr>
<td>Quaternion</td>
<td>1.78e-4</td>
<td>9.39e-7</td>
<td>Yes</td>
</tr>
<tr>
<td>MRP</td>
<td>1.78e-4</td>
<td>9.20e-7</td>
<td>Yes</td>
</tr>
<tr>
<td>Angular Velocity</td>
<td>3.83e-7</td>
<td>5.92e-5</td>
<td>Yes</td>
</tr>
<tr>
<td>Control Torque</td>
<td>1.33e-5</td>
<td>1.33e-5</td>
<td>Yes</td>
</tr>
</tbody>
</table>

For all cases the average and maximum offset requirements are sufficed, thus the controlled attitude code is verified.

4.2.3 Attitude Summary

These results verify that the tool can propagate both controlled and tumbling targets. Small linearly increasing offsets are the result of small offsets in initial conditions and differences in propagators, not issues with the model. Results shown in this section were from the ground-based simulations. Although the attitude code is the same for space-based simulations, out of an abundance of caution, similar cases were tested for both controlled and tumbling objects in space-based simulations. Appendix C shows these results and proves that they also meet the verification requirements.

4.3 Reflection Coefficients

4.3.1 Verification Plan

To ensure that the edits to the specular and diffuse coefficients were effective, three cases were tested for both coefficients. Each coefficient was varied separately while the other was kept at zero, isolating the effects of the tested variable. Table 4.18 shows
the tested values for each coefficient. These values were chosen to see the difference in light curve caused by a range of reflection coefficients. All other variables were held constant.

**Table 4.18: Summary of test cases for reflection coefficients.**

<table>
<thead>
<tr>
<th>Case</th>
<th>Tested Coefficient</th>
<th>Other Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 1</td>
<td>0.1</td>
<td>0</td>
</tr>
<tr>
<td>Case 2</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>Case 3</td>
<td>0.9</td>
<td>0</td>
</tr>
</tbody>
</table>

A successful test will show the same light curve trend with three different amplitudes. For both coefficients, Case 1 should have the lowest amplitude, while Case 3 should have the highest amplitude of reflected light. To compare the trends of all three cases, each case will be normalized by dividing all data points in that case by the maximum brightness in that case. This will make each measurement relative to the maximum brightness. A successful test will show all three cases having the same normalized brightness at every time step.

### 4.3.2 Specular Results

As expected all three specular coefficient cases show the same trend of light curve but at different amplitudes. Figure 4.18 shows the absolute brightness of each case on the top and the normalized brightness of each on the bottom. The absolute brightness shows that as the specular coefficient increases, the brightness increases, but when each run is normalized all three cases are right on top of each other.
Figure 4.18: Specular Coefficient comparison with total counts on the top and normalized counts on the bottom.

4.3.3 Diffuse Results

The same is shown in Figure 4.19 for the diffuse test. Again the amplitude of each case is different, but the trend is the same.
Figure 4.19: Diffuse Coefficient comparison with total counts on the top and normalized counts on the bottom.

4.3.4 Reflection Coefficient Summary

These results show that the edits made to the reflection calculation code were successful and the specular and diffuse coefficients are validated.
Chapter 5

RESULTS AND ANALYSIS

In this chapter, the capability of the tool will be explored and qualitative differences between input configurations will be explained.

5.1 Geometry

As mentioned earlier, the tool can model six different premade geometries: a box, box wing, rectangle, long rectangle, plate, and cylinder. To test how different geometries affect the observed light curve, each geometry was modeled using the LEO Minotaur TLE with Cal Poly as the observer. The simulations had the same tumbling attitude, reflection coefficients, and observation window. Figure 5.1 shows the resultant light curve from each geometry for the same pass.
Figure 5.1: Light curve result for all six different premade geometries.

Box, box wing, rectangle, long rectangle, and cylinder all follow similar overall trends that start at a brighter magnitude and end the pass near zero brightness. This is expected as each object has the same ephemeris. Box and box wing have counts with a magnitude of $10^5$ while long rectangle and cylinder have counts about an order of magnitude larger. Rectangle is about in the middle of the four. This makes sense because the box is only $1m^3$ so each face has an area of only $1m^2$ for the box and $2m^2$ for the box wing satellite, if the solar panels are included. On the other hand, the cylinder is $9m$ long and the largest face of the rectangle and long rectangle are $6m^2$ and $10m^2$ respectively. This also explains why the long rectangle is about twice as bright as the rectangle, because it has about twice as much surface area.

