Stereoscopic Light Curve Analysis of Space Debris Objects

Ewan Schafer

DLR

Document Properties

Title	Stereoscopic Light Curve Analysis of Space Debris Objects
Author	Ewan Schafer
Document Type	Master's Thesis
Institute	DLR Institute for Technical Physics
Supervisor 1	Prof. Andreas Nüchter - Julius Maximilians Universität Würzburg
Supervisor 2	Prof. Mathias Milz - Luleå Tekniska Universitet
External Supervisor	Dr. Jens Rodmann - DLR Institute for Technical Physics
Submitted	December 3, 2017

This project has been funded with support from the European Commission. This publication (communication) reflects only the views of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Declaration of Authorship

I hereby declare that the thesis submitted is my own work. All direct or indirect sources are acknowledged as references.

This paper was not previously presented to another examination boards and has not been published.

Signature: _____

Place, Date: _____

Abstract

As an object in orbit rotates, the amount of sunlight which it reflects towards Earth varies. When measured by an observer on the ground, these perceived changes in brightness are know as a light curve.

The shape of a light curve is determined, in part, by the attitude and rotation state of the object but also depends on a great many other parameters. Research into how useful information might be recovered from an object's light curve is an active area of research. If solved it would have important implications in the fields of orbit propagation and active space debris removal. One possible route to better understanding light curves is through stereoscopic observation, where two observers simultaneously measure the brightness changes from different vantage points.

This thesis documents the planning and execution of a collaborative campaign of stereoscopic light curve acquisition between the German Aerospace Center's Institute of Technical Physics and Institute for Planetary Research. In these experiments, the light curves of space debris objects in MEO were observed using a pair of telescopes separated by a baseline of 1595 km. These two observatories being, the Uhlandshöhe Forschungsobservatorium (UFO) in Stuttgart, Germany and the Centro Astronómico Hispano-Alemán (CAHA) in Almeriá province, Spain.

Also documented is the complete development of *Raxus Prime 2.0*, a program for performing the analysis and simulation of stereoscopic light curves.

An early attempt at estimating the spin state of space debris objects from stereoscopic light curves using *Raxus Prime 2.0* is detailed.

This work is dedicated to the memory of Patricia Campsie and David Osborn who I miss dearly.

Special thanks is given to my family and friends for supporting me, financially and emotionally, through my studies.
I would like to thank my supervisor Dr. Jens Rodmann, and all of the staff at the DLR Institute for Technical Physics in Stuttgart, for their guidance and kindness. I would also like to thank Stephan Hellmich at the DLR Institute for Planetary Research, Berlin, for his patience and cooperation in the observations carried out as part of this thesis.

Contents

1. Introduction

In October 2017 the world celebrated the 60th anniversary of the launch of Sputnik 1, the first man-made object placed in orbit. In the six decades since that first space flight, the utilization of space technology has grown into a $340 billion industry [1]. As a consequence of this, the space environment is rapidly becoming crowded with so called space debris consisting of defunct satellites, used rocket bodies and small fragments, all travelling at many kilometres per second. As of November 2017, Celestrack's online Satellite Catalogue (SATCAT) lists 18,808 man-made space objects larger than 10 cm, of which only 1,916 are active payloads [2].

It is important to monitor space for collisions and break-up events - the shedding of debris poses a collision risk for other satellites. The high relative velocity of objects in orbit means that even small pieces of debris can impact with huge amounts of kinetic energy and these collisions can generate even more more debris. Models have shown that the atmospheric drag in LEO is not sufficient to prevent a runaway growth of debris objects. This effect is known as 'Kessler Syndrome' [3].

To date there have been at least four accidental hypervelocity collisions between satellites in orbit. The largest being the 2009 the collision between the defunct Cosmos 2251 and Iridium 33, an active satellite. This collision alone produced upwards of 1500 debris fragments [3]. The largest debris creation event to date was the intentional destruction of the Chinese FY-1C weather satellite in a missile test in 2007. 3037 debris fragments were catalogued as a result [4]. Figure 1-1 shows the density of catalogued objects in low Earth orbit (LEO). The rings of debris resulting from these two events can be distinguished.

With several mega-constellations of satellites planned, the potential for accidental collisions is set to grow considerably.

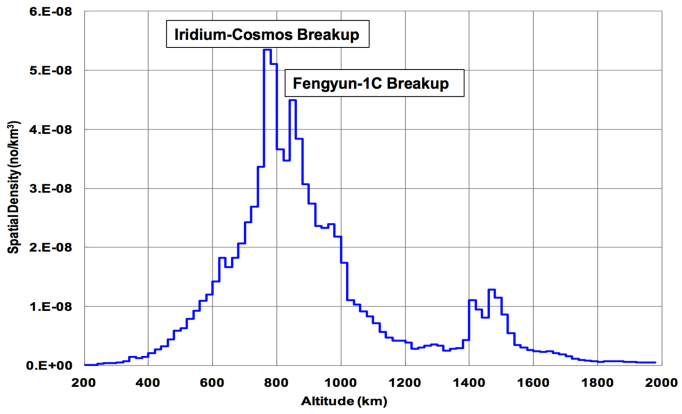


Figure 1-1: Distribution of catalogues debris objects in low Earth orbit [4]

Another area where space debris is a great cause for concern is in geostationary orbit. Because of its unique property of having an orbital period of exactly one sidereal day this is a

popular orbit for telecommunications and weather satellites. At this high altitude, de-orbiting debris is not feasible. Instead, to prevent this region from becoming overly congested, current guidelines suggest that at the planned end-of-life of a geostationary satellite it should move itself into a higher 'graveyard orbit' and purge any remaining propellant to reduce the chance of explosion. Even with this approach many satellites will fail before their planned end-of life. According to the Inter-Agency Space Debris Coordination Committee, in 2015 only 40% of the satellites in GEO which reached end-of-life were in compliance with debris mitigation guidelines [5].

As recently as August 2017 the breakup of the active geostationary satellite Telkom-1 was observed by passive optical means by ExoAnalytic Solutions, a private sector space situational awareness company [6].

In order to maintain safe access to space there is a need to be able to characterise, catalogue and accurately model the orbits of space debris populations. In low Earth orbit the largest nongravitational force acting on a satellite is atmospheric drag. At higher orbits solar radiation pressure becomes the dominant effect [7]. In both cases, these forces are dependent on the cross-sectional area and shape of the object. To accurately propagate orbits it is important to be able to correctly model these forces. Currently the most common approach to this problem is to make the assumption that the objects are spherical (so-called 'cannonball model'). This eliminates the dependence on attitude and shape but has been shown to be a poor approximation, especially for objects which have a high area-to-mass ratio (HAMR)[7].

The forces which result from solar radiation pressure also depend on the reflection properties of the materials which the debris is composed of. The reflection properties of materials in space change over time as they are exposed to radiation. It would be beneficial to be able to monitor the material properties remotely from the ground to better understand the nature of these changes.

The topic of active debris removal to control space debris populations is an area of active research. One approach which is being investigated is the use of powerful pulsed lasers which might impart momentum into the satellite through the ablation of material from its surface [8]. Another proposed solution is to send a satellite, equipped with a robotic arm, to grasp the debris and perform de-orbiting or boosting manoeuvres [9].

In both of these scenarios, the properties of the debris such as its rotation state would need to be determined from the ground beforehand - so that the outcome can be predicted and so that appropriate targets can be selected for removal.

Several papers have suggested that these properties can be determined by observing the changing brightness of sunlight reflected from the debris to an observer on the ground, through a set of techniques known as light curve inversion [10][11][12].

1.1. Light Curves

When a satellite is observed through a telescope, its perceived brightness is seen to vary with time. This one-dimensional time series is what is known as a light curve. Light curve analysis is widely employed in the astronomy community where it has been used to study eclipsing binary stars, discover exoplanets and to characterise asteroids. In the context of space debris there are a number of parameters which determine the amount of reflected sunlight measured by an observer, which include:

- Position of the observer (3 parameters)
- Position of the target (3 parameters)
- Direction of light incident on the target (2 parameters)
- The angular velocity of the target (3 parameters)
- The moments of inertia of the target (3 parameters)
- The material of the target
- The shape of the target

Figure 1-2 shows a simplified diagram of the geometry involved in light curve acquisition. H is the angular momentum vector of the target and is shown in order to indicate the rotation state of the satellite. A normalised vector $\mathbf{n}$ describes the surface normal of a point on the target. A normalised vector $\mathbf{L}$ describes the direction of incoming light from the sun. A vector $\mathbf{R}$ results from the reflection of $\mathbf{L}$ around $\mathbf{n}$. As indicted by the bright cone, $\mathbf{R}$ is only the centre of a distribution of brightnesses in all outgoing light directions, which is related to the Bidirectional Reflectance Distribution Function (BRDF) of the surface's material. β is an angle known as the phase angle and is defined as the angle between the observer, target, and sun. This angle can be used to define a plane known as the phase angle plane.

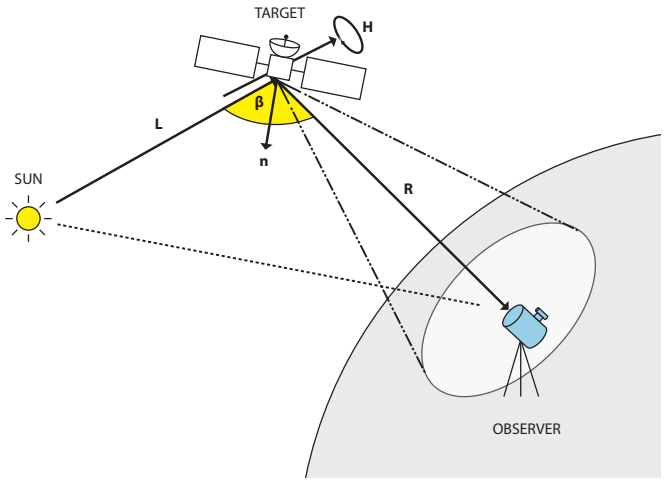


Figure 1-2: A simplified overview of the geometry involved in light curve acquisition

Because light curves are a 1D representation of phenomena which may depend on all of the aforementioned parameters, a well developed model and understanding of the factors which

cause these changes in brightness is required.

Forward modelling of light curves provides an opportunity to study how changes in each parameter correspond to changes in the resulting light curve. Also, in order to assess the validity of existing models, light curves of known objects can be generated and compared to real-world data. Once a model has been proven sufficiently accurate then the real-world parameters might be estimated by fitting simulated light curves to the real data. This is a common approach in the space debris community [13][14][15].

Of course, there are some limitations to this approach. By reducing all of these variables to a single dimension much information will be lost. For example, it can be shown that the exact shape of an object cannot be recovered through its light curve, there can be many differently shaped objects which produce the same light curve [16].

1.2. Stereoscopic Light Curves

An open question in light curves analysis is how much extra information can be gleaned from having two observers, at different locations, simultaneously measuring light curves of the same target.

On beginning this thesis it was suspected that this might be a novel approach to space debris analysis, however it was discovered during literature review that this technique had been trialled by researchers at Odessa National University in 2013 [17]. In that paper researchers conducted simultaneous observations of Envisat from Odessa and Yevpatoria and were able to estimate the rotation axis and rotation rate.

The space debris group at DLR's Institute for Technical Physics in Stuttgart carries out research into satellite laser ranging with the goal of developing a system of autonomous observatories which can discover and track space debris objects. The group operates a satellite laser ranging station close to the city centre of Stuttgart, known as the Uhlandshöhe Forschungsobservatorium (UFO).

A series of joint observation campaigns were planned for the measurement of stereoscopic light curves between the UFO and the Institute of Planetary Research, DLR Berlin, who remotely operate a 1.23m telescope at the Centro Astronómico Hispano-Alemán. These experiments were carried out as part of this thesis.

Detailed information on these two observatories can be found in Section 2.1 and Section 2.2.

The geometry of stereoscopic debris observation would also lend itself to stereo-triangulation of the debris object. Theoretically as bi-product of having two observers tracking the same debris object, the 3D position of the debris could be determined. This might be useful as an easy way of estimating the orbit of the debris. Several attempts were made to do this as part of this thesis but were unsuccessful for reasons described in section 2.4.2.

1.3. Raxus Prime

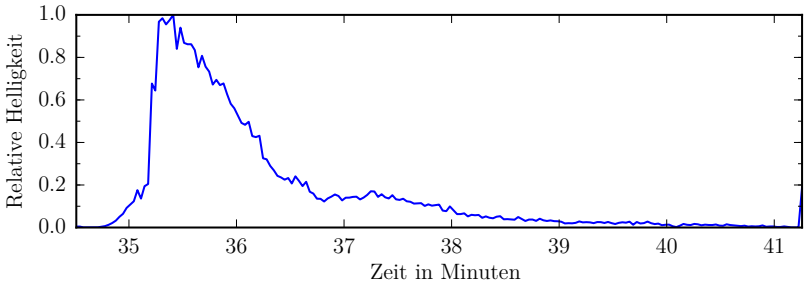
The space debris group at the Institute for Technical Physics at DLR Stuttgart have developed a software package for the analysis and simulation of light curve data. The initial development of this software was carried out by Daniel Burandt as part of his bachelor's thesis in 2016/2017 [18]. This initial version is referred to as Raxus Prime 1.0 in this thesis.

Raxus Prime 1.0 can be used to produce simulated single-observer light curves of known targets. It takes as input a textured 3D model of the target, orbit information about the target in the form of its Two-Line-Element set (TLE) and a number of parameters describing an initial attitude and rotation. These parameters are propagated forward to the desired times and as the satellite is rendered as it would be seen by an observer. The sum of the pixel values in the resulting image is used as a brightness value for the simulated light curve.

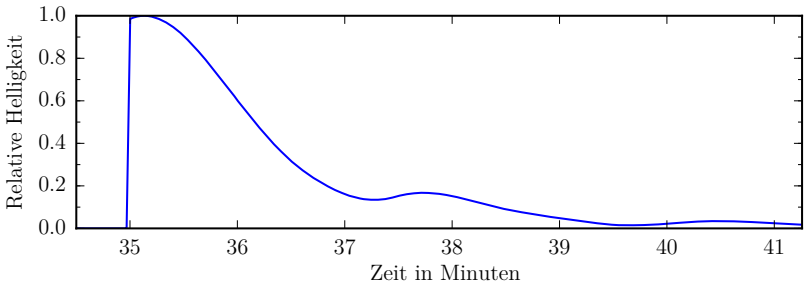
To produce the images, from which a light curve can be extracted, Raxus Prime uses the open-source ray tracing software POV-Ray [19]. The benefit of this is that POV-Ray's sophisticated rendering algorithms can be used to model complex phenomena without having to implement a custom reflection model. However a severe drawback is the very long rendering times required. A discussion of POV-Ray and rendering techniques is included in section 4.4.

The simulations in Raxus Prime 1.0 are exclusively a forward modelling tool. It is possible for a user to manually change the input parameters to the simulation but there is no way for the user to directly compare the result of this simulation with a real light curve in-program. That means that parameter fitting is currently not possible. Nevertheless, by manual tuning of the parameters, similar features can be produced as those which appear in real light curves. This is a good indication that the approach to simulation in Raxus Prime 1.0 is valid. Figure 1-3 shows an example of a simulated light curve produced with Raxus Prime 1.0 with manually tuned parameters, compared to a real measured light curve. Each run of a simulation of this kind would take 15 minutes to render in Raxus Prime 1.0 - this means that parameter fitting, which requires many simulations to be run, is not feasible with the Raxus Prime 1.0 simulation pipeline.

One of the primary goals of this thesis was to upgrade Raxus Prime 1.0 to allow the analysis and simulation of multi-observer light curves. This task is documented in chapter 5.



(a) Measured light curve



(b) Simulated light curve

Figure 1-3: Similar features can be seen in the real light curve of a Zenit rocket body in LEO and those produced in simulation by Raxus Prime 1.0 [18]

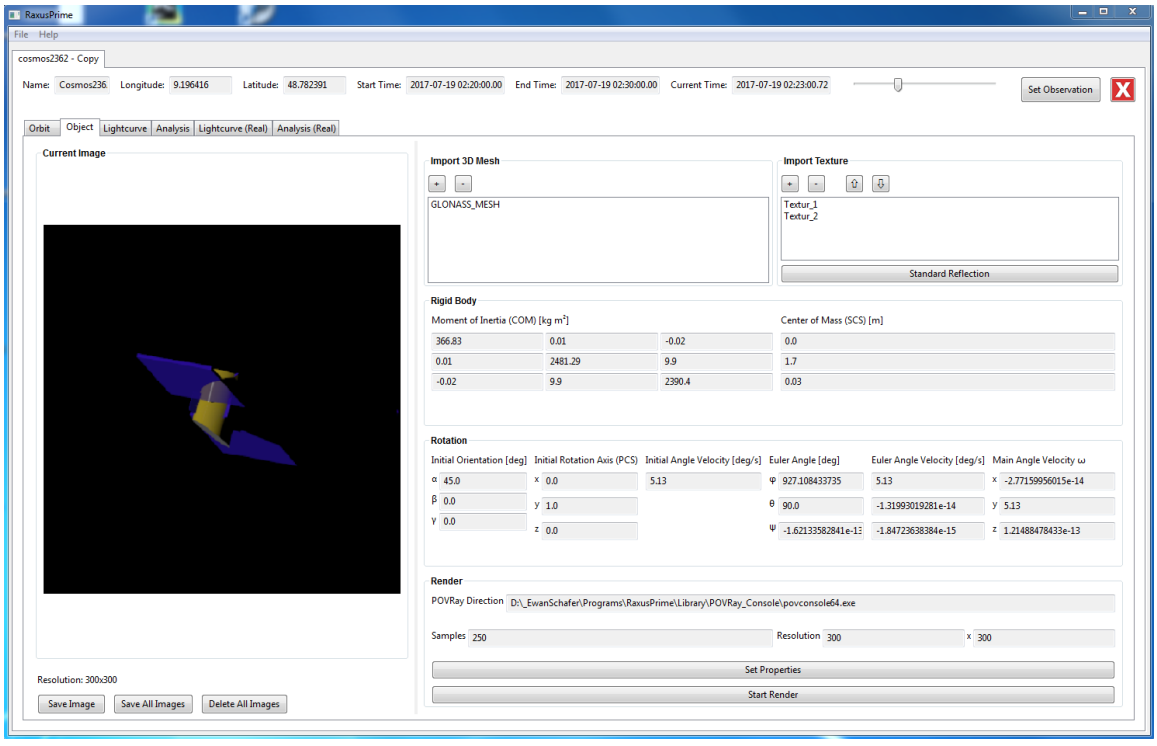


Figure 1-4: Screenshot of Raxus Prime 1.0

2. Measurement of Real Light Curves

2.1. Uhlandshöhe Forschungsobservatorium

Figure 2-1: The 43 cm telescope at the Uhlandshöhe Forschungsobservatorium, Stuttgart

The UFO is a small observatory located in Stuttgart, Germany. It lies at 48°46'56.6" N, 9°11'47.1" E at an altitude of 399 m above the WGS84 Earth ellipsoid. It is primarily used for research into satellite laser ranging and is an engineering station for the International Laser Ranging Service (ILRS).