The $1m$ square plate light curve is different from the other five geometries because it has zero thickness. This means that it only has two faces that reflect light while
the rest have at least six. Having few reflective surfaces makes the brightness very directional. For the first two minutes of the pass, the plate recorded no reflected light because its faces were not angled properly to reflect sunlight towards Cal Poly. About half way through the pass the plate tumbles into an orientation that periodically reflects light to the observer at a magnitude of 1e4 counts and decreases as the pass continues. This relatively simple light curve, with clearly defined spikes at regular intervals, makes the plate a good geometry for light curve analysis methods to be tested against. This is why authors like Linares et al. and Wetterer et al. chose to use this geometry in their research [5][6][7].

While the plate is a good starting point for analysis methods to be tested against, each premade geometry can be useful to light curve researchers. The box, rectangle, and long rectangle are still convex shapes with no self shadowing appendages, but they have more reflective surfaces which causes more variance in the light curve. This variance in the light curve more accurately represents the data from real observations. The cylinder can be used to simulate a rocket body tumbling in orbit, which is a fairly common orbital debris. The cylinder could be useful for comparing results with real observations of rocket bodies, which could help bridge the gap between simulated data and real observations. The box wing is the only self shadowing geometry and can be used to model an active or tumbling satellite. Because of the self-shadowing solar panels, the box wing is the most challenging premade geometry for a light curve analysis method.

5.2 Attitude Profiles

The attitude profile can effect the light curve measurements as well. For this section, the LEO Cosmos to NEOSSat .json file was used. Two objects were modeled as
cylinders, one was controlled to stay at nadir while the other had an angular velocity of \([0.1; -0.2; 0.3]\). Figure 5.2 compares the light curves for the tumbling and controlled object.

![Figure 5.2: Tumbling vs controlled light curve.](image)

Again, the two objects have similar magnitudes for counts because they have the same ephemeris. But the tumbling cylinder has more variance because as it rotates multiple faces with differing areas reflect sunlight at different angles.

The variance of the light curve makes it easier for the Kalman filter to settle on a solution, which is why most researchers focus on tumbling debris. However, the option to model controlled objects will be useful to researchers looking to refine their analysis method. A light curve analysis method that can predict the attitude motion of an actively controlled satellite could be verified with real observation data. If the researchers have access to data from attitude sensors onboard the active satellite, they can take light curve observations of the target and use the attitude sensor data to verify their analysis model. Unlike observations of tumbling debris where the
researchers do not know the state of the target, the researchers would know the true attitude of the target and could verify their results. This could also help bridge the gap between simulated and observed data.

5.3 Ground-Based vs Space-Based

One of the concepts explored in this thesis is the difference between ground-based observations and space-based observations. As mentioned earlier, one of the benefits of a space-based observer is the variance in light curve data and synodic periods when observing a slowly rotating GEO object [16]. Ground and space-based observations of SXM-8, an active GEO communications satellite, were simulated to show this difference. AMC-12, which was used in the space-based pass verification section, could not be used in this case, because AMC-12 is not stationed above Cal Poly. SXM-8 was modeled as a nadir pointing long rectangle. Although SXM-8 has solar panels the long rectangle shape represents the size of the large GEO communications bus better than a 1m box wing. Figure 5.3 shows the observed light curve from Cal Poly, while Figure 5.4 shows the light curve recorded by NEOSSat. Both passes were recorded on the night of March 20, 2021.
Figure 5.3: Light curve observation of nadir pointing SXM-8 observed from Cal Poly.

Figure 5.4: Light curve observation of nadir pointing SXM-8 observed from NEOSSat.
Both observations start at exactly the same time, when SXM-8 exits eclipse. SXM-8 then stays in view of NEOSSat for about $1.75\text{hrs}$. Cal Poly sees the satellite decrease in brightness for a total of $5.25\text{hrs}$ or until sunrise at about 12:30 UTC or 5:30AM PST. Figure 5.5 shows the two light curves on the time scale of the space-based observation for easier comparison.

![Figure 5.5: Ground and Space-based observation of SXM-8 on same time scale.](image)

The magnitude of the two light curves cannot be compared directly because the observers are at different locations, but the trend of each can be compared to point out the differences between ground-based and space-based observations. While the difference in trend is not as large as the tumbling vs controlled results, there is a noticeable discrepancy between the two light curves. As NEOSSat orbits the Earth, it is able to see SXM-8 from more angles. This causes an increase in brightness about halfway through the pass, then decrease as NEOSSat returns to a similar position in
its orbit as at the beginning of the observation. This is an example of the variance in
light curve data caused by the quickly moving space-based observer. This variance is
beneficial to light curve analysis methods.

The ability to see objects from more angles, not be limited by the time of day or
weather, and to see nearly all of the objects in GEO and nearby graveyard orbits are
what makes space-based observations viable. This simulation tool can be used by
light curve researchers to develop space-based light curve analysis methods that may
one day utilize real observed space-based data.

5.4 User Interface

While the user interface with the .json file is explored in detail in the Methodology
section, the user experience with the python code will be shown in this section. All
that is needed to run the simulation is the path to where the .json file is saved and the
function call out run_sim(path). The simulation can be called from a console, but
the author made a main script to save many paths to quickly and easily run different
configurations. Figure 5.6 shows an example of the path definition and simulation
call out.

```
# Space-Based SXMB
path = '/Users/Andyt/OneDrive/Desktop/Ochoa_Thesis_Code/RESULTS_GEO_SPACE/results_geo_space.json'
run_sim(path)

# Ground-Based SXMB
path = '/Users/Andyt/OneDrive/Desktop/Ochoa_Thesis_Code/RESULTS_GEO_GROUND/results_geo_ground.json'
run_sim(path)
```

Figure 5.6: Example of simulation function call out.
As the simulation runs, the console updates the user about what calculations are being done. The code first repeats the path to the user so they can check if they are using the correct .json file. Then it reports each valid pass as it propagates the target and observer. Figure 5.7 shows an example of what the code displays for a configuration that requires three passes.

![Example output for pass prediction.](image)

**Figure 5.7: Example output for pass prediction.**

Once all the passes are predicted, the code calculates the attitude and light curve data for each pass. A loading bar, found on stackoverflow [24], estimates how many calculations are remaining, as shown in Figure 5.8. This can be very useful for passes that last several hours, which can take several minutes to simulate. The loading bar shows the user that the code is not frozen.

![Example of loading bars.](image)

**Figure 5.8: Example of loading bars.**
Chapter 6

CONCLUSIONS

In this thesis a light curve simulation tool was developed and verified. This tool can model ground-based observations, space-based observations, self-shadowing geometry, tumbling debris, and nadir pointing spacecraft. While light curve simulation tools have been developed for each of these capabilities, this tool is unique because it incorporates all of them and was developed with the intent to publish the code. This work will save light curve researchers time developing a simulation model and can act as a basis for comparisons between light curve analysis methods.

The tool was verified with STK, matlab, and simulink. It accurately predicts the start and end times of passes within 1% of the total pass duration when compared to STK. The attitude propagator was compared to matlab and simulink code provided by committee member Dr. Mehiel. The tool predicts the attitude motion of the target with offsets less than 0.06° on average and a maximum offset less than 0.6°. It was also shown that edits made to Rush’s reflection calculation code were successful. The code can now accept the specular and diffuse coefficients of the target as user inputs rather than being hard coded for each geometry.

This thesis created a useful simulation tool that could be utilized by light curve researchers. The tool is capable of accurately simulating light curves with several different configurations and use cases, but the user interface can be unwieldy. It requires the user to define all the necessary parameters in a .json file where a single typo or syntax error will stop the simulation from running. Example .json files and
user documentation were developed to help overcome this learning curve, but a GUI with drop downs and built in explanations would be beneficial.
Chapter 7

FUTURE WORK

This section will focus on ways that the tool can be improved and some of the capabilities that can be added to get the simulated data closer to observed data.

7.1 Importing Geometry as CAD Files

Creating a process that allows the user to input CAD files into the tool may be the single most important area for improvement. While the premade geometries serve their purpose as generic shapes, by allowing the user to define the target in a CAD file, the simulation tool would be much more customizable. The user could define the size, shape, mass distribution, and material properties of the target as they desire. The tool would have to be able to break the CAD file into flat facets for the ray tracing algorithm and be able to import the specular and diffuse coefficients of the defined materials. It would also be beneficial to have the code calculate which facets can occlude other facets rather than relying on the user.

A tool that allows the user to specify the exact geometry and materials of the target in a CAD file, without requiring the user to define the facets that occlude other facets, could potentially serve as a thesis project for future students.