The telescope is a PlaneWave corrected Dall-Kirkham with a 43 cm (17") diameter primary mirror and a focal length of 3 m. It is mounted on an ASTELCO NTM-500 series equatorial mount which provides a maximum slewing velocity of $20°s^{-1}$. The UFO has an absolute pointing accuracy of around 30 arcseconds although a tracking accuracy of around 2 arcseconds can be achieved through closed loop tracking (see section 2.4.2). A drawback of using the equatorial style mount is that when tracking objects which pass close to zenith, the mount must perform a flip across the local meridian to avoid collisions of the telescope with the legs of the mount. Although the telescope slews quite quickly, the time taken to perform this manoeuvre can cause loss of tracking and loss of data when observing fast-moving targets.

The sensor used is an Andor Zyla 5.5 sCMOS which has a full-frame resolution of 2560×2160 pixels with a maximum dynamic range of 25,000:1. High dynamic range is important for our purposes because the observed brightness of space debris can vary dramatically as it rotates. It is crucial to be able to capture both the high and low ends of the light curve without having to change the exposure time during the observation. The sensor is sensitive to light with wavelength between 300 and 1000 nm, with a peak quantum efficiency of 60% at 600 nm. Light entering the sensor is not filtered, this means that it is able to capture reflected sunlight from space debris across the visible spectrum and a small amount of the near-infrared. After the focussing optics, the sensor has a full-frame field of view of 0.33°.

A Photonic Solutions ID400 single photon detector is also mounted on the telescope. This is used to precisely time laser light returns during satellite laser ranging experiments. Some experiments assessing the suitability of this instrument for passive-optic light curve experiments were carried out as part of this thesis and are discussed in section 7.1.3.

The observatory primarily uses its CMOS sensor for closed-loop tracking of objects to keep the target within the narrow field of view of the single photon detector. For this application a very high frame rate is not important. As such the maximum frame rate of the sensor is currently limited between 0.5 and 1Hz, with the limiting factor being the time taken to transfer the image from the sensor to the computer. The telescope is operated using a sophisticated custom control system and user interface, known as OOOS (Orbital Objects Observation Software) [20].

2.2. Centro Astronómico Hispano-Alemán

Figure 2-2: The 1.23m telescope at the Centro Astronómico Hispano-Alemán, on Calar Alto, Spain

The Centro Astronómico Hispano-Alemán (CAHA) is an observatory located on Calar Alto, a mountain in Almeriá province, Spain. It lies at 37°13'25" N, 2°32'46" W at an altitude of 2,212 m above the WGS84 Earth ellipsoid.

The observatory has three telescopes on site, with primary mirror diameters of 1.23 m, 2.2 m, and 3.5 m. DLR's Institute for Planetary Research leases the 1.23 m aperture telescope approximately 100 nights per year, and is able to remotely operate it from Berlin.

The sensor used in the stereoscopic light curve campaigns was the DLR MKIII CCD which has 4096×4112 pixels. The sensor was operated in frame transfer mode to achieve faster frame rates. With 3×3 pixel binning and fast readout mode the sensor is able to achieve a frame rate of 1 Hz.

The sensor has a full frame field of view of 21×21 arcminutes but in frame-transfer mode this

is reduced to 5×10 arcminutes. This is a fairly small field of view. Using TLEs as the source of orbit information and SGP4 as the orbit propagator can produce errors in the predicted target position greater than this accuracy, especially for targets in LEO.

The telescope is mounted on an equatorial mount and, although the theoretical maximum slewing rates for this mount are unknown, experimentally it has been determined that the telescope mount is able to slew fast enough to track LEO objects. In practice, however, the dome of the telescope was found to rotate too slowly to keep up. Furthermore, the dome is not able to rotate more then 360°, meaning that when tracking a target which moved past this limit, the dome would have to reverse direction and move a full rotation.

2.3. Campaign Planning

Consideration was given to the factors which govern a stereoscopic light curve observation. This was done both to prepare for the UFO/CAHA campaigns and to assist the the planning of any future campaigns.

2.3.1. Baseline Calculation

The straight line distance between two observers is known as the baseline. This is an important factor in stereoscopic observations for a number of reasons. The greater the baseline the more different the viewing geometry of the two observers will be. The difference in phase angle will be larger and in some situations different faces of the target satellite will be visible.

To calculate the baseline the geodetic coordinates of the observers are first converted to Earth Centred, Earth Fixed coordinates using eq. (2-1). Geodetic coordinates are Latitude, ϕ. Longitude λ, and altitude above the surface of the WGS84 ellipsoid, h.

$$\mathbf{r}_{\mathrm{ECEF}} = \begin{bmatrix} (\frac{a}{C} + h)\cos\phi\cos\lambda \\ (\frac{a}{C} + h)\cos\phi\sin\lambda \\ (\frac{a(1-f^2)}{C} + h)\sin\phi \end{bmatrix} \tag{2-1}$$

Where $a = 6378.137$ km is the equatorial radius of Earth, $f = 1/298.25722$ is the WGS 84 flattening factor and $C = \sqrt{(1 - f^2\sin^2(\phi))}$

The baseline is then the magnitude of the difference between the ECEF position of two observers. In the case of CAHA and UFO the baseline is 1595 km.

2.3.2. Earth-Central Angle

The angle between the centre of Earth and two observers, σ is most easily calculated using the dot product of the position vectors of the two observers (providing that these are defined in an earth centred frame):

$$\sigma = \arccos\left(\frac{\mathbf{r}_1 \cdot \mathbf{r}_2}{||\mathbf{r_1}||\,||\mathbf{r_2}||}\right) \tag{2-2}$$

The Earth-central angle between UFO and CAHA is 14.37°.

2.3.3. Minimum Mutually Observable Object

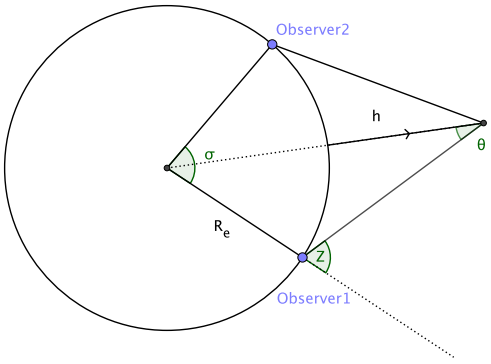
To help determine which objects it might be possible to observe from two locations on Earth, the lowest altitude point which could be jointly observable by the UFO and a second observer was calculated for a grid of observer locations.

This is not intended to be an accurate calculation - more of an estimate to use as a rough guide for the planning of future stereoscopic observations. Because of this, the assumption of a spherical Earth is made for simplicity. This means that the minimum mutually visible point will be at some height above a point directly between the two observers.

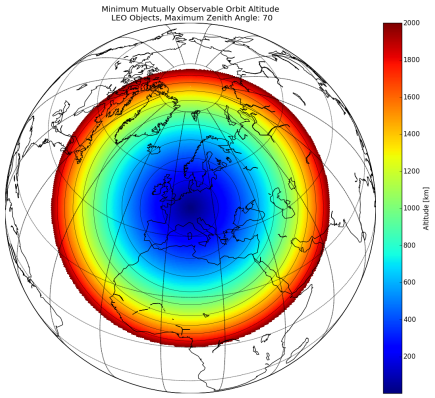
We assume that the two observers are able to observe objects below a maximum zenith angle of z_{max}. By inspection of the geometry shown in fig. 2-3a it can be seen that:

$$h = R_e \left[\frac{\sin(z_{max})}{\sin(z_{max} - \frac{\sigma}{2})} - 1 \right] \tag{2-3}$$

where R_e is the mean Earth radius (6371 km). Figure 2-3b shows the minimum altitude of LEO objects which would be mutually visible with the UFO, at each location on Earth.



(a) Geometry



(b) Minimum altitude of LEO objects which would be mutually observable with UFO for observers with $z_{max} = 70°$.

Figure 2-3: Baseline geometry and minimum mutually observable orbit for two observers on a spherical Earth.

2.3.4. Target Visibility Analysis

A preparatory investigation was carried out into the criteria which determine when and for how long objects are mutually visible by two observers.

In order for a successful stereoscopic passive optical observation to take place, four basic requirements must be met:

1. **Both observers must be in darkness:**

Because the atmosphere of Earth scatters sunlight, during times when the sun is up, the signal to noise ratio of the reflected sunlight from the target is so small as to be essentially 0. This effect persists for some time even after the sun has fallen below the horizon (twilight).

The day can be split into 5 classifications based on the position of the sun. Representing the local elevation of the sun, in horizontal coordinates with the symbol $El_{\odot}$, and neglecting the atmospheric refraction, the times of day can be defined in the following way:

- Day ($El_{\odot} > 0°$)
- Civil Twilight ($-6° < El_{\odot} \leq 0°$)
- Nautical Twilight ($-12° < El_{\odot} \leq -6°$)
- Astronomical Twilight ($-18° < El_{\odot} \leq -12°$)
- Night ($El_{\odot} \leq -18°$)

Figure 2-4 shows the sun rise, sun set and twilight times for UFO and CAHA for every day of the year in 2017. It was found during the observation campaigns, that passive optical observations could be carried out between evening and morning nautical twilight ($El_{\odot} < -6°$).

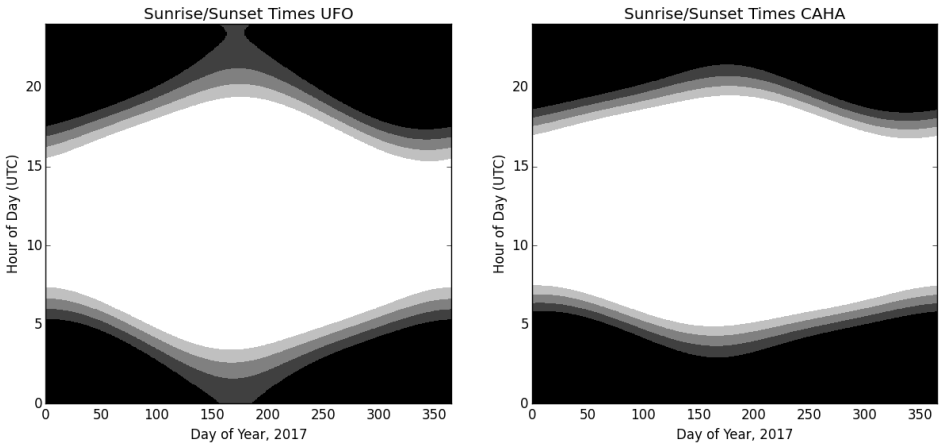
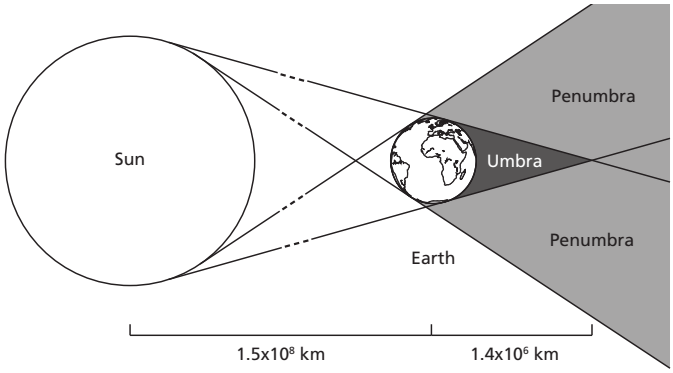


Figure 2-4: The annual variation of day, night and twilight times for UFO and CAHA. The times corresponding to the above mentioned classifications of twilight are shaded from white (day) to black (night)

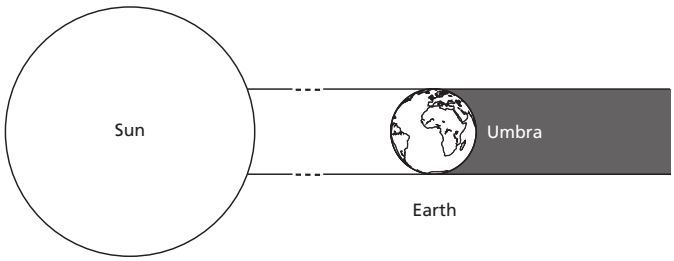
2. **The target must be in the field of view of both observers:**
   This can be determined by propagating the orbit of the target forward to a particular time and transforming the resulting position into local horizontal coordinates for each observer. In this frame it is possible to check if the target will be above the horizon for each of the observers.

3. **The target must be in direct sunlight:**
   Especially for targets in LEO it is important to consider that the orbit may pass into the shadow of the Earth and when this happens the object will cease to be visible.

The shadow of the Earth has conical structure (as show in fig. 2-5a) due to the fact that the sun is a finite distance from the earth, and has a larger radius than the earth. An outer cone, called the penumbra results in areas where the sun is partially eclipsed by the earth. An inner cone called the umbra is the region where the sun is completely eclipsed by the Earth - this region extends to approximately 1.4×10^6 km from the Earth. A third region, the antumbra, occurs beyond the umbra, where an annular eclipse would be observed.
If the Sun is assumed to be a point source at infinity, the light at Earth will have parallel rays and the shadow of the earth can be modelled as a cylinder (as shown in fig. 2-5b). Since the distance to the sun is so great and artificial satellites orbit relatively close to Earth this is actually a reasonable assumption.
It is feasible that an object might be visible through reflected earthshine, city lights or moonlight but these effects are not considered in this thesis.



(a) Conical shadow model



(b) Cylindrical shadow model

Figure 2-5: Two models of the shadow of Earth which are almost equivalent for the altitudes at which satellites orbit Earth. Neither the size or the distances are to scale.

4. **The weather conditions must be favourable for both observers:**
   One of the biggest issues with stereoscopic observations is that it requires skies to be clear at the same time in two locations. Because the observed brightness of the target is being compared between frames, even slight cloud cover can produce inaccurate results.
   To obtain insight into the average cloud cover around the world, MODIS Cloud Product data was downloaded from the NASA Earth Observation web portal. The MODIS cloud product, contains historical daily, weekly and monthly averages of local cloud fraction

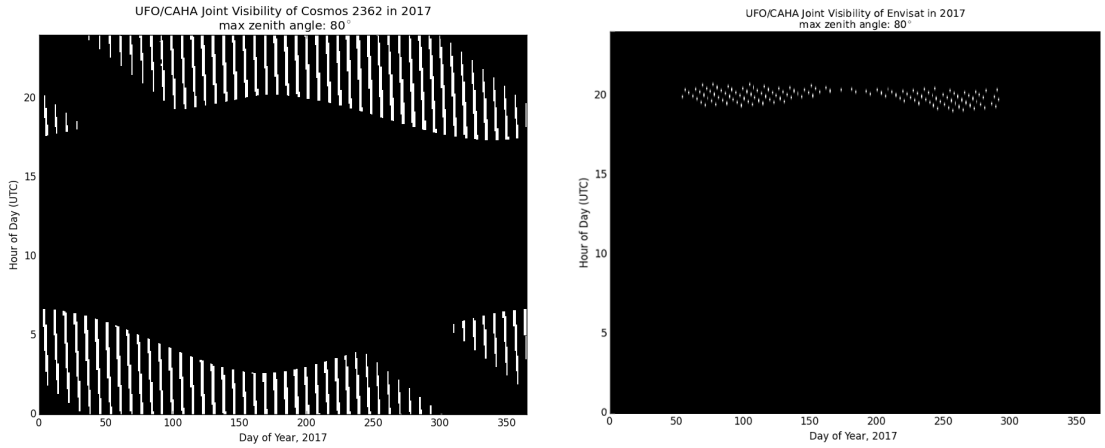
(CF). Although not strictly accurate, we can take the cloud fraction to be approximately equal to the probability that an observation will fail. If we make the assumption that the weather at the two sites are completely probabilistic and mutually independent, then a rough probability that an observation will be successful might be estimated by:

$$P_{success} \sim (1 - CF_1)(1 - CF_2) \tag{2-4}$$

This method estimated a probability of successful observations between UFO and CAHA in June 2017 of 31%, and 24% in July (Real success rates were 8% in June and 25% in July - see section 7.1.4)

To gain an understanding of how periods of joint visibility vary for objects in different orbits, the above criteria were evaluated for every minute in an entire year. Two satellites were considered, COSMOS 2362, a MEO debris object, and Envisat - a LEO debris object. All periods of mutual visibility for these two objects are shown in fig. 2-6. Times when the debris would be mutually observable by UFO and CAHA are shown in white.

The LEO object has much fewer and shorter periods of joint visibility - the low altitude of the object means that the satellite is visible to a smaller area of the surface of Earth at any given time. The periods of visibility form a band close to twilight. This is because there is only a short period of time where the observer is in darkness and the satellite is not in the shadow of Earth. In contrast, the MEO object spends less time in the shadow of Earth and is visible from a larger area.



(a) Cosmos 2362 (MEO Object)

(b) Envisat (LEO Object)

Figure 2-6: Times of mutual visibility for CAHA and UFO in 2017, assuming a maximum zenith angle of 80°.

2.3.5. Target Selection

In order to try and obtain the best possible data for the observation time available. A list of candidate target objects was compiled according to the following criteria:

- The target should be in MEO, to allow tracking by CAHA
- The target should have a rotation frequency $<<$ the Nyquist frequency due to the image frame rate of the telescope.
- The target should have a rotation period $<<$ the duration of a pass, so that multiple rotations can be seen in a pass.
- The target should be a defunct satellite, which is intact, so that its shape can be assumed to be known.

MEO is relatively sparsely populated, so the choice of potential targets is limited.

A 2015 Paper by Steindorfer et al, describes the measurement of rotation periods of satellites though light curve analysis [21]. Included in this paper is a list of determined rotation periods for many defunct GLONASS satellites. The objects in this list which have rotation periods less than 120 seconds were selected as candidate targets for this thesis because this would allow multiple rotations to be observed in the short observation slots which were available.

Also selected were defunct Molniya satellites, which were an attractive choice due to their fixed orientation solar panel arrays - this means there is less uncertainty about the shape of the object. Molniya objects have highly elliptical orbits and it was hoped that tracking these objects as their altitude decreased might help test the limitations of CAHA's tracking capabilities.

Defunct MIDAS satellites were also chosen because they are very large bright objects. They have relatively low orbits for MEO objects. This would result in a larger angle between the observers and the satellite.

2.3.6. Pass Planning

Although Raxus Prime 2.0 could now, theoretically, calculate passes for stereoscopic observations, at the time that the observation campaigns were underway, this feature had not yet been developed. Instead, pass planning was carried out using AGI's Systems Tool Kit (STK) software package [22].