7.2 Articulating Solar Panels

The solar panels are highly reflective surfaces on spacecraft. The light curves may be very different if the code updated the orientation of the solar panels to maintain point-
ing towards the Sun as the satellite orbits Earth, rather than keeping them stationary relative to the body frame. Even for inactive tumbling satellites that have stationary solar panels, the light curve is effected by the static orientation of the panels. The code could allow the user to choose between solar panels that are controlled, static defined, or static random, similar to the attitude definition. Simulating articulating solar panels, along with the nadir pointing control loop already implemented, could be used to model active spacecraft and get the simulation closer to real data.

7.3 Noise

Including options to more realistically model noise from atmospheric effects and the sensor could be an important improvement. The noisiness of light curve data is often cited for difficulties with Kalman filter analysis methods, so adding it to the simulation could be useful to researchers [1][3]. This would require more information about the ground site and sensor being used, but could prove worthwhile.

7.4 Attitude Disturbances

The attitude of the target affects the light curve. By including attitude disturbances in the simulation, targets would rotate more like they do in orbit and the light curve would better reflect real data. Many of the attitude perturbations depend on the geometry and inertia of the target, both of which are already utilized in the tool. Therefore, no extra information about the target is needed to implement this improvement.
7.5 User Interface

The .json file for user inputs is a little clunky and unforgiving. A GUI could be developed that allows users to choose options from drop downs and gives a description of each input field. It could have animations for the orbit, attitude dynamics, and reflections. This would allow the user to more intuitively verify that they are running the correct configuration and would show important data in a clear, user friendly manner.
BIBLIOGRAPHY


APPENDICES

Appendix A

TLES

A.1  Cosmos 2414 TLE

1 28521U 05002A 21110.78147721  .00000051 00000-0 30394-4 0 9996
2 28521 82.9536 27.7661 0039264 300.3883 59.3406 13.87631894822932

A.2  Minotaur RB TLE

1 28637U 05011B 21110.54637096  .00000052 00000-0 52399-4 0 9997
2 28637 98.7532 127.1396 0012458 226.5696 133.4440 14.11649562825550

A.3  Breeze RB TLE

1 28527U 05003B 21110.12224794  .00000219 00000-0 00000-0 0 9991
2 28527 18.5312 191.7138 5340687 204.1286 143.3318 1.93171456114356
A.4  AMC-12 TLE

1 28526U 05003A 21110.41195833 -.00000248 00000-0 00000+0 0 9994
2 28526 1.7480 91.6150 0003104 288.2188 299.7385 1.00279247 59427

A.5  SXM-8 TLE

1 48838U 21049A 21284.48382417 -.00000214 00000-0 00000+0 0 9998
2 48838 0.1047 101.4477 0001703 270.0037 97.8079 1.00269451 1480

A.6  NEOSSat TLE

1 39089U 13009D 21124.39907330 .00000023 00000-0 23391-4 0 9993
2 39089 98.4462 325.4368 0012349 57.5444 302.6934 14.34560888428562

A.7  Cal Poly Location

Observation Site Latitude [Deg]: 37.30
Observation Site Longitude [Deg]: -122.66
Observation Site Altitude [km]: 0
Appendix B

PREMADE GEOMETRIES

B.1 BOX

![Diagram of BOX geometry]

Figure B.1: Drawing of premade geometry BOX.
B.2 BOX_WING

Figure B.2: Drawing of premade geometry BOX_WING, with solar panels of zero thickness.
Figure B.3: Drawing of premade geometry PLATE, which has a thickness of zero.
B.4 RECTANGLE

Figure B.4: Drawing of premade geometry RECTANGLE.
Figure B.5: Drawing of premade geometry LONG_RECTANGLE.
Figure B.6: Drawing of premade geometry CYLINDER, which is made of 20 flat facets.
Appendix C

SPACE-BASED ATTITUDE VERIFICATION RESULTS

Figure C.1: Space-based tumbling attitude offsets for Case 1.
Figure C.2: Space-based tumbling attitude offsets for Case 2.

Figure C.3: Space-based tumbling attitude offsets for Case 3.
Table C.1: Space-based average tumbling attitude offsets for all cases, with a requirement of 1e-3 or less.

<table>
<thead>
<tr>
<th></th>
<th>Euler Angle Input</th>
<th>Quaternion Input</th>
<th>MRP Input</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler Angles</td>
<td>4.40e-04</td>
<td>4.39e-04</td>
<td>4.77e-04</td>
<td>Yes</td>
</tr>
<tr>
<td>Quaternion</td>
<td>1.19e-04</td>
<td>8.42e-05</td>
<td>1.92e-04</td>
<td>Yes</td>
</tr>
<tr>
<td>MRP</td>
<td>9.12e-05</td>
<td>8.55e-05</td>
<td>1.54e-04</td>
<td>Yes</td>
</tr>
<tr>
<td>Angular Velocity</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The max offset for each case is shown in Table 4.14. All coordinate frames for all cases had a maximum offset of less than $10^{-2}$.