An STK scenario was created which includes the list of target candidates and the location of the two observers. A zenith-facing sensor was added to both observers with a field of view of 160° ($2z_{max}$). The STK scene was propagated forward to a time when an observation was planned to take place.

A list of targets which are mutually visible at that time, is generated and pass reports were produced which included times and Alt-Az tracks for both sites - these were communicated with Calar Alto.

On the night of the observation, the list of target candidates was loaded into the UFO's control software OOOS, which commands the slewing of the telescope for the pass.

2.4. Image Acquisition

With regards to the acquisition of light curve data there are two possible approaches:

1. Streak Imaging & Leapfrog tracking
2. Closed-loop tracking

Currently both approaches require prior knowledge of the target's orbit. This information comes in the form of the Two Line Element set of the satellite which is freely available online. The TLE of the object is recorded by OOOS at the time of observation and saved as part of the observation log for later use. This is important because the TLE's are required for analysis and simulation of the observation and historical TLEs are generally not available online.

2.4.1. Streak Imaging

In this approach the telescope is slewed to a position in the sky where it is predicted that the satellite will pass though the field of view. The sensor begins a number of long exposures and waits until the satellite has passed. It then tracks to a point ahead of the satellite again, in a 'leapfrog' type motion.

In this kind of image the satellite will produce a streak of light against a background of stationary stars. An example of this kind of image produced at the UFO is shown in fig. 2-7a[23]. This image shows the variations of brightness along the track of the satellite.

In this approach the light curve would be extracted from the image by examining the brightness of pixels along the streak. This means that the temporal resolution of the resulting light curve would only limited by the speed of the target and the size of the pixels.

It was decided to not acquire light curves in this mode because dividing the streak into even time sections requires precise knowledge of the orbit and because this method is not suitable for faint objects.

(a) Leapfrog Tracking (b) Closed-Loop Tracking

Figure 2-7: Tracking Techniques

2.4.2. Closed-Loop Tracking

The approach which was decided on for this thesis was a closed-loop tracking approach. In this method shorter exposures are taken constantly and image analysis is performed on these on-the-fly to identify the target. A PID controller attempts to keep the target centred in the image.

This functionality is available through OOOS and is the method used to centre the target in

the narrow field of view of the single photon detector when the UFO is being used for laser ranging.

This approach makes it much easier to extract photometric (brightness) information from the images and also means that fainter objects can be imaged. It has the disadvantage of limiting the time resolution of the light curve to the total exposure and readout time of the image.

Another slight disadvantage is that with fast-moving targets the background stars appear stretched into streaks in the resulting image. This makes it very difficult to carry out astrometry on the images - the algorithms traditionally used for this attempt to identify stars in the image and match them to star catalogues to determine the direction and field of view of the image. By extension this makes it possible to determine the perceived celestial coordinates of any other objects in that image. Importantly these star catalogues contain information on the brightness of stars - this means that the apparent brightness of the target can be calibrated relative to these known values in a process known as calibrated photometry. An open source implementation of this process is available and is descriped in Mommert 2016 [24]. This software is capable of extracting the apparent magnitude to ≤ 0.03 mag and the angular position to within 0.3 arcsec. Performing calibrated photometry would be very valuable to the subject of stereoscopy light curved since it would allow the comparison of absolute values of brightness, rather than only comparing two normalised datasets.

Several attempts were made to use a different online tool, astrometry.net [25] to assess the feasibility of carrying out astrometry on the images obtained using closed-loop tracking. All attempts at this failed. One such failed attempt is shown in fig. 2-8.

It is feasible that stereo-triangualtion could still be performed by obtaining the pointing direction from the encoders of the telescope mounts but to do so would require future work to be done in calibrating the telesopes and sensors. For example, the precise orientation of the sensor's imaging plane relative to the telescope itself has not yet been determined.

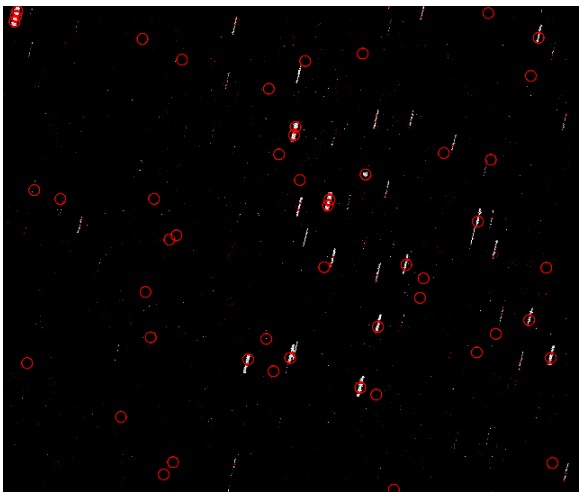


Figure 2-8: Failed attempt at performing astrometry on an image of COSMOS 2362 obtained using the closed-loop tracking technique.

2.5. Image Correction

There are a number of sources of noise in a digital image which can be corrected by studying the noise characteristics of the sensor which is used to take the images.

Python routines for performing these corrections were written as part of this thesis. Although many astronomy software packages already exist which are able to perform these corrections, the custom implementation is tailored to the file structure produced by OOOS and therefore can be run with minimal input from the user and potentially fully automated and integrated into OOOS in the future.

2.5.1. Bias Frame

We can characterise the readout noise of the sensor by taking a series of images with the shortest possible exposure time that the sensor will allow. This series of images is combined in a median stack. Figure 2-9a shows the bias frame which was obtained at the UFO.

The Andor Zyla 5.5, the sensor used at the UFO, is an sCMOS sensor. CMOS sensors differ from CCDs in that each individual pixel performs its own charge to voltage conversion. Then this is converted to a digital signal one row at a time. The Zyla has a split readout architecture, where the two halves of the sensor are read out. This can be seen in the bias frame (fig. 2-9a) - the two halves of each column of sensor have different readout noise.

The bias frame is subtracted from the raw images to remove read-out noise.

2.5.2. Dark Frame

Thermal noise is especially a problem for astronomical imaging where the exposure times are usually long and the sensitivity of the sensor is usually set high. This increases the signal from thermal electrons. The sensor at the UFO is thermoelectrically cooled to around 0°C to reduce this effect.

The amount of thermal noise being generated by the sensor is estimated by taking exposures when there is no incoming light. In practice this was done by taking a run of images at night, with the observatory dome closed and dust cover on the telescope. The images are combined in a median stack and the previously calculated median bias frame is subtracted from this. Figure 2-9b shows the resulting dark frame for the sensor at the UFO.

The assumption is made that the dark current in the sensor scales linearly with time. Making this assumption means that we can use the same dark frame for correcting images with different of exposure times. The prepared dark frame is multiplied by the exposure time of the raw image which is to be corrected and is then subtracted from it.

2.5.3. Flat Field

Even in the case of perfectly uniform incoming light, not all of the pixels on the sensor would register the same amount of light; each pixel has a slightly different sensitivity.

This can be due to the pixels themselves having slight irregularities in their manufacture but a more pronounced effect is due to areas of darkness cast onto the imaging surface by dust or

scratches on the optics.

Pixels further from the optical axis will appear darker due to a series of effects, collectively know as vignetting. Vignetting can be caused by the geometry of the optical elements in the telescope which may cause light to be blocked. Further from the optical axis, light will be incident on the sensor at a shallower angle. A photon which has shallower angle of incidence will be less likely to liberate an electron in the sensor. The Andor Zyla sensor is actually fitted with a microlens array which helps to combat this problem.

Correcting these flat field effects is important to light curve analysis because it would be disastrous if the measured brightness of a target depended on where its image fell on the imaging sensor.

To correct for these effects a series of images called flat frames are created and applied in the following way:

1. A series of images of a uniform light source were taken. At the UFO, this was done by imaging the sky during civil twilight.
2. This series of images is combined in a median stack which then has the bias frame and dark frame subtracted from it.
3. The brightness of the resulting image is normalised to 1.
4. Raw images are divided by this normalised flat field image. This has the effect of brightening the areas which would otherwise erroneously appear dark.

Figure 2-9c Shows a flat field image for the telescope at the UFO. Vignetting and doughnut-shaped optical artefacts from motes of dust are clearly visible.

2.6. Light-Curve Extraction

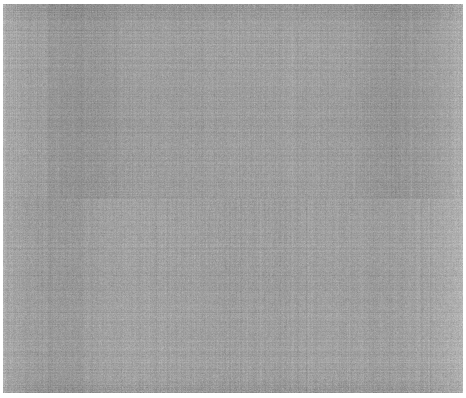
Light curve extraction was carried out using DLR's in-house software *Light Curve Maker*. This program attempts to automatically track and extract the relative changes in brightness of a user-specified target in a sequence of .fits images.

To do this, the user is required to specify a region of interest (ROI) around the the target in a number of the source images. A second field of view (FOV) box is specified and this is the area in which light curve maker looks for the target object in the following frame.

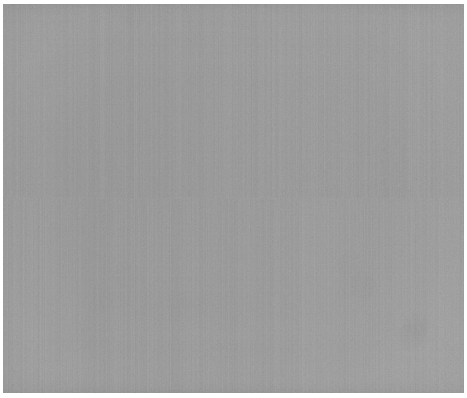
The background noise level is estimated from the average pixel value of the border formed around the FOV by the ROI. If the image corrections described in section 2.5 have not been applied then this can also act as a rough approximation the sensor noise correction but is a less sophisticated approach.

This noise level is then subtracted from the pixels inside the ROI and the sum of these corrected pixel values for each of the source images is saved along with the time that the image was captured. This is the light curve. The time stamp saved in the light curve corresponds to the time at the middle of the exposure (t_0 + 0.5t_{exp}).

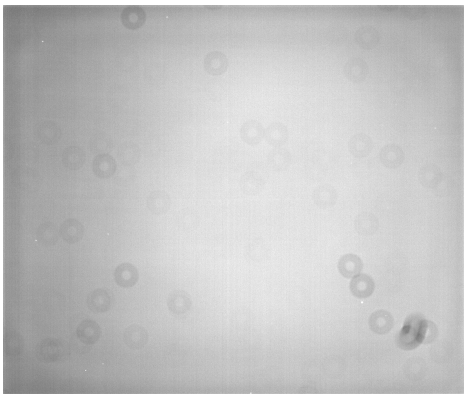
Modifications were made to Light Curve Maker's source code as part of this thesis project, to



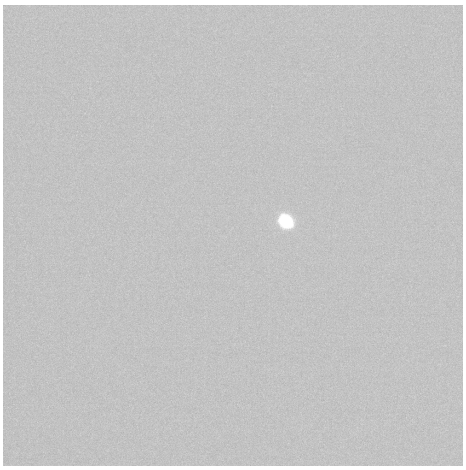
(a) Bias Frame



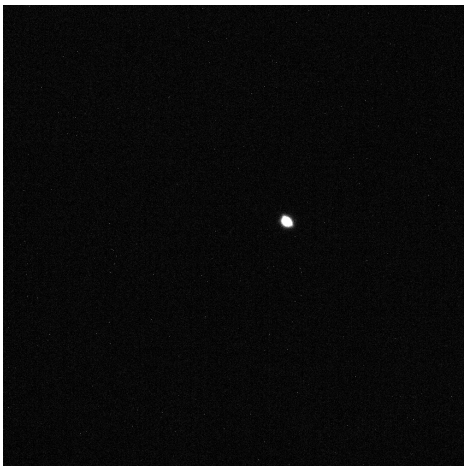
(b) Dark Frame



(c) Flat Field



(d) Raw Image



(e) Corrected Image

Figure 2-9: Calibration frames for the UFO telescope. The brightness and contrast of all of these images have been stretched to highlight the features.

allow it to accept images from CAHA, which have a different file header.

A shortcoming of this approach is that if a background star streak passes though the field of view specified, the brightness of the satellite will be incorrectly measured. In this case it is

better to disregard the frame completely and have a gap in the data rather than an incorrect value.

2.7. Light Curve Correction

The rotation of the object is not the only factor which determines the observed brightness. Extra corrections can be applied to the extracted light curve to eliminate these effects and isolate the changes due to the object's rotation and shape.

In all of the observations in this thesis, the orbit of the object was known - the target was selected in advance and the telescopes were pointed according to the target's TLE, which was recorded at the time of the observation.

2.7.1. Range Correction

The observed brightness, I, of any object falls off with the inverse square of the distance between the object and observer, x. The relative distances between the observer and a target in orbit can vary greatly over the course of an observation. If this distance is known, then this change can be corrected for, as shown in eq. (2-5).

$$I_0 = Ix^2 \tag{2-5}$$

In theory this calculation should also depend on the distance of the target from the sun. However in the worst-case scenario of a satellite in geostationary orbit at the time of Earth's perihelion, the insolation on the far side of the orbit is still 99.9% of the insolation on the near side. The target's change in distance from the sun over the course of an observation is such small fraction of the total distance from the sun that its effect is considered negligible in this thesis.

2.7.2. Airmass Correction

Due to the Beer-Lambert law of attenuation, the amount of light received by an observer will be reduced exponentially by the amount of atmosphere through which the observer is looking.

In astronomy, the amount of air through which light must travel to reach an observer is often measured in units of relative airmass. 1 unit of airmass is the amount of air an observer at sea level would experience when looking toward zenith.

A commonly used approximation for calculating the airmass for a given zenith angle, z, is to assume a slab of atmosphere of uniform density and a flat earth. This is known as the parallel plane model and is a good approximation for the true airmass, below a zenith angle of 80° (see fig. 2-10). In this approximation, the airmass, X can be calculated as:

$$X = \sec z \tag{2-6}$$

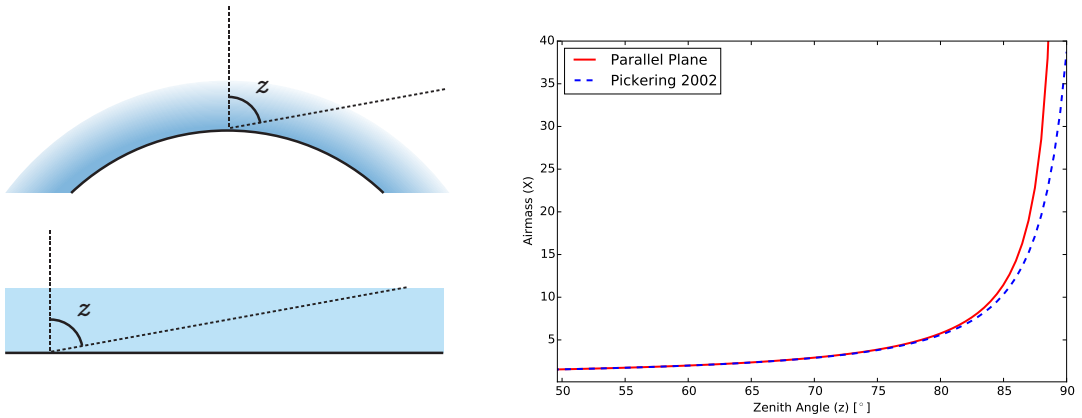
Applying the Beer-Lambert law gives the relationship between the true brightness I_0 and the observed brightness I:

$$I = I_0 \, A \, e^{-\tau X} \tag{2-7}$$

Where τ is the aerosol optical depth of the atmosphere, and A is some constant factor.

No effort is made to determine the aerosol optical depth of the atmosphere while observing (eg. by observing a star with known brightness) because in the context of normalised light curves only the relative changes in brightness are of interest, not the absolute brightness of light reflected from the satellite. This requires the assumption that this value remains constant over the course of an observation.

This is a valid assumption but in future observations there would be value in measuring several stars of known brightness as part of the calibration procedures. Since knowing the absolute amount of light being reflected could help place constraints on the size, shape and material of the satellite.



(a) Above: A realistic model of the atmosphere Below: The parallel plane simplification.

(b) Comparison of the parallel plane model and an empirically derived model.

Figure 2-10: Validity of the parallel plane atmospere model for airmass

2.8. Sucessful Stereoscopic Observations

From the 18 nights where telescope time was allotted at CAHA only two usable joint observation light curves were obtained. Even during these observations weather conditions were far from ideal. Appendix A shows the full record of observations. The two successfully joint observations were:

- COSMOS 1988 on 2017-06-01
- COSMOS 2362 on 2017-07-19

This low success rate was due to the limited telescope time available for this project at CAHA but most importantly because of adverse weather conditions and the increased probability of

failure which arises from joint observations as described in section 2.3.4.

Another two successful joint observation light curves were also captured in late November 2017. However, too late for inclusion in this thesis.

2.8.1. COSMOS 1988

On 2017-06-01 a joint observation of the defunct GLONASS object COSMOS 1988 was conducted. Weather conditions on this night were ideal at CAHA but poor in Stuttgart. An attempt at capturing joint observations of another satellite, Molniya 1-75 had failed in the 15 minutes previous due to patches of thick cloud cover. During the observations of COSMOS 1988, the satellite was actually being viewed through cloud cover and the noise which arose from this is visible in the light curve.

At CAHA 642 frames of the satellite were captured between 20:55:01 UT and 21:06:01 UT.

At UFO 685 frames of the satellite were captured between 20:46:01 UT and 21:09:10 UT.

Figure 2-12 shows the normalised stereoscopic light curve which was obtained from joint observations on 2017-06-01. A logarithmic scale is used in fig. 2-11a to account for the large variations in brightness. Through frequency analysis, detailed in the following section, the debris was determined to have a rotation period of 8.208 s as measured from the UFO light curve and also 8.208 s as determined from the CAHA data.

2.8.2. COSMOS 2362

On 2017-07-19 a joint observation of the defunct GLONASS object COSMOS 2362 was conducted. The weather at the UFO was ideal for this observation but the weather at CAHA was poor. As with the previous successful joint observation, one of the observers was observing COSMOS 2362 through cloud cover. This is reflected in the noisy light curve data obtained at CAHA, which can be seen in Figure 2-12.

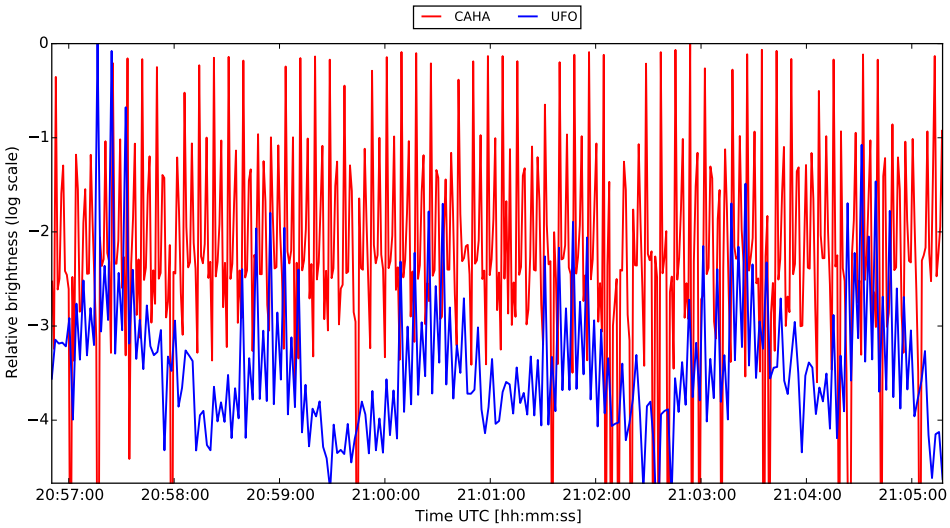
At CAHA 495 frames of the satellite were captured between 02:07:02 UT and 02:28:48 UT.

At UFO 1240 frames of the satellite were captured between 01:48:47 UT and 02:30:39 UT.

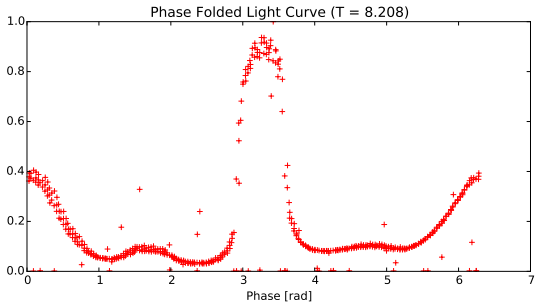
Figure 2-12 shows the normalised stereoscopic light curve which was obtained from joint observations on 2017-07-19. A logarithmic scale is also used in fig. 2-12a. In the UFO data the peaks are $> 100\times$ larger than the lowest measured brightness. In this data the periodicity of the light curve is determined to be 70.21 s for the UFO data and 70.50 s for the data from CAHA. It is difficult to draw any conclusions from this difference in frequency, given that the data is of such poor quality, and that only 6 full periods were observed.

On viewing initially, the brightness peaks in the UFO and CAHA light curve are exactly coincident, to within the error in time resulting from the frame rate of the sensors.

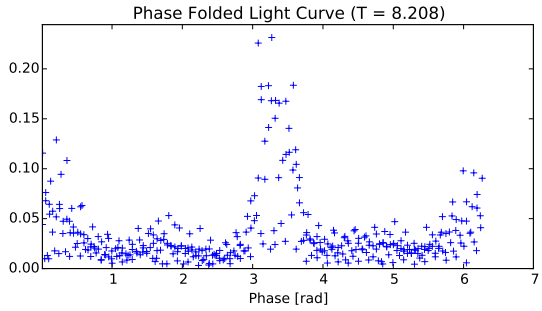
On consideration though, we can see that this has to be the case: The Observer-Satellite-



(a) Raw stereoscopic light curves UFO (blue) and CAHA (red)



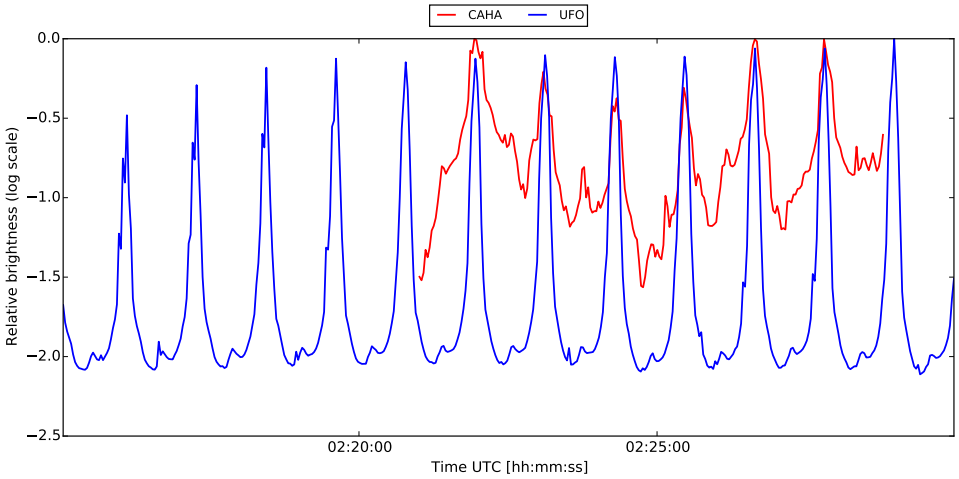
(b) CAHA phase folded light curve



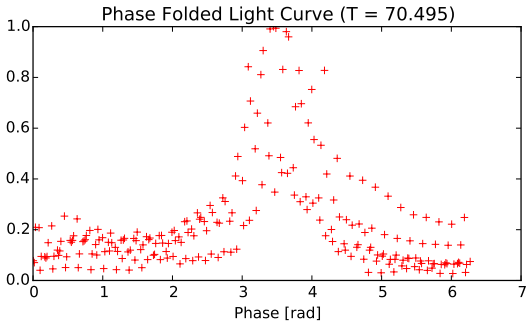
(c) UFO phase folded light curve

Figure 2-11: Stereoscopic light curve of Cosmos 1988 on 2017-06-01.

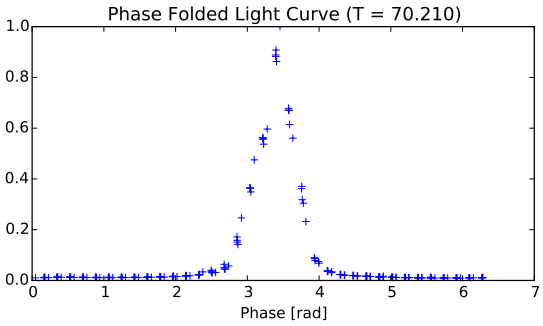
Observer (OSO) angle during this observation ranges from between 4.71 and 4.65 degrees (see fig. 2-13). Knowing that the satellite rotates once per 70.2 seconds - the time that the satellite takes to rotate through the OSO angle is approximately 0.9 seconds - lower than the frame rate of both UFO and CAHA.



(a) Raw stereoscopic light curves UFO (blue) and CAHA (red)



(b) CAHA phase folded light curve



(c) UFO phase folded light curve

Figure 2-12: Stereoscopic light curve of Cosmos 2362 on 2017-07-19.

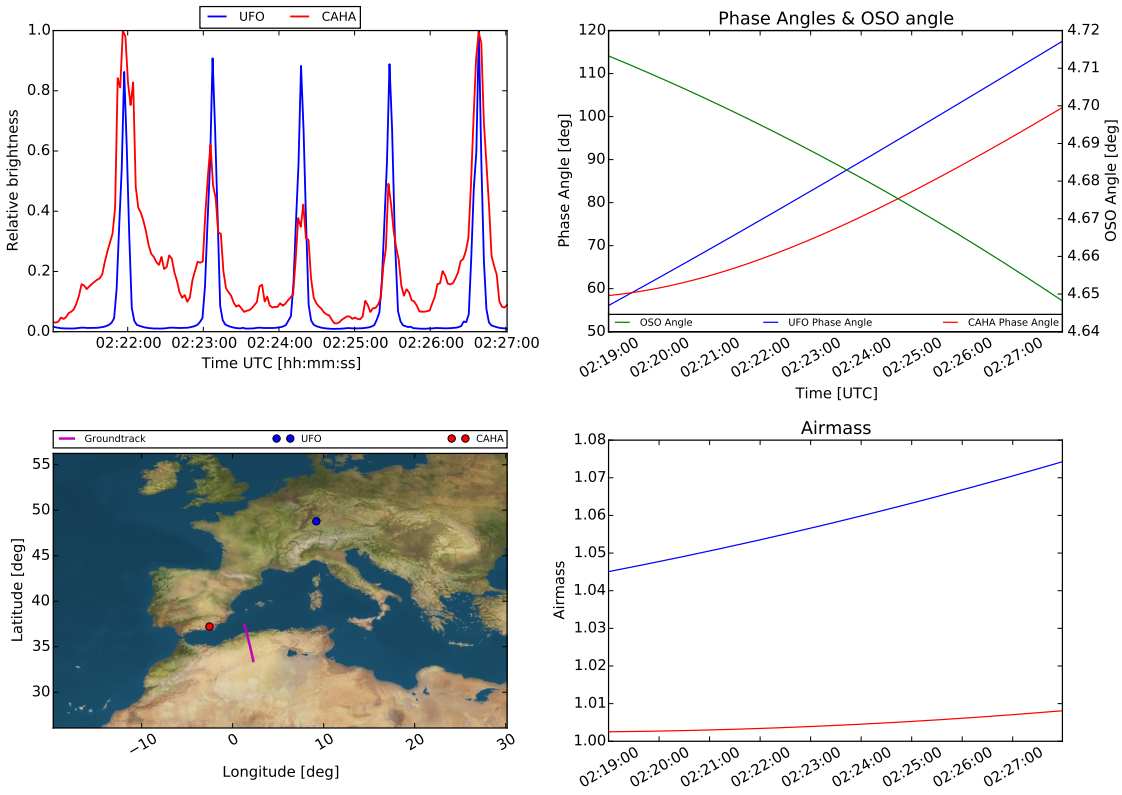


Figure 2-13: Analysis of the COSMOS 2362 joint observation produced with Raxus Prime 2.0

3. Light Curve Analysis

3.1. Frequency Analysis

Many light curves have characteristic shapes which are periodic in time. In an effort to understand what these repeating shapes might reveal about the rotation of the satellite, various methods of frequency analysis can be performed.

3.1.1. Fourier Transform

The discrete Fourier transform (DFT) transforms the discretely sampled time domain light curve into a power spectrum of constituent frequency components. The frequency resolution, df of the resulting series is given as the sample rate of the light curve and the total number of samples, N:

$$df = \frac{f_s}{N} \tag{3-1}$$

If the number of samples is increased, dt will become smaller. N can be artificially increased by padding the light curve with zeros. Although this does not improve the ability of the DFT to pick out extra frequency components in the signal, it does have an interpolating effect between the resulting frequency components. This can allow more precise determination of the maxima in the frequency spectrum.

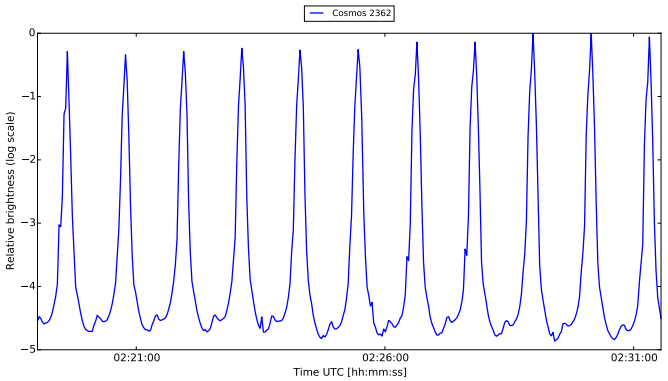
The minimum recoverable frequency is df and the maximum frequency which it is possible to recover, f_{max} - also known as the Nyquist frequency - is determined by the sample rate of the light curve dt:

$$f_{max} = \frac{1}{2dt} \tag{3-2}$$

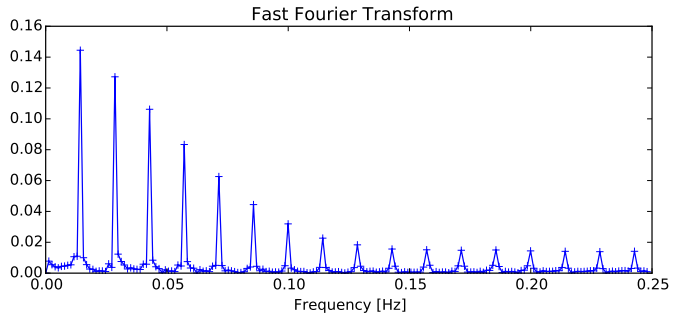
Light curves with evenly spaced sharp spikes approximate a Dirac comb. The DFT of a Dirac comb with spikes at kT, where k is an integer, is itself a Dirac comb, but with the spikes occurring at integer $\frac{k}{T}$ for all integers k. Care must be taken in this case to not mistake these harmonics as frequency components in their own right.

The DFT requires the sample rate to be constant and there to be no gaps in the data. Very often this is not the case for light curves.

Figure 3-2 shows the result of the DFT for the light curve of Cosmos 1988, obtained at CAHA. The DFT shows a strong maximum corresponding to a period of 4.104 seconds. A strong peak also exists at 8.20 seconds which is double this frequency. Also present are the repeating peaks discussed previously. Peaks at a period of $3/8.204$ and $4/8.20$ are present. A peak at a period of 9.76 seconds is the result of beating between the two, previously mentioned, higher frequency components. A peak also exists at a period of 51.64 seconds which is yet to be explained.



(a) Measured light curve



(b) FFT of the above light curve

Figure 3-1: Repeating sharp spikes cause strong harmonics to appear in the FFT

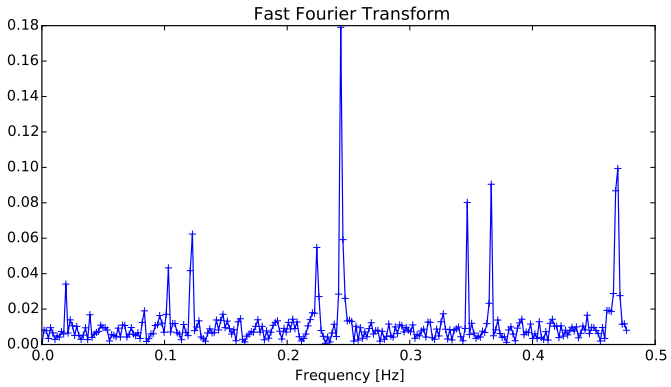


Figure 3-2: DFT Power spectrum for COSMOS 1988

3.1.2. Least-Squares Spectral Analysis

Where the light curve has a non-constant sample rate or gaps, the Lomb-Scargle Periodogram can be used to produce the power spectrum.

This approach estimates the power spectrum by performing a least squared fit of sinusoids to the data.

The power spectrum which results from Least squares spectral analysis (LSSA) of the light curve of COSMOS 1988 obtained at CAHA is shown in fig. 3-3. This spectrum shows good agreement with the Fourier transform but is to be trusted more since the sample rate of the light curve is not perfectly constant. It too shows the strongest frequency component to have a period of 4.10 seconds.

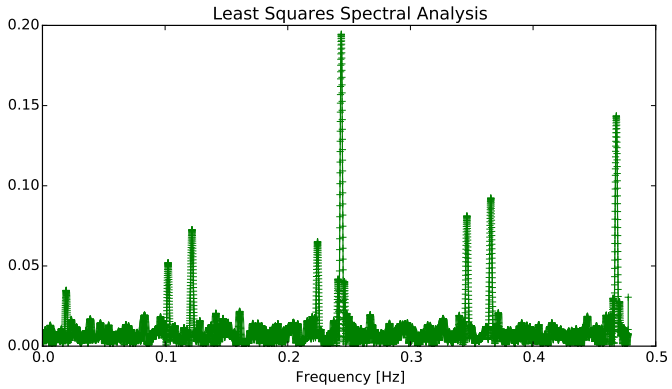


Figure 3-3: LSSA power spectrum of COSMOS 1988

3.1.3. Phase Dispersion Minimisation

Phase dispersion minimisation (PDM) is a technique which is popular in the astronomy community. The technique was developed to determine periods in light curves with non-sinusoidal structures, which the Fourier transform and Lomb-Scargle spectral analysis do not handle well [26].

This algorithm performs a search over a range of possible periods. It divides the full light curve up into sections with the length of this test period. These sections are layered on top of each other and the resulting distribution is subdivided up into a series of bins. The variance of the data in each of these bins is calculated and the overall variance of these bins is then used to produce a value which varies between 1 (for a poor fit) and close to 0 if the period is a good fit.

The PDM 'periodogram' for the light curve of Cosmos 1988 is shown in fig. 3-4. Unlike the spectra produced by the DFT and LSSA, this method produces a global minimum at 8.208 s. There is also a minimum at half this value, which corresponds to the strong component seen in the previous techniques, but here it represents a less good fit. Other harmonics of these frequencies are also visible.

3.1.4. Phase Folding

In the same way that Phase dispersion minimisation 'folds' the light curve on itself for each test period, the light curve can be manually folded so that each data point is plotted against its phase in this trial period.

Phase folding is technique which can be carried out on the light curve to help investigate the physical meaning of an obtained frequency component. In this way, phase folding can

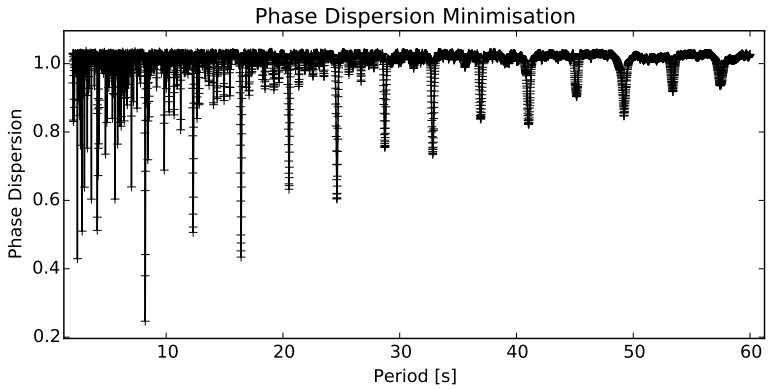
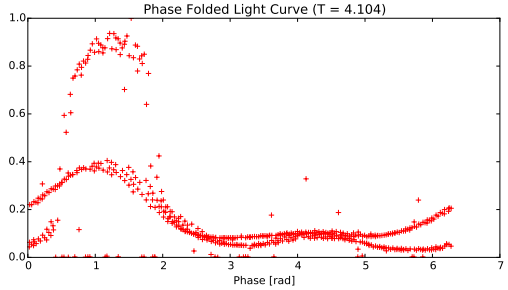


Figure 3-4: Phase dispersion minimisation periodogram of COSMOS 1988

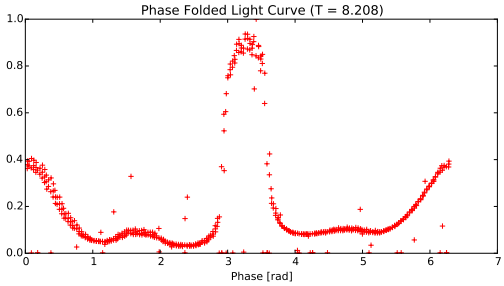
provide a 'sanity check', to ensure that the frequency components obtained from the above methods really correspond to features of the light curve. It is important to do this because the strongest frequency component is not always the frequency at which the light curve repeats.