Table C.2: Space-based maximum tumbling attitude offsets for all cases, with a requirement of 1e-2 or less.

<table>
<thead>
<tr>
<th></th>
<th>Euler Angle Input</th>
<th>Quaternion Input</th>
<th>MRP Input</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler Angles</td>
<td>1.29e-03</td>
<td>1.29e-03</td>
<td>1.41e-03</td>
<td>Yes</td>
</tr>
<tr>
<td>Quaternion</td>
<td>2.66e-04</td>
<td>1.83e-04</td>
<td>3.49e-04</td>
<td>Yes</td>
</tr>
<tr>
<td>MRP</td>
<td>2.01e-03</td>
<td>2.03e-03</td>
<td>2.16e-03</td>
<td>Yes</td>
</tr>
<tr>
<td>Angular Velocity</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Figure C.4: Space-based controlled attitude offsets for Case 1.
Figure C.5: Space-based controlled attitude offsets for Case 2.

Figure C.6: Space-based controlled attitude offsets for Case 3.
Table C.3: Average controlled attitude offsets for all space-based cases with a requirement of 1e-3 or less.

<table>
<thead>
<tr>
<th>Method</th>
<th>ECI Offset</th>
<th>LVLH Offset</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler Angles</td>
<td>1.74e-04</td>
<td>4.47e-07</td>
<td>Yes</td>
</tr>
<tr>
<td>Quaternion</td>
<td>1.73e-04</td>
<td>4.14e-07</td>
<td>Yes</td>
</tr>
<tr>
<td>MRP</td>
<td>1.73e-04</td>
<td>3.96e-08</td>
<td>Yes</td>
</tr>
<tr>
<td>Angular Velocity</td>
<td>6.34e-07</td>
<td>2.48e-07</td>
<td>Yes</td>
</tr>
<tr>
<td>Control Torque</td>
<td>5.30e-08</td>
<td>5.30e-08</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table C.4: Maximum controlled attitude offsets for all space-based cases with a requirement of 1e-2 or less.

<table>
<thead>
<tr>
<th>Method</th>
<th>ECI Offset</th>
<th>LVLH Offset</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler Angles</td>
<td>5.63e-04</td>
<td>1.19e-06</td>
<td>Yes</td>
</tr>
<tr>
<td>Quaternion</td>
<td>5.62e-04</td>
<td>9.45e-07</td>
<td>Yes</td>
</tr>
<tr>
<td>MRP</td>
<td>5.62e-04</td>
<td>9.21e-07</td>
<td>Yes</td>
</tr>
<tr>
<td>Angular Velocity</td>
<td>1.43e-06</td>
<td>6.32e-05</td>
<td>Yes</td>
</tr>
<tr>
<td>Control Torque</td>
<td>1.45e-05</td>
<td>1.45e-05</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Appendix D

USER DOCUMENTATION

This section will document how to download, run, and see the data produced by the light curve simulation tool developed in this thesis.

D.1 Download Code

First the user must download their favorite python integrated development environment or IDE. The author used spyder because its layout is similar to matlab, but any IDE will do. Several libraries must also be downloaded. Some of these libraries may have already been downloaded with the chosen IDE, but below is a complete list of the libraries utilized.

- math
- numpy
- scipy
- json
- spg4
- jplephem
- matplotlib
- datetime
- pandas
- os
- sys
Then the user must download the code from the tool. It is important to save all the code in the same folder so that functions can reference other functions easily. Below is a list of the necessary functions.

- main
- run_sim
- generate_lightcurve
- Reflection_Funcs
- Aero_Funcs
- Control_Funcs

### D.2 Make a `.json` file

Once all the necessary code and libraries are downloaded, the user can begin to create `.json` files to configure the simulation. Included with the code is a list of example `.json` files and the files used in the verification and results sections. The author recommends starting with these files until the user is comfortable with the syntax. For a detailed explanation of the syntax and necessary parameters, refer to the Methodology section where all the possible user definitions are explored.