The data from COSMOS 1988 is a good example of this. Without further analysis of the data it might be assumed that the target has a rotation period of 4.10 seconds. If the light curve is folded assuming this value it can be clearly seen that this really represents half of the rotation period (see fig. 3-5a). Plotting the folded light curve for a period of 8.208 seconds produces a much better result.

This is also an invaluable tool for another reason; By folding the light curve a kind of 'reconstruction' is formed, which reveals the underlying pattern in light curves which have large gaps or when the sample rate is low compared to the frequency of rotation. The original light curve, shown in fig. 2-11a appears to be a jagged, pulsing wave, one might make the assumption that this 'beating' in the signal was a feature of the rotation of the object. When the light curve is folded correctly, as in fig. 3-5b this behaviour disappears and it becomes clear that the beating is the result of the difference in frequency between the satellites rotation and the observer's frame rate.



(a) Light curve of Cosmos 1988, folded with a period of 4.104s



(b) Light curve of Cosmos 1988, folded with a period of 8.208s

Figure 3-5

3.1.5. Sidereal/Synodic Frequency

The spin rate of the target will be different depending on the frame of reference from which it is measured.

Synodic frequency is the rate of rotation with respect to the observer, who's frame of reference might also be moving. The frequency of rotation measured in a light curve will always be the synodic frequency.

Sidereal frequency describes the rate of rotation in inertial space, i.e. with respect to background stars. In a sense this is the 'true' rate of rotation.

It was hoped that, through differences in the measured synodic frequency of stereoscopic light curves, it would be possible to determine the relationship between synodic and sidereal frequency. This is rather non-trivial for the case of space debris where there are multiple moving reference frames and complex rotation around an unknown axis. Somewhat surprisingly the measured synodic frequency of the two observers are in perfect agreement for the case of the two successfully obtained objects.

It is likely that this is due to the fact that MEO objects were being observed. The change in position of the target, relative to the observers over the relatively short period of data acquisition was likely too small for these variations to be noticeable.

3.2. Analysis of Specular Reflections

Specular reflections are bright reflections which are due to mirror-like reflection from a surface. Perfect specular reflectors will reflect light exactly along the path determined by the mathematical reflection of the incoming light around the normal vector of the surface. Figure 3-6 shows a BRDF which approximates the properties of a solar panel and its cover glass. As can be seen, there is a large lobe in the BRDF centred around the reflection vector - which is determined by the incoming light vector and surface's normal vector. In reality this lobe would be much narrower but the roughness of the material has been increased to exaggerate this effect, for the purpose of demonstration.

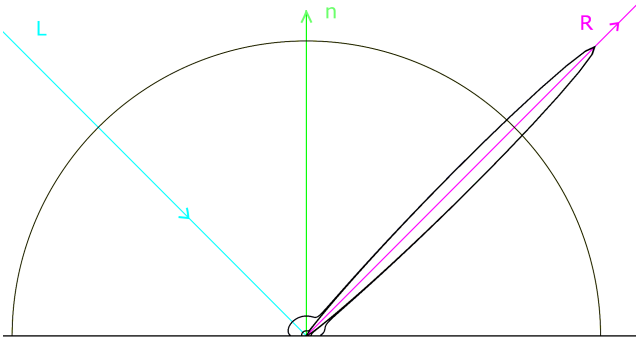
Specular reflections can sometimes be seen in measured light curves - they appear as sudden sharp spikes in the brightness. When this occurs, the observer can know that they fall in the specular reflection cone of the satellite. If the position of the satellite, position of the observer and time that a specular reflection occurred is known, then the incoming light vector $\mathbf{L}$ and reflected light vector $\mathbf{R}$ can be calculated.

The surface normal can then be estimated to within the uncertainty of the angular width of the specular cone of the BRDF. This is done by calculating the half angle vector between $\mathbf{L}$ and $\mathbf{R}$. The solution will lie in the phase angle plane. If the two vectors are normalised beforehand then this is as simple as calculating the normalised difference between the two vectors:

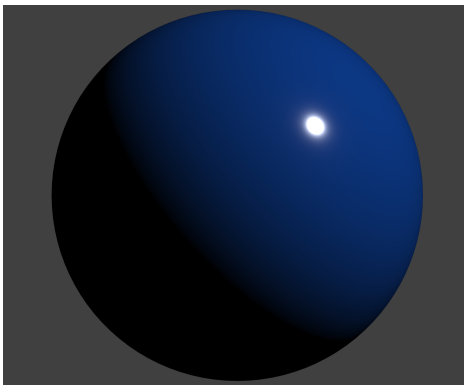
$$\mathbf{n} = \frac{\mathbf{R} - \mathbf{L}}{||\mathbf{R} - \mathbf{L}||} \tag{3-3}$$

If the shape of the satellite is known then the surface normals of any large, flat smooth surfaces are also known in the body frame of the satellite. It is hypothesised that the attitude of the satellite can be constrained by aligning this vector to the calculated normal (half-angle vector) in the inertial frame - done using the Rodruigues' rotation formula (see section 7.1.4). This doesn't fully determine the attitude because all rotations around the surface normal vector are valid solutions.

Uncertainties in this calculation also arise from the fact that the sun is not a point source at infinity and, in fact has an apparent angular diameter of around 32 arminutes as observed from Earth. Another uncertainty results from the fact that solar arrays consist of smaller cells which will not necessarily be perfectly aligned - this effect can be seen in the measured BRDF [13].



(a) Polar plot of a cross-section of the BRDF of a material approximating solar panel and cover glass. A logarithmic scale is used.



(b) The BRDF applied to a sphere

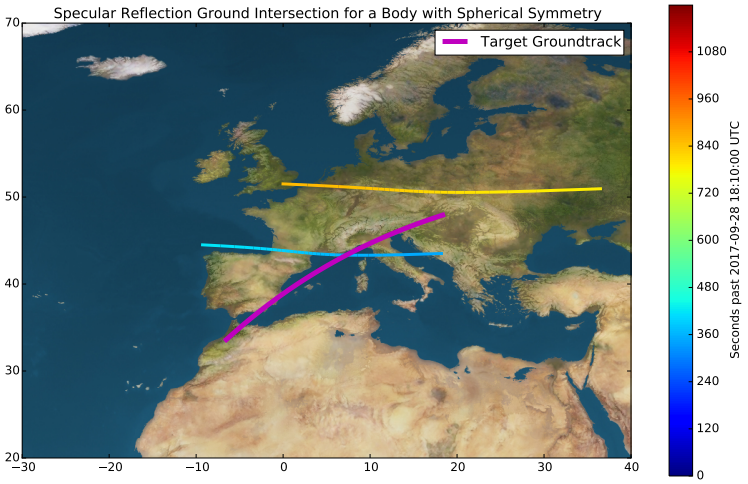
Figure 3-6

It was hoped that in stereoscopic light curve observations the same specular reflection might be seen in the light curves of the two observers. The time offset between the specular reflection in the two light curves would indicate the rotation axis of the target. No such event was seen in the two successful stereoscopic light curves.

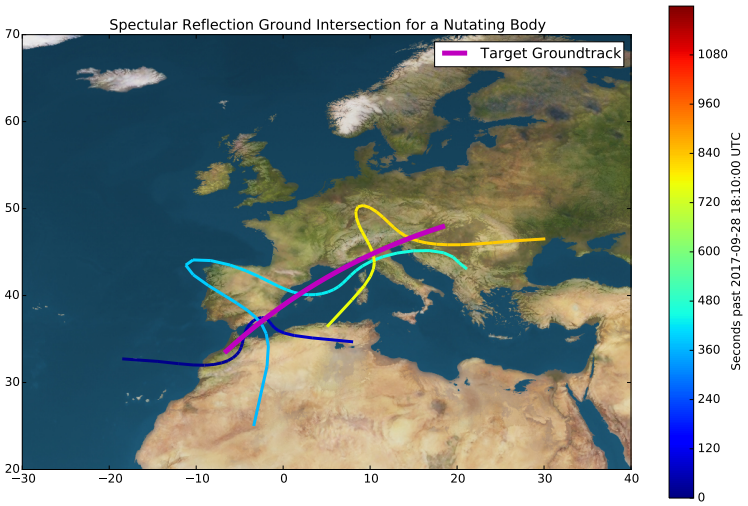
In Raxus Prime 2.0, the development of which is described in chapter 5, simulations of a body with a large specularly reflecting surface, were run to determine the path that the specular reflections would trace out on the ground. The results of two simulations of this hypothetical object, sharing the same LEO orbit as the ISS, are shown in fig. 3-7.

Figure 3-7a shows the ground intersection of a specular reflection from an object with a spherically symmetric inertia tensor which is in a spin state around the $+\hat{z}$ ECI axis.

Figure 3-7b shows a satellite with an axi-symmetric inertia tensor, rotating about an axis very slightly offset from its major principle axis of inertia. The small nutation which occurs is amplified greatly in the path of the specular reflection. It would not be possible to obtain a complete description of the rotation state based on a specular reflection being seen by only two observers. The complicated motion could not be uniquely determined given only an attitude at two epochs.



(a) Specular reflection ground intersection for a satellite with a spherically symmetric inertia tensor



(b) Specular reflection ground intersection for a nutating oblate axisymmtrix satellite

Figure 3-7: Two simulations showing the path of intersection of specular reflections from a rotating object in LEO.

4. Forward Modelling of Light Curves

4.1. Orbit Propagation

The position of a satellite at any given time can be estimated using the Simplified General Perturbations #4 (SGP4) orbit propagation model. This model considers forces arising from the J2 and higher order harmonics of the Earth's gravitational field, atmospheric drag, solar radiation pressure and the gravitational influence of the Sun and the Moon.

The model takes, as input, the Two-Line-Element set of the satellite and a time, and estimates the cartesian coordinates of the satellite for that time in Earth-centred inertial space.

Both the TLEs and the model introduce inaccuracies into the simulaton and this worsens as the time from the reference Epoch of the TLE increases. As mentioned previously, the TLE of a target is recorded when the real observation takes place and it is important to use this TLE when attempting to simulate a past observation.

Both the TLEs and the SGP4 model are made available to the public online by the United States Department of Defence through space-track.org [27].

4.2. Rigid Body Simulation

The satellite is modelled as a rigid body with inertia tensor $\mathbf{J}$ (to distinguish it from the identity matrix which is represented by the symbol $\mathbf{I}$), centre of mass c_m. The body has an angular velocity ω. When combined with the inertia matrix, the defines its angular momentum, $\mathbf{H}$, as shown in eq. (4-1).

$$\mathbf{H} = \mathbf{J}\omega \tag{4-1}$$

The body's rotational kinetic energy, E_k is obtained as shown in eq. (4-2).

$$E = \frac{1}{2}\mathbf{J}\omega^2 \tag{4-2}$$

Finally an attitude matrix $\mathbf{A}$ rotates the representation of a vector from a rest frame to the rotating body frame.

The satellite's rotation is modelled using the torque free Euler equations for rigid body rotation (eqs. (4-3) to (4-5)).

The real satellite will, in reality not be a truly rigid body and will not have perfectly torque free motion. Several forces will actually be acting on it:

- Gravity gradient
- Solar radiation pressure
- Aerodynamic forces
- The sloshing of any fuel which may be on board

- Momentum transfer through flexible appendages / moving parts

Some simulations attempt to account for these forces, for example, a software called ιOTA has been developed as part of ESA's *Debris Attitude Motion Measurements and Modelling* project, by Hyperschall Technologie Göttingen GmbH, specifically for accurate space debris attitude simulation. This software accounts for all of the aforementioned forces [28].

However, since the periods of observation are relatively short in this thesis, the assumption is made that an insignificant amount of momentum transfer will take place over the duration of an observation and torque-free case is used.

$$\dot{\omega_1}(t) = \frac{J_2 - J_3}{J_1}\,\omega_2(t)\,\omega_3(t) \tag{4-3}$$

$$\dot{\omega_2}(t) = \frac{J_3 - J_1}{J_2}\,\omega_3(t)\,\omega_1(t) \tag{4-4}$$

$$\dot{\omega_3}(t) = \frac{J_1 - J_2}{J_3}\,\omega_1(t)\,\omega_2(t) \tag{4-5}$$

The goal of the rigid body simulation is to find a matrix, $\mathbf{P}(t)$ which will transform from an initial attitude matrix $\mathbf{A}_0$ to an attitude matrix at time t.

$$\mathbf{A}(t) = \mathbf{P}(t)\mathbf{A}_0 \tag{4-6}$$

4.2.1. Principal Axes of Inertia

The body frame of a satellite can be defined arbitrarily - the 3D model will not necessarily be aligned with the principal moments of inertia.

The moment of inertia of the model is determined in the CAD software assuming a uniform density solid body.

For an inertia tensor in an arbitrary body frame $\mathbf{J}_{body}$ the principal axes of inertia in the body frame are the eigenvectors $\mathbf{e}_{1,2,3}$ of the inertia tensor.

The principal moments of inertia, $J_{1,2,3}$ are the eigenvalues of $\mathbf{J}_{body}$:

We obtain a rotation matrix from body to principal axis frame by using the the eigenvectors as columns as below:

$$\mathbf{A}_{BP} = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{4-7}$$

In this new frame the inertia tensor is constructed by placing the eigenvalues along the diagonal:

$$\mathbf{J}_P = \begin{bmatrix} J_1 & 0 & 0 \\ 0 & J_2 & 0 \\ 0 & 0 & J_3 \end{bmatrix} \tag{4-8}$$

It is in this 'Principal Axis Frame' that the Euler equations must be solved.

4.2.2. Spherically Symmetric Body

In the case that $J_1 = J_2 = J_3$ then the body is said to be spherically symmetric.

Substituting this relationship into eqs. (4-3) to (4-5), yields the following result:

$$\dot{\omega_1}(t) \;=\; \dot{\omega_2}(t) \;=\; \dot{\omega_3}(t) \;=\; 0 \tag{4-9}$$

Meaning that the angular velocity vector is constant in both the rest and body frame. The attitude of the body can therefore be represented by a constant rate of rotation about a fixed axis:

$$\mathbf{P}(t) \;=\; \mathbf{R}\left(\frac{\omega}{\|\omega\|},\; -\|\omega\| t\right) \tag{4-10}$$

where $\mathbf{R}\,(\mathbf{e}, \theta)$ denotes the Rodrigues' formula for the determination of a rotation matrix describing a rotation around an axis $\mathbf{e}$, by an angle θ (see Appendix B - Axis/Angle Rotation).

An example of this type of rotation is shown in fig. 4-1

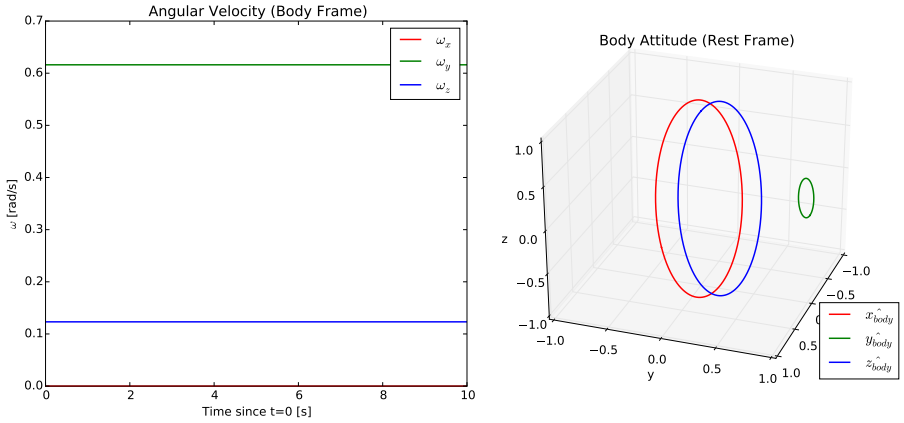


Figure 4-1: Angular velocity and attitude of a spherically symmetric body

4.2.3. Cylindrically Symmetric Body

In the case where only two of the principal moments of inertia are equal, the body is said to be cylindrically symmetrical (or axisymmetric). If we let $J_1 = J_2$ then the torque-free Euler equations, eqs. (4-3) to (4-5), become:

$$\dot{\omega_1}(t) = \omega_p\, \omega_2(t) \tag{4-11}$$

$$\dot{\omega_2}(t) = -\omega_p\, \omega_1(t) \tag{4-12}$$

$$\dot{\omega_3}(t) = 0 \tag{4-13}$$

Where ω_p is the precession frequency of the body, given by:

$$\omega_p = \left(1 - \frac{J_3}{J_1}\right) \omega_3(0) \tag{4-14}$$

The rotation of the body is described by a rotation around the axis of symmetry of ω_p and a rotation around the angular momentum vector with a frequency of $\Omega = \frac{\|\mathbf{H}\|}{J_1}$ Thus the rotation from initial attitude to attitude at time t (in the principle axis frame) is given by:

$$\mathbf{P}(t) \; = \; \mathbf{R}(\mathbf{e_3}, -\omega_p t)\, \mathbf{R}\left(\frac{\mathbf{H}}{\|\mathbf{H}\|}, -\Omega t\right) \tag{4-15}$$

As an example, an oblate cylinder which has moments of inertia $J_x = 1$, $J_y = 1$, $J_z = 10$ kgm^2 is considered. Figure 4-2a shows the resulting angular velocities and attitude when the body is made to rotate at a rate of $120° s^{-1}$ around an axis which is close to one of its minor principal axes of inertia ($[1, 0, 0.3]$). Figure 4-2b shows the case where the rotation axis is close to the axis of symmetry ($[0.3, 0, 1]$).

4.2.4. Asymmetric Body

A body which has three different principal moments of inertia is said to be asymmetric (or triaxially symmetric in some sources). Unlike the previous two cases, none of the angular velocities will necessarily be constant in time.

An optimised algorithm for producing the attitude matrix of a rotating asymmetric body, which is numerically accurate to machine precision, was published in the Journal for Computational Physics in 2007 [29]. This algorithm is used for the asymmetric rotation propagation in this thesis.

As a test object, a cuboid with dimensions 1×4×9 is considered. The body is assumed to have perfectly uniform density. The moments of inertia of such an object will have the ratios of $J_x : J_y : J_z = 17 : 97 : 82$.

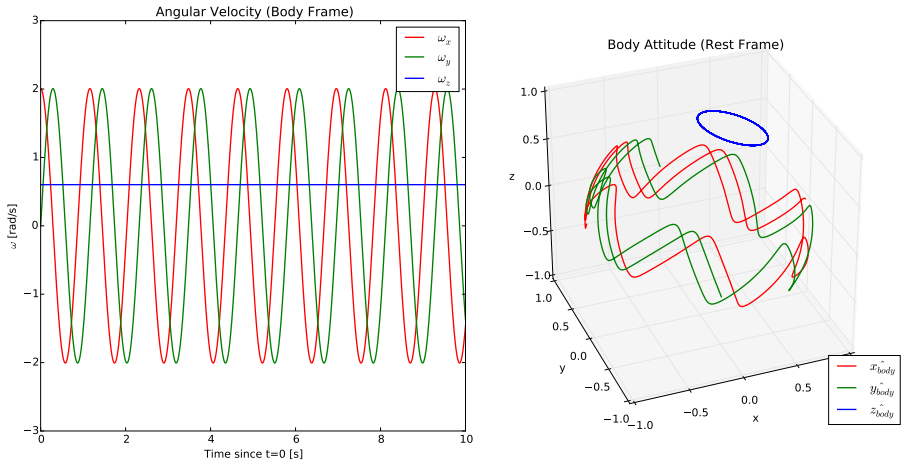
The body is given an initial rotation around an axis which is offset by a very small amount from the intermediate axis of inertia, $[0, 1.0, 0.00001]$

4.2.5. Stable / Unstable Rotation

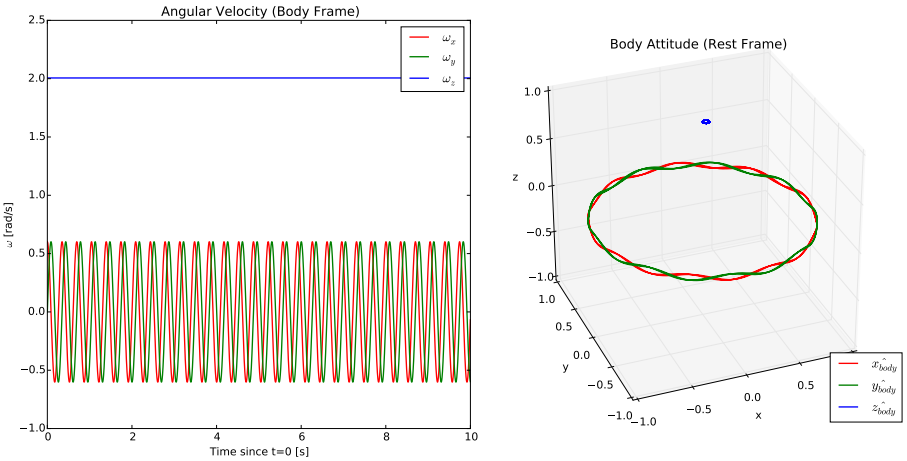
There is some inconsistency in the literature regarding the use of the terms stable and unstable when it comes to the rotation of space debris. Some papers refer to satellites which are no longer operational (actively stabilised) as unstable, even if they rotate around a principle axis. Other papers describe any body with significant nutation as unstable.

In the rigid-body sense of the word, however, stability refers to a rotation state which is not perturbed greatly by small torques. As an example, an asymmetric body rotating around its intermediate axis is in an unstable rotation state - any small deviation from this axis will result in growing nutation and a flipping motion.

In this thesis, the term stability refers to the rigid body sense of the word. Rotation states which have large nutation or the above-mentioned intermediate axis rotation are referred to



(a) Angular velocity and attitude of an oblate cylinder rotating close to its minor principal axis of inertia



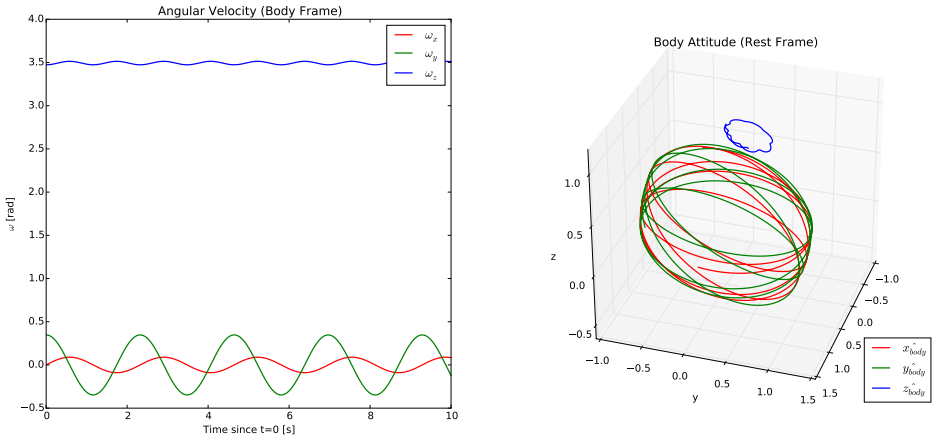
(b) Angular velocity and attitude of an oblate cylinder rotating close to its axis of symmetry

Figure 4-2

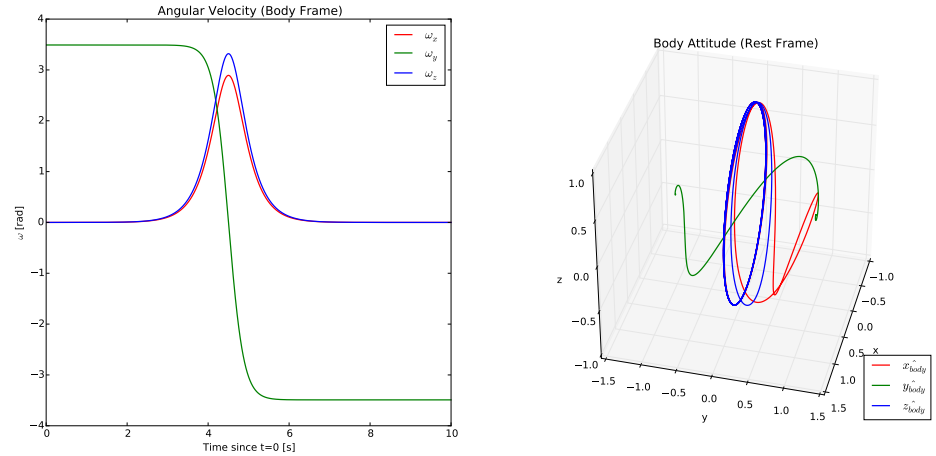
as tumbling.

Any body which is rotating exactly around one of its principal axes of inertia will have rotation which does not nutate, but this is not necessarily stable.

In a body where momentum exchange can occur, the rotation will tend towards the state of lowest kinetic energy - rotation around the major principle axis [30]. This state is referred to as 'flat spin'. Momentum transfer can occur through sloshing of liquid fuel, the free rotation of reaction wheels, the flexing of booms and solar panel arrays. Many space debris objects will posses one of these features and so the assumption of flat spin may be reasonable, or at least a good starting point when attempting parameter estimation.



(a) Angular velocity and attitude of an asymmetric body with rotation close to the major axis of inertia



(b) Angular velocity and attitude of an asymmetric body with rotation very close to the intermediate axis of inertia

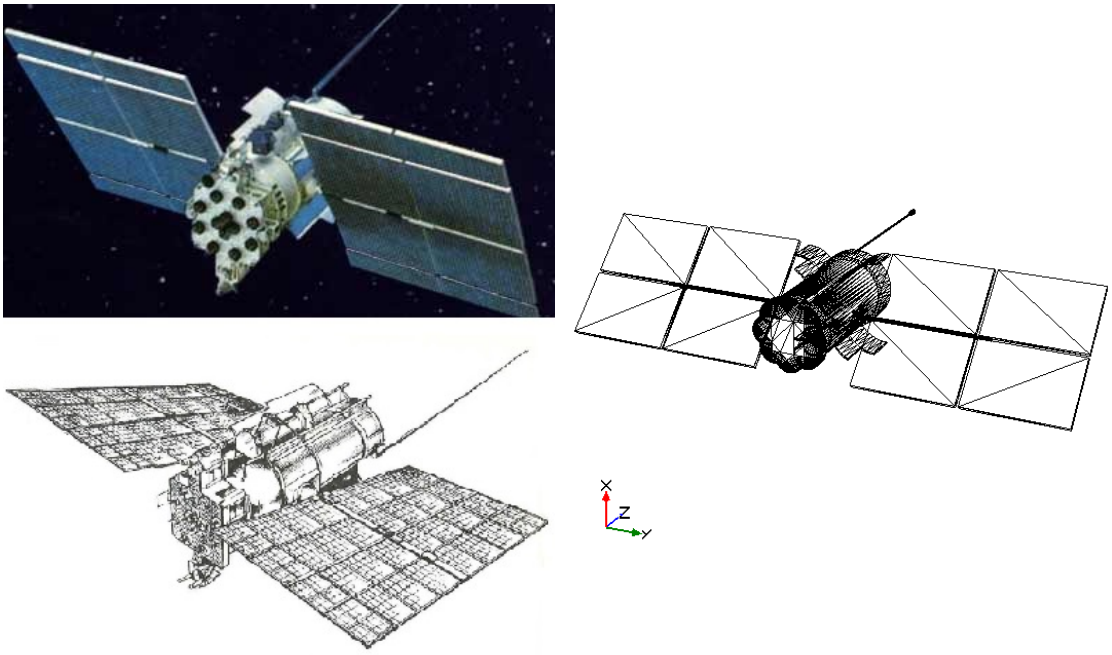
Figure 4-3

4.3. 3D Model

The most important factors that determine the shape of a light curve are the shape of the object and the materials from which it is constructed. For some satellites detailed information on the dimensions and materials used are readily obtainable. For others, such as the GLONASS objects which were observed, this information is not in the public domain. To create the 3D model for these satellites, reference images and information have to be pieced together from a number of less reliable sources.

Both Cosmos 1988 and Cosmos 2362 are Uragan 11F654 models. The reference images shown in fig. 4-4a were obtained from a magazine article from 1993 [31] and a space enthusiast's blog [32]. Using GLONASS satellites as targets presents a large problem. These box-wing type satellites have variable pitch solar panels for tracking the sun.

In the case of a known target, photographs and technical drawings of the satellite can be used to create a 3D model which approximates the satellite's shape.



(a) Reference images of a first generation Glonass satellite [31][32].

(b) Wireframe of the GLONASS model

Figure 4-4

4.4. Scene Rendering

Once the geometry of the scene has been determined a description of the scene can be passed to a rendering engine to simulate the reflection of light from the target.

In this thesis rendering was performed in two ways - through ray tracing and through the more traditional rasterization and fragment shading pipeline implemented by OpenGL.

4.4.1. Ray Tracing

Raxus Prime 1.0 used the open-source ray tracing engine POV-Ray [19] for rendering the images which were used for generating synthetic light curves. This technique proved itself capable of producing light curves which were comparable to real light curves and was kept and improved on in Raxus Prime 2.0.

As opposed to traditional rasterization techniques, ray tracing works by modelling the path of light through a scene.

To improve computational efficiency, the rays of light are usually shot from a virtual camera, through a virtual imaging plane and into the scene. A number of these primary rays can be

shot to achieve a higher sample rate-per-pixel This can be done a number of times to achieve a more realistic value for the resulting pixel value. In this thesis a supersampling of 4 was used, meaning that a grid of 16 rays are cast per pixel. This helps to overcome aliasing effects.

If the cast ray intersects a surface in the scene then the BRDF of the surface at this point is evaluated. Secondary rays are cast from this surface to check for reflections of light from other surfaces and shadowing effects. Tertiary rays can be cast from the intersection points of these secondary rays and so on. In this way, Ray tracing can model a number of complex phenomena, such as refraction, multiple reflections and self-shadowing.

Materials in POV-Ray are described by specular, diffuse, ambient, roughness and metallic parameters - values of these for some common material are provided with the POV-Ray distribution. The option to have POV-Ray conserve energy during reflection can also be set in the material properties. In this work it is important to make sure that this keyword is set to keep the rendering as physically based as possible.

Although more physically accurate than other methods, Ray Tracing can be very slow. POV-Ray carries out the ray tracing computation on the CPU.

4.4.2. OpenGL

OpenGL is a widely used, cross-platform API for 2D and 3D computer graphics. OpenGL manages the low-level calls to the GPU drivers.

The default OpenGL pipeline breaks the model down into a number of ‘fragments’, related to the projection of the pixel area onto the surface. A reflection model is evaluated for each of these fragments to obtain a colour for the corresponding pixel.

By default OpenGL uses a Blinn-Phong shading model. This is a model based on the Phong model but which trades physical accuracy for computational efficiency. The Blinn-Phong actually performs interpolated flat (Gouraud) shading.

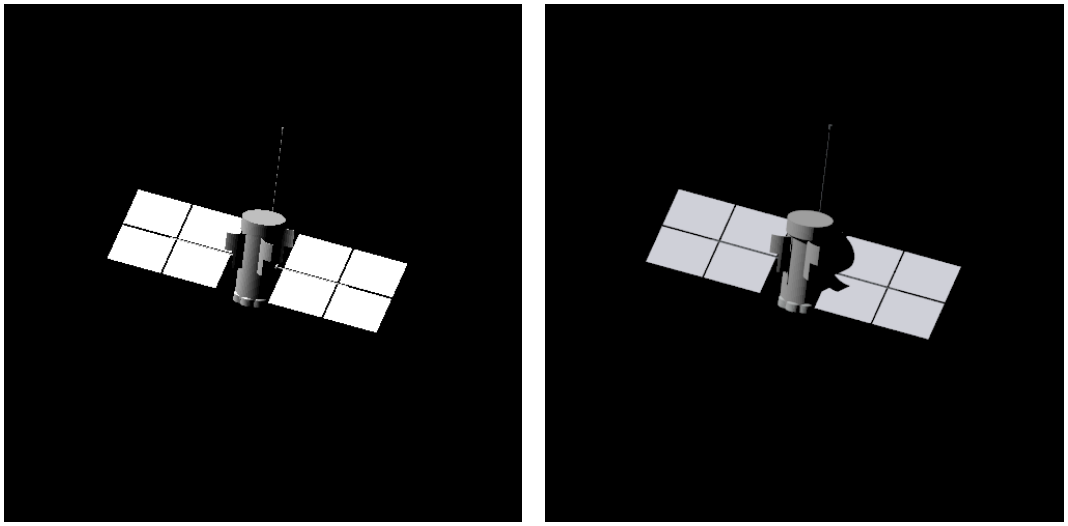
Materials in this reflection model are defined by 4 parameters: Three colors - Ambient, Diffuse, Specular and a 4th parameter - Shininess. By varying these four parameters it is possible to approximate the reflection properties of various materials. It is possible to implement more sophisticated reflection models but in this work the choice to implement an OpenGL rendering pipeline was made to minimise the computation time required, not to be as physically accurate as possible.

Although it is easy to model shadows which occur due to the facets facing away from the light source, it is non-trivial to model self-shadowing which occurs in concave bodies.

4.5. Corrections

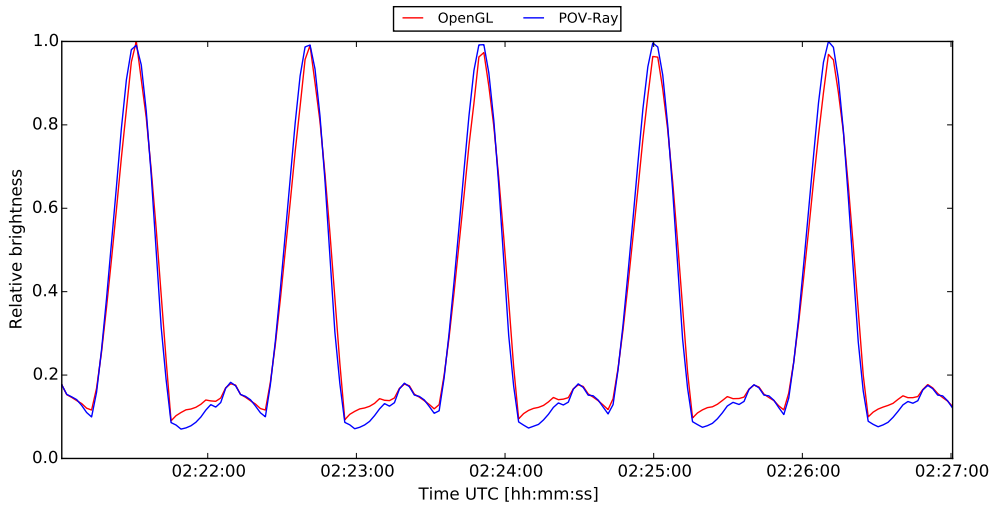
4.5.1. Gamma Correction

Gamma correction is a non-linear correction applied to brightness values in a digital/print image which stretches and compresses the brightness to make the image more pleasing to the human eye. The human eye is more sensitive to changes in brightness at low intensity,



(a) Image of a GLONASS satellite rendered using OpenGL

(b) Image of a GLONASS satellite rendered using Pov-Ray



(c) Comparison of the light curves produced using OpenGL (red) and POV-Ray (blue).

Figure 4-5: The same scene rendered using OpenGL and POV-Ray. Self-shadowing is noticeably missing on the solar panel of the satellite rendered with OpenGL. This effect can be seen in the light curves, in the regions where the OpenGL light curve is brighter than the ray-traced counterpart.

than the same change at high intensity. Put simply, the eye can more easily differentiate between black and 1% gray than 99% gray and white.

Since the range of possible pixel values in a digital image is discrete, an efficient use of these units is to assign dark areas more pixel values than light areas. When creating image files, many programs will apply a gamma compression to do this, according to the equation below:

$$I = A\, {I_0}^{\gamma} \tag{4-16}$$

Where γ is the encoding gamma, a compression factor usually around 0.45 and A is a constant.

Afterwards, on displaying the image, the computer or monitor will perform a gamma decoding which produces an image which looks correct on the particular viewing device.

This makes more efficient use of the pixel values from a aesthetic point of view but it means that the image file is no longer a linear representation of the original brightnesses. This is especially a problem for generating light curves where the values of all of the pixels in an image must be summed.

When producing synthetic light curves, it is more important that the results are physically accurate. To do this gamma must be fixed to 1 when the simulated images are rendered to avoid this compression. The plot below shows an example of a light curve when gamma correction has and has not been applied.