The author recommends saving the `.json` files in the user defined directory folder. This ensures that the data and the `.json` file used to create the data are stored in the same place. This makes referencing simulation data easier in the future. For consistency the author named the `.json` file and the “Directory” the same thing, but the “Directory” was in all capital letters while the `.json` file name was in all lower case.
D.3  Run Simulation

The simulation can be ran from a console, but the author prefers to use the main file. The path of many different .json files can be saved in the main file so that the simulations can easily be ran again in the future. To run the simulation all that is needed is the path to the .json file as a string and the call out “run.sim(path)”. The path will be different depending on where the user saved the .json file, but an example simulation call out is shown in Figure D.1.

![Image of simulation call out.]

Figure D.1: Example of simulation function call out.

D.4  Data Output

All data will be saved in the “Directory” folder. Each pass will create a new folder in the “Directory” with the start time of the pass as the name of the folder. The data will be saved both as .npy files, which can be easily used in the python environment, and .csv files that can be used in other tools like matlab. Plots will be saved as PNG files. Plots include the light curve, pass prediction, and ECI attitude. If the target is controlled, the LVLH attitude and control torque will also be plotted. If the user wishes to see the plots as pop ups, the user can enter the following into the console: %matplotlib auto. This will allow the user to zoom into the graphs and change the perspective of 3D plots, similar to figures in matlab. The author recommends this
for space-based simulations, as the default space-based pass prediction plot may not always be shown in the best perspective.
Appendix E

GLOSSARY

A - Area

Counts - The amount of photon received by the sensor during a measurement.

CCD - Charge couple device, or component in the telescope that detects and converts the number of photons received to counts.

$C_m$ - DCM from data using Dr. Mehiel code

$C_t$ - DCM from data using thesis code

$C_{lb\, LVLH}$ - Rotation matrix from the LVLH frame to the body frame

$C_{LVLH\, ECI}$ - Rotation matrix from the ECI frame to LVLH frame

$C_{sun,vis}$ - Total flux of visible light from the Sun

$d$ - Distance from the target to the observer
date.s - Global variable for the date

ea - Euler Angles

Earth.exclusion.req - Required Earth exclusion angle that is defined by the user in the simulation configuration

Earth.FOV - The observer’s field of view of that is obscured by the Earth in degrees

ECEF - Earth Centered Earth Fixed frame

ECI - Earth Centered Inertial frame

F.obs - Total visible power of light reflected by the target

I - Inertia matrix of the target

Iₙₙ - 3 × 3 Identity matrix

kd - Derivative gain
$kp$ - Proportional gain

$LBI$ - The transformation matrix that is used to calculate the derivative of Euler Angles

$LVLH$ - Local Vertical Local Horizon frame

$MRP$ - Modified Rodriguez Parameters

$MVC$ - Dr. Mehdi's verification code

$p$ - Variable used to represent Modified Rodriguez Parameters in equations

$p_{total}$ - Sum of reflected intensity of light from the diffuse and specular components

$q$ - Quaternion

$qc$ - Control quaternion

$r$ - Position vector
**range** - Vector from observer to target in ECI frame

**r_obs** - Position vector of the observation satellite in the ECI frame

**r_sun** - Position vector of the Sun in the ECI frame

**r_site** - Position vector of the ground site in the ECI frame

**r_targ** - Position vector of the target in the ECI frame

**Sun_exclusion_req** - Required Sun exclusion angle that is defined by the user in the simulation configuration

**Sun_FOV** - The observer’s field of view of that is obscured by the Sun in degrees

**Tc** - Control Torque

**Tc_s** - Global variable for control torque

**Td** - Disturbance Torque
\( \mathbf{u}_n \) - Unit normal vector off the surface of a facet

\( \mathbf{u}_{obs} \) - Unit vector from target to observer

\( \mathbf{u}_{sun} \) - Unit Sun pointing vector

\( \mathbf{v} \) - Velocity vector

\( \varepsilon \) - Vector part of quaternion

\( \eta \) - Scalar part of quaternion

\( \theta \) - angle around the positive Y axis.

\( \phi \) - angle around the positive X axis

\( \omega \) - Angular velocity

\( \omega_{b.lvlh.s} \) - Global variable for angular velocity of the body relative to the LVLH frame
\textit{beci} - Vector defining the body frame in ECI coordinates

\textit{b lvlh} - Vector defining the body frame in LVLH coordinates

\textit{err} - Difference or error between two measurements

\textit{lvlh eci} - Vector defining the LVLH frame in ECI coordinates

\textit{\cdot} - First derivative