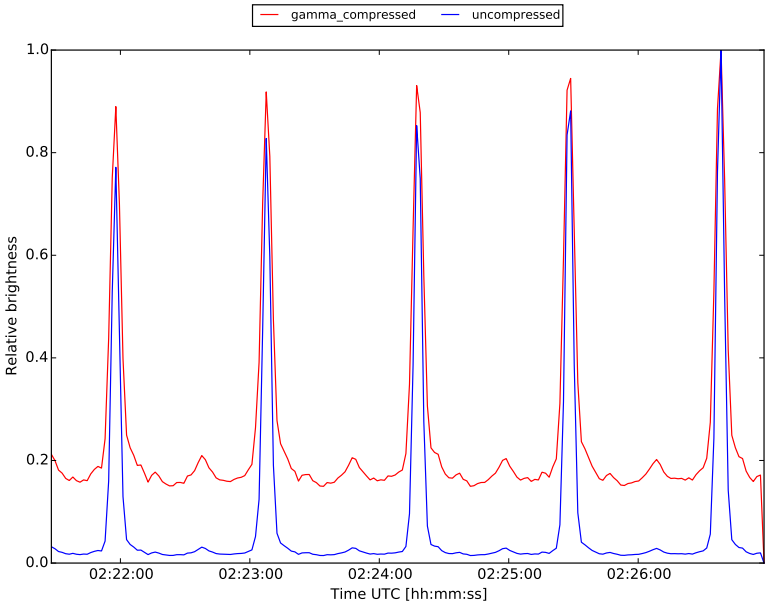


Figure 4-6: A simulated light curve before and after applying gamma compression (blue and red respectively).

4.5.2. Range Correction

As described in section 2.7.1, the inverse square law describes how the received light falls off with the square of the distance between the target and observer.

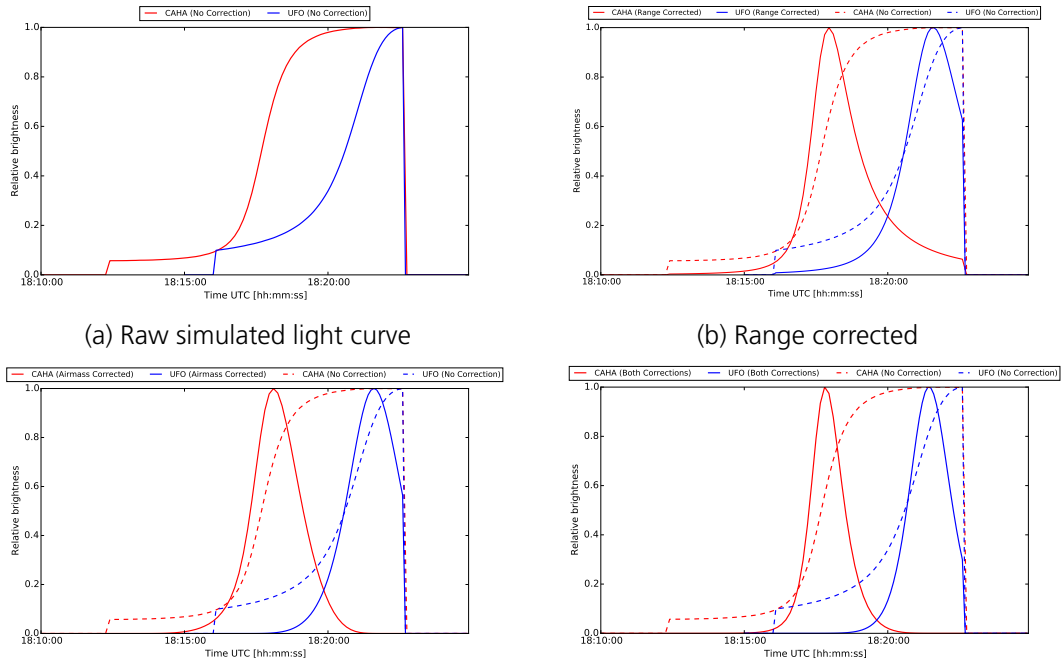
Rather than applying this correction to the real data, it is also possible to simulate this effect on the synthetic data.

Since the positions of the target and observer are calculated for the simulation, the correction is simply a case of dividing the simulated brightness by the square of this distance.

$$I = \frac{I_0}{x^2} \tag{4-17}$$

4.5.3. Airmass Correction

The inverse of the airmass correction in section 2.7.2 can also be applied in simulation to mimic the effect of the Earth's atmosphere. Figure 4-7c



(a) Raw simulated light curve

(b) Range corrected

(c) airmass corrected

(d) Range and airmass corrected

Figure 4-7: A pass of a diffusely reflecting spherical object in LEO, with and without the various light curve corrections.

4.5.4. Motion Blur Correction

When a real light curve is produced the sensor essentially performs an integration of the reflected brightness over the exposure time. In contrast, the simulated images which are rendered by Raxus Prime are a representation of the brightness of a static scene (for an infinitesimal exposure time).

When the exposure time is relatively long compared to the angular velocity of the target this can present a large problem. Specular reflections from planar surfaces can be very bright and occur over a short amount of time. These brief flashes will be captured by the real sensor but may not be seen in the simulated image which only renders a particular instant.

To make more meaningful comparisons between synthetic and real light curves it is possible to emulate this effect in a crude way. It is possible to render a number of sub-frames, at times between the exposure start and end time. The brightness of these sub-frames are then combined to obtain a value which is more representative of a real exposure. Obviously it can

become very computationally expensive to render all of these extra frames.

Note that there is another type of motion blur present in the real images, which arises from inaccuracy in the telescope tracking and from atmospheric seeing. It is not important to simulate these effects since they do not change the total amount of light which is incident on the sensor significantly.

5. Development of Raxus Prime 2.0

5.1. Requirements

There is a need for a software in which light curves can be analysed and simulated for the purpose of developing an understanding of how each of the many variables at play contribute to the final shape of a light curve. This software would also be a step towards the ultimate goal of automatically being able to estimate these parameters.

As mentioned in section 1.3, Raxus Prime 1.0 is very successful in its ability to generate single observer simulated light curves.

As part of the description for this thesis project, two requirements were identified for improving Raxus Prime 1.0:

1. Support for multi-observer light curve simulation
2. Faster rendering than Raxus Prime 1.0 to allow for automatic parameter fitting in the future.

In order to meet these requirements in a sustainable and worthwhile way, the following requirements were also added and the decision was made to begin work on a completely new version of Raxus Prime, (v2.0).

3. Easy comparison of data
4. Robust and Logical Data Hierarchy
5. Maintainable and Expandable Code

5.2. Design

This section describes how each of the requirements for Raxus Prime 2.0 translated into design decisions.

5.2.1. Maintainable and Expandable Code

The codebase for Raxus Prime 1.0 consisted of 6758 lines of almost entirely uncommented code. The majority of the code was contained within a single .py file, with this file containing 5330 lines. A crucial requirement for the development of Raxus Prime 2.0 was that the code should be readable, maintainable, and as modular as possible to allow the software to be easily expanded in the future.

Raxus Prime 2.0 was written in Python 2.7 and uses the PyQt4 Python bindings for Qt, a widely used and comprehensive GUI framework [33][34]. The decision was made to port to PyQt4 from Tkinter (which is the framework used by Raxus Prime 1.0) to maximise the compatibility with the other software developed by the Space Debris group at DLR Stuttgart.

Another factor was that Qt was found to offer a more flexible and extensive framework which would likely make Raxus Prime easier to update and maintain in the future. The Qt framework lends itself to a modular style of development. A core concept in the Qt framework is the use of widgets. Effort was made as much as possible to keep the GUI logic separate from the underlying functionality of the program.

From a code readability standpoint, effort was made to adhere to the Python Enhancement Proposal 8 (PEP8) style guidelines [35] as strictly as possible.

Similarly, effort was made to comment all files, classes and methods with Google-Style Docstrings [36]. The reason for doing this is twofold; to adhere to a particular style of commenting to enhance readability of the source code and so that documentation for the project can automatically be compiled. In addition care was taken to write effective comments within the code, especially at points where a reference to a paper, or online resource could be provided.

5.2.2. Robust and Logical Data Hierarchy

It was decided that the data storage should take place in the form of a tree like hierarchy of information. This structure is shown in fig. 5-1. Each arrow represents a python dictionary structure which can contain multiple instances, indexed by name. Each of the labels represents a 'project element' - a class which contains stores relevant data and associated functions.

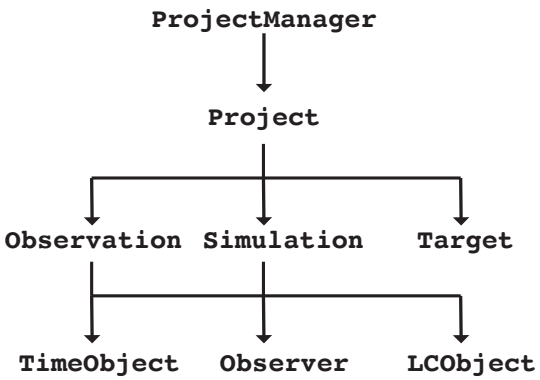


Figure 5-1: Hierarchical structure of project elements in Raxus Prime 2.0

Raxus Prime 2.0 contains only one instance of `ProjectManager`. This was done so that all of the data in the open projects can easily be made available to any class or function which needs it, simply by passing a reference to the `ProjectManager` object.

Currently each project may only contain one `Target` object - this was decided to provide some distinction between projects. This decision has had the unintended side-effect of making it difficult to compare the light curves of different objects side-by-side. In the future the structure should be adapted to allow multiple target objects. On reflection, this decision should have always been left to the user, rather than forced by the software. This is a change

which would have to be made before analysis and simulation of the light curves resulting from satellite proximity operations (as described in [37]) could be carried out.

As well as a robust data *storage* hierarchy in Raxus Prime 2.0 the *flow* of information through Raxus Prime 2.0 is structured in a logical manner. The flow of information through Raxus Prime 1.0 was very convoluted - the core logic of the program was contained within the classes of the Graphical User Interface. This made any expansion of the code very difficult.

In Raxus Prime 2.0 the core logic of the program is separate to the GUI. The Project elements described above act as an intermediate layer (see fig. 5-2)

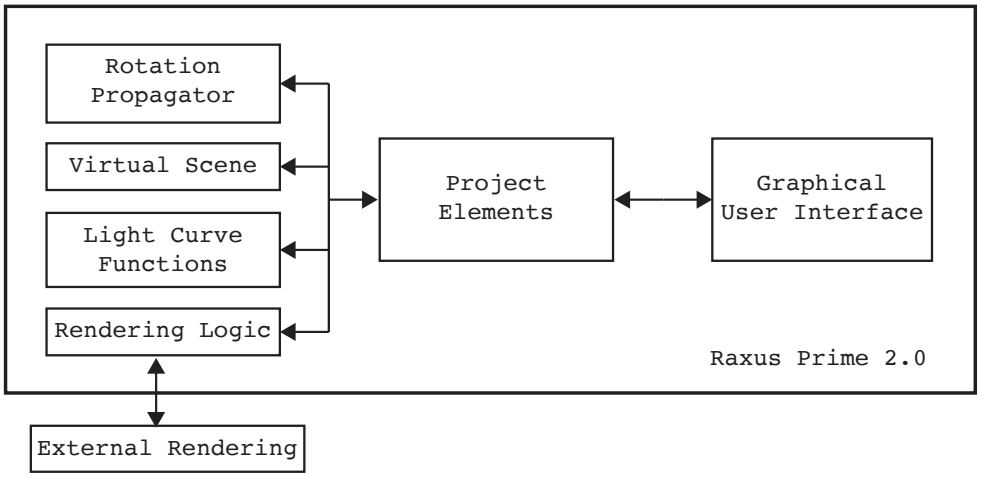


Figure 5-2: Simplified overview of the flow of information through Raxus Prime 2.0

5.2.3. Easy Comparison of Data

Raxus Prime 1.0 made use of a tab-like user interface where the GUI elements was displayed very openly but spread over a number of separate tabs which could not be viewed simultaneously. The program was divided up into a number of tabs which the used could select

The main window contained a toolbar and a single tab widget into which the whole functionality of the program was divided into 6 tabs. The benefit of this layout was that all of the elements and information was very openly displayed but a great disadvantage is that only one tab can viewed by the user at a time. The practical consequence of this was that the user could not view real and simulated light curves at the same time.

With Raxus Prime 2.0 the decision was made to base the GUI around a central MDI (multiple document interface) area where subwindows could be created, moved and displayed alongside each other. This approach lends itself not only to side-by-side comparison of data but because multiple instances of a certain widget can be instantiated simultaneously. Around this central area are regions where sidebars or toolbars can be docked / undocked. If more area is required in the MDI area to display subwindows next to each other, these widgets

can be resized, shown, hidden, and repositioned as required. In the default Raxus Prime 2.0 layout the sidebar to the right of the MDI area shows.

5.2.4. Support for Multi-Observer Light Curves

Support for multi-observer light curves emerges as a result of the chosen data hierarchy. A project can contain multiple observers, multiple simulations and, in theory, multiple targets although this is not fully implemented yet in practice.

Pre-sets for observer latitudes, longitude, and heights can be saved to a human-readable settings file so that the properties for commonly used sites do not have to be re-entered.

Because a reference to the `ProjectManager` object is passed to most of the tools for viewing and analysing data, the user can be presented with a list of all the available relevant data in the project.

To give the user a sense of continuity between tools, each observation and simulation has a colour assigned to it on creation. This colour is then used in any subsequent plots which use data from that object.

A feature which is currently being tested is the support for mobile observers - for example from SOFIA - a joint NASA/DLR airborne observatory which is mounted on a Boeing 747SP [38] or from a space-based observer, as discussed in section 7.1.4.

5.3. Implementation

Described below are a number of classes which are important in Raxus Prime 2.0 and their development is discussed.

5.3.1. `VirtualScene`

`VirtualScene` is the class which holds all of the functions for determining the geometry of the scene. Some of the most important features are listed below:

- Determining whether a target is visible to a particular observer.
- Calculating the baseline and Earth-central angle between two observers.
- Determining the position of the sun
- Calculating phase angles, observer-satellite-observer angles.
- Determining the half-angle vector corresponding to a specular reflection, seen by a particular observer.
- Calculating the airmass between an observer and target.

`VirtualScene` also contains many static functions for transforming to and from different coordinate frames.

There can be many instances of `VirtualScene` within a single `Project`. Each `Simulation` and `Observation` has its own instance of `VirtualScene`.

An open source python implementation of the 2010 version of the simplified generic pertubation (SGP4) model is used for orbit propagation.

The python package Astropy [39] is used for efficient and accurate handling of large batches of coordinate frame transformations, for finding the position of the sun and for unit conversions.

5.3.2. `RotationObject`

`RotationObject` is the class in Raxus Prime 2.0 which carries out the rigid body rotation simulation of the target. It is initialised with the inertia tensor, initial attitude of the target in the ECI frame, and the rotation axis which can be given in either the ECI, body or principal axis frame. Given this information, the `RotationObject` decides which one of the situations in section 4.2 applies according to the flow chart shown in fig. 5-3. The rotation object is then evaluated for a particular time, t, given as a `datetime` object, by calling `RotationObject.evolve(`t`)`. This returns the angular velocities in the body frame and the attitude matrix which describes the transformation from ECI to body at t.

5.3.3. `ProjectManager`

As introduced in section 5.2.2, `ProjectManager` Is the top-level data storage object. Raxus Prime 2.0 only contains one `ProjectManager` and this stores all of the open instances of `Project`.

`ProjectManager` is responsible for the creation, opening from a file, saving to a file and closing of `Project`s. All of the main data storage classes in Raxus Prime manage the lower-level classes using python dictionary structures. This means that everything is indexed by name. `ProjectManager` ensures that no problems arise from duplicated names or other indexing errors.

Raxus Prime 2.0 project files are saved with the file extension of '.rxp'. This is the same extension used as Raxus Prime 1.0 although there is no backward compatibility.

5.3.4. `Project`

`Project` is the container for all of the lower-level project elements and some information about the state of the GUI. `Project` itself does not have much functionality but it is this object with is binarised, saved and loaded if the user wants to store or share a Raxus Prime 2.0 project.

5.3.5. `ProjectTreeWidget`

`ProjectTreeWidget` is a GUI element which is usually docked to a sidebar in the main window of Raxus Prime 2.0. This element displays a tree structure representing the `Target`, `Simulation` and `Observation` objects which are contained within the open project. The user can use `ProjectTreeWidget` as a shortcut for editing these objects and for creating new ones.

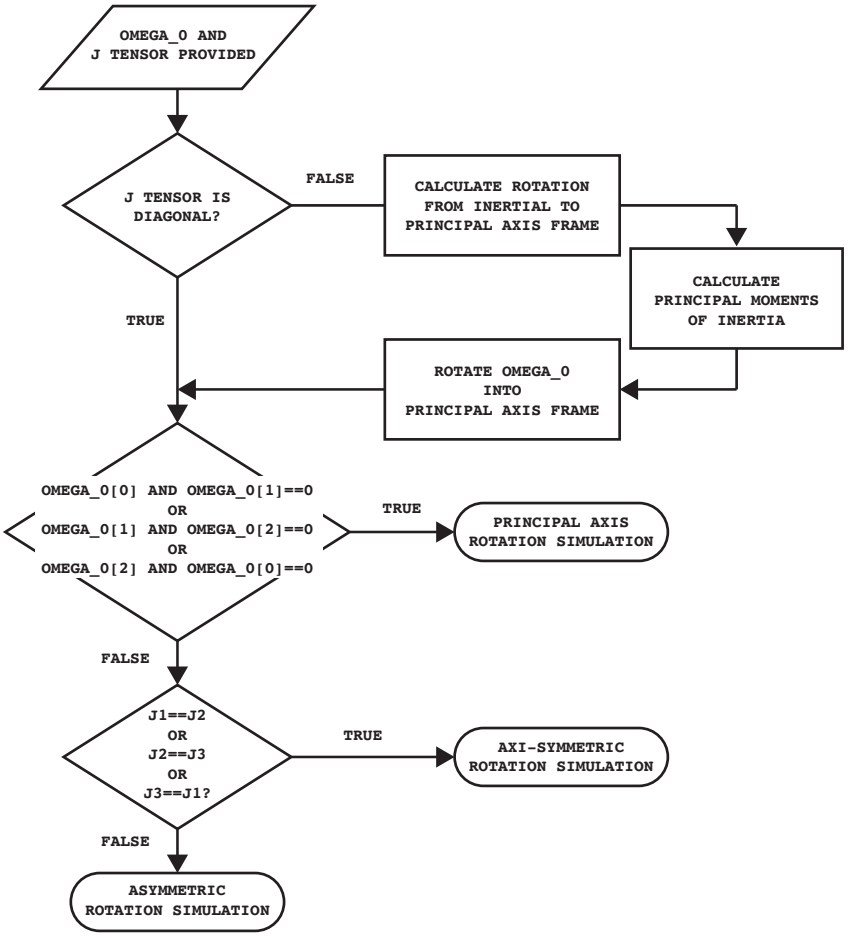


Figure 5-3: `RotationObject`'s process for deciding the appropriate rotation propagation algorithm to use

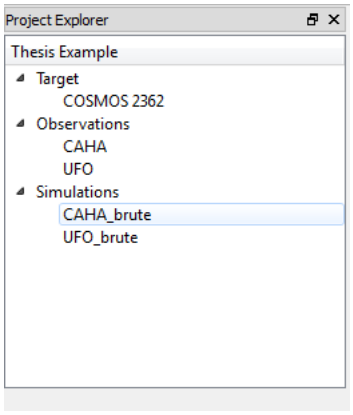


Figure 5-4: `ProjectTreeWidget` in Raxus Prime 2.0

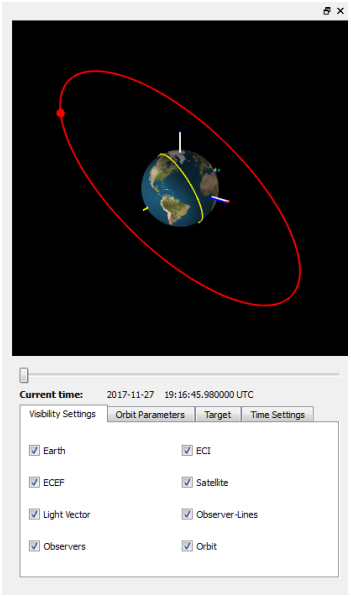


Figure 5-5: `OrbitDockWidget` in Raxus Prime 2.0

5.3.6. `OrbitDockWidget`

`OrbitDockWidget` is a GUI element in Raxus Prime 2.0 which allows the user to view a 3D representation of the target's orbit, observer locations and the axes of relevant coordinate frames. It is the evolution of the orbit tab from Raxus Prime 1.0. The decision was made, as part of Raxus Prime 2.0's policy to allow as much side-by-side comparison as possible, to implement this as a dockable/undockable sidebar widget. A screen shot of `OrbitDockWidget` can be seen in fig. 5-5

In Raxus Prime 1.0 the orbit viewer made use of the Python 3D graphic module, VPython. VPython is no longer maintained and has compatibility issues with the most recent released of numpy, a mathematics library which is used extensively in Raxus Prime 2.0. VPython is not designed to be embedded within another application and Raxus Prime 1.0 made use of a number of 'hacks' to facilitate the use of this module. Out of the necessity of using the most recent numpy releases, and because it was desirable to have a 3D module which ran natively within Raxus Prime, The VPython dependency was dropped and an OpenGL implementation was developed. This approach makes for a better user experience and is far more customisable - leaving open the possibility of easily expanding the capabilities of this widget. The methods developed for drawing a textured Earth, the axes, sunlight, orbit and click-and-drag camera controls have been wrapped into a module which is available for easy integration into any future PyQt projects.

The `OrbitDockWidget` has a panel in the lower half of the widget where the user can toggle the visibility of the various elements in the visualisation, view the Keplerian parameters of the `Target`'s orbit, view the apparent position of the satellite in local horizontal coordinates

for the location of any Observation or Simulation in the open Project. There is also a tab for setting the times to display. The user can either select a custom range of times or use the times from an Observation or Simulation. Once a range of times has been selected, the user can scroll through the times using a horizontal slider.

All of this information is made available by passing the widget a reference to ProjectManager on creation. Through the use of Qt's signal and slot system, the visualisation automatically updates in response to any changes made to the open Project.

5.3.7. TargetEditor

TargetEditor is the GUI element for editing the Target project element. This is the window where the user sets the TLE, model and moments of inertia of the target.

On opening the window, a deep copy of the target object is made, and this copy is edited. Once the user accepts the changes, the TargetEditor pushes these changes to the project through the ProjectManager object. This is done to prevent accidental changes to important information.

When the Target is saved, its orbit appears in the 3D area of the OrbitDockWidget and its orbit information appears in the panel below.

5.3.8. ObservationEditor

Similar to TargetEditor, the ObservationEditor is the GUI element where the user can edit and create Observation project elements. It also makes all changes through the ProjectManager.

The ObservationEditor is where the observer's location is set - the user is presented with a number of pre-sets for commonly used observatories. This information is pulled from a .json file where new pre-sets can be added if necessary.

This is also where light curve data is loaded into Raxus Prime 2.0. A colour is associated with each Observation, this can also be edited by the user.

When the Observation is saved, it appears in the 3D visualisation in the OrbitDockWidget.

5.3.9. SimulationEditor

SimulationEditor is the GUI for interacting with Simulation objects in Raxus Prime 2.0. As with the previously discussed editors, the Simulation which the editor works on is a deep copy of the Simulation, to prevent accidental loss of data.

In this interface the user can set the rotation and attitude parameters for the simulation. The user can decide whether to use use the locations for the satellite produced by the VirtualScene's orbit propagator or to set them manually. The sun direction and observer location can also be set manually here. There is the option to simulate the view for any Observer in the open Project and to use any Observer's light curve as the source for simulation times. It is also here that the user decides which render engine to use - POV-Ray or OpenGL.

As the simulation runs, the images are shown to the user in custom `PhotoGallery` widget that the user can browse through.

The results of the simulation are shown in a large plot area in the editor and can be viewed with and without the corrections described in section 4.5.

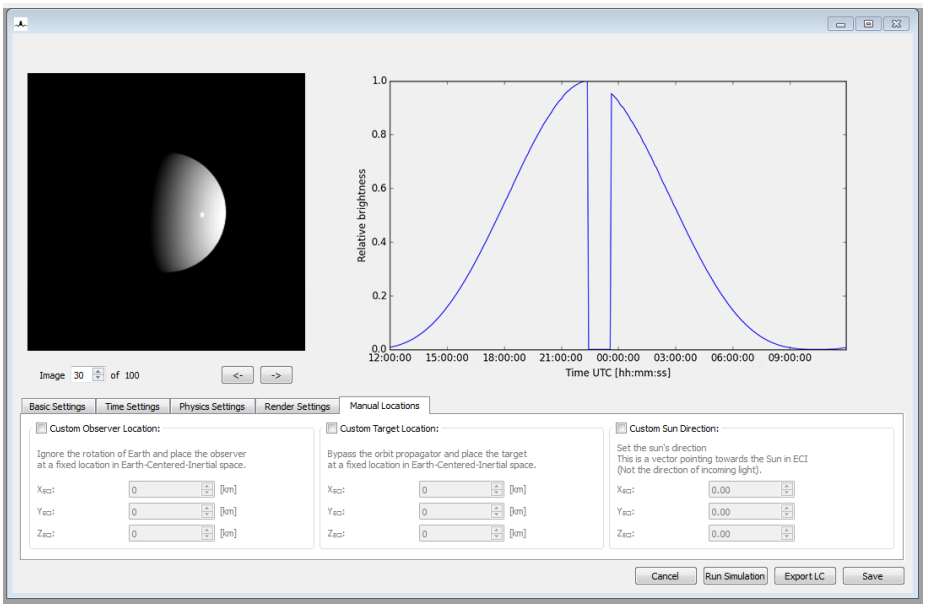


Figure 5-6: Simulation Editor of Raxus Prime 2.0

5.3.10. `LCPlotWidget`

`LCPlotWidget` is the GUI element for viewing the light curve data which is in the open project. A list of available data is provided by the `ProjectManager` and the user can check and uncheck boxes to show or hide light curves. The plot widget used the colour associated with the `Observation` or `Simulation` to provide a sense of continuity between the windows.

5.3.11. `FreqAnalysisWidget`

`FreqAnalysisWidget` is the GUI for performing the frequency analyses described in section 3.1. The user is presented with all of the data sources available in the open `Project`. And is able to select the desired technique from a dropdown box. The specific implementation of the DFT analysis is the fft function in the python package, scipy's fftpack [40]. The LSSA option uses the lombscargle function from scipy's signal module [40] Manual phase folding and PDM use functions from the package PyAstronomy [41]. The result of the analysis is shown in a plot in the `FreqAnalysisWidget` and below it, a light curve which has been phase folded with the strongest frequency component is shown (see fig. 5-7). Also shown is a list of the strongest frequency components and, the corresponding period, and the relative strength of the peak in the power spectrum.

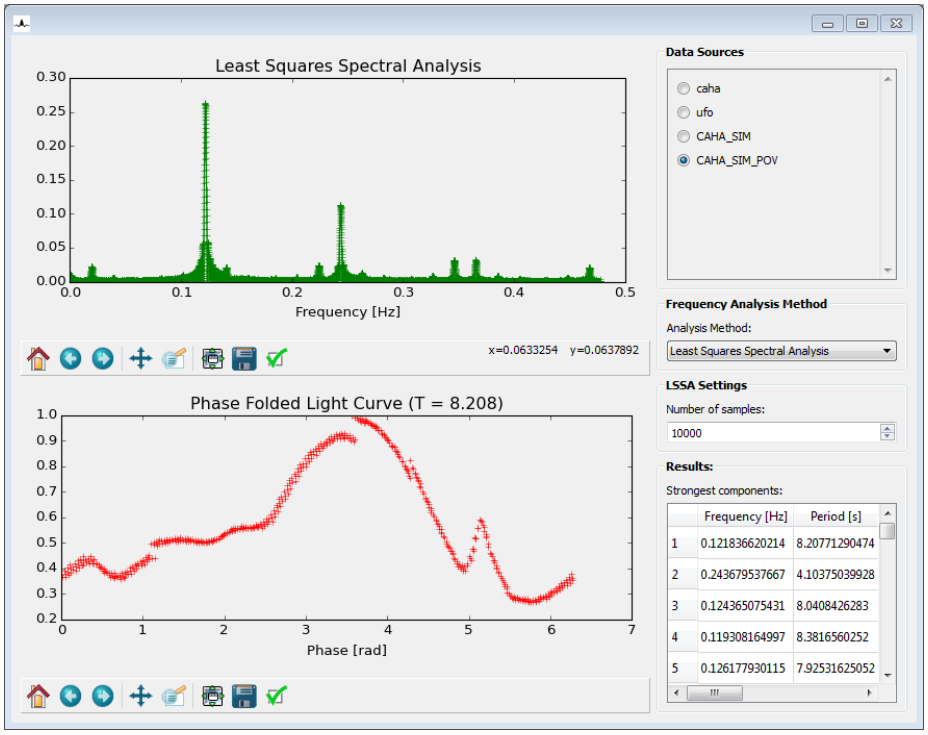


Figure 5-7: Frequency analysis tools of Raxus Prime 2.0

5.3.12. `POVRayObject`

`POVRayObject` is a class which acts as a wrapper for the POV-Ray console and contains the functions for generating .pov and .ini files, which are used to describe the scene and rendering parameters to POV-Ray.

The approach to generating these files is similar to the approach that Raxus Prime 1.0 used - a template file which contains snippets of POV-Ray 'Scene description language' is included in the Raxus Prime source code and these snippets are formatted and combined as required.

There is a crucial difference between the ray-tracing pipeline in Raxus Prime 2.0 and Raxus Prime 1.0. Raxus Prime 1.0 re-described the scene every time a frame was rendered, waited for the render to finish, analysed it and then rendered the next frame. In Raxus Prime 2.0 POV-Ray's animation capabilities are exploited and the every frame of the simulation is passed to a POV-Ray console running in a separate process. `POVRayObject` runs a separate thread which continually checks for output from the POVRay console and analyses the images. This means that Raxus Prime 2.0 does not become locked-up during a render job and the user is free to perform other tasks in the program. The improvements in rendering performance in Raxus Prime 2.0 are discussed in section 5.4.

5.3.13. `GLRenderObject`

`GLRenderObject` acts as the OpenGL equivalent of `POVRayObject` - the infomation about the virtual scene is used to generate images. This object does not make use of any intermediate libraries (except for PyQt's QOpenGLWidget wrapper for embedding in Qt appli-

cations) - the rendering of the satellite model is all a custom implementation written in raw OpenGL. The decision to do this was made so that it would be easy to implement custom shaders for more realistic real-time rendering in the future.

A function, `load_inc` was written which allows the conversion of POV-Ray .inc 3d mesh files into into OpenGL vertex buffer objects so that the 3D models which were produced for use with Raxus Prime 1.0 could also be used in the OpenGL rendering in v2.0.

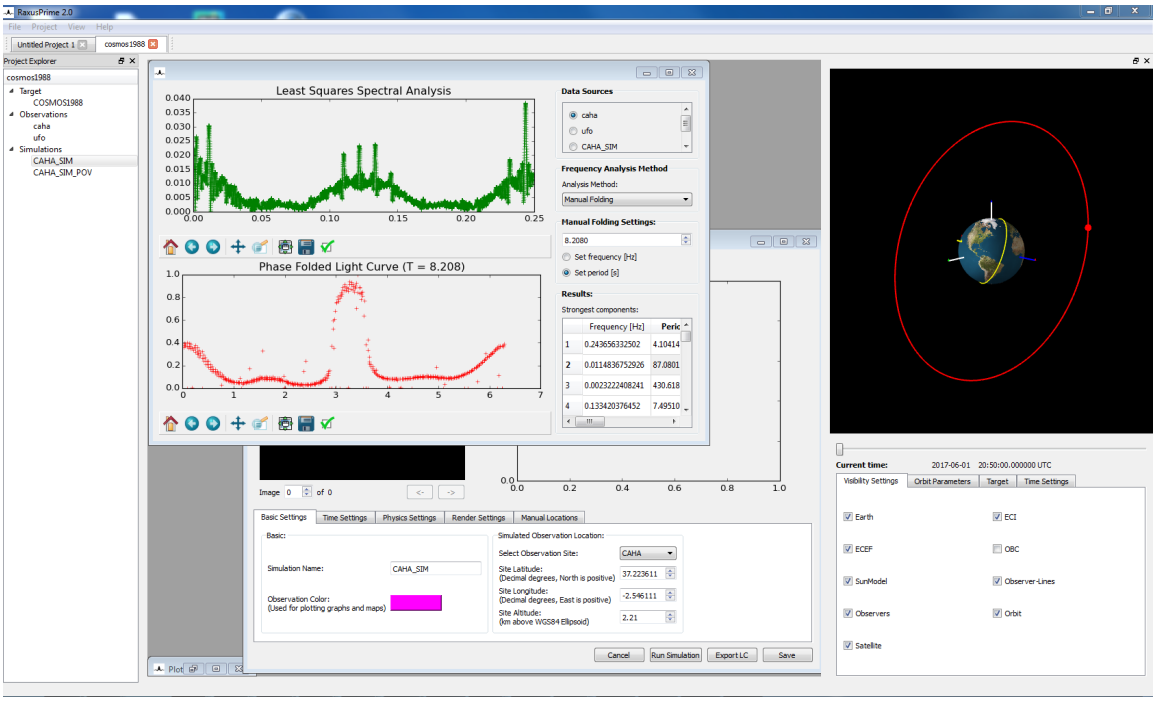


Figure 5-8: Screenshot of Raxus Prime 2.0

5.4. Benchmarks against Raxus Prime 1.0

The way that Raxus Prime 1.0 communicated with POV-Ray was the most severe bottleneck that limited render speed. Rendering was carried out synchronously and for each frame that was rendered, the scene was described in 'POV-Ray scene description language', a .pov file was saved and a new POV-Ray process was started. The biggest inefficiency in this process is the time taken for POV-Ray to initialise and parse the scene description, accounting for a minimum of 1.2 seconds per frame.

Rather than re-describing the scene and launching a new POV-Ray subprocess for each frame. Raxus Prime 2.0 makes use of POV-Ray's animation capabilities. All of the rotation matrices, sun, observer and target locations are passed to POV-Ray in a single .pov file, indexed by frame number.

In its current form Raxus Prime 2.0 is able to carry out ray-traced simulations up to 4.6× faster than Raxus Prime 1.0. When using OpenGL for simulations, Raxus Prime 2.0 is up to 25.4× faster than Raxus Prime. In tests the OpenGL simulations have been made to run up

to 125× faster than Raxus Prime 1.0 - although these optimisations have not yet been fully implemented in Raxus Prime 2.0. Figure 5-10 shows a comparison of the render times for models with increasing number of vertices.

POV-Ray's usual mode of operation is to save each rendered frames to the computer's hard disk when it has finished rendering. This presents a second inefficiency since time is being wasted in saving and re-opening potentially hundreds of files per simulation. To avoid this, the `POVRayObject` of Raxus Prime 2.0 initially attempted was to pipe the output of POV-Ray through the stdout of the POV-Ray console. Instead of writing to and reading from the hard disk, POV-Ray could pass the result through memory into Raxus Prime and continue rendering the next frame of the simulation. Raxus Prime, in parallel, could then either extract the brightness value and discard the image or save the file to the disk in a separate process. This approach showed promise initially, with an improvement of 0.0305 seconds per frame. For renders of low-poly models this can account for around 10% of the total render time.

When outputting images in this way, errors were occasionally noticed in the resulting image. Inspection of the raw hexadecimal output of the POV-Ray console showed that this was due to bytes being lost *before* they were received by Raxus Prime 2.0. This lead to this approach being temporarily abandoned pending further investigation.

(a) Image piped through the povconsole shows 'wobbly' artifacts due to the loss of data.

(b) Image saved to disk by POV-Ray shows no artefacts.

Figure 5-9

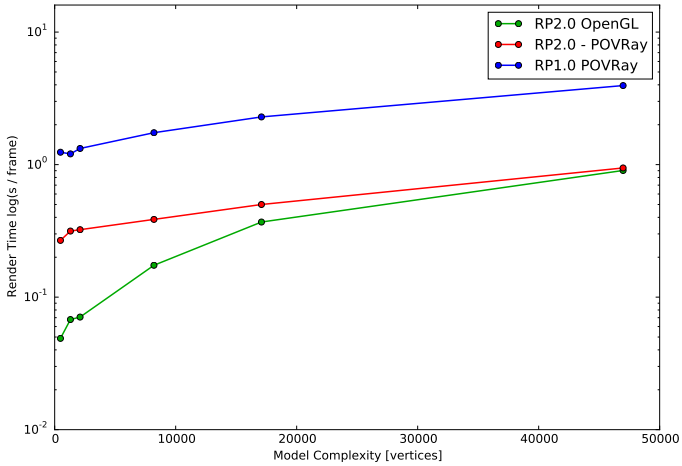


Figure 5-10: Render time per frame for models of different complexity, using Raxus Prime 1.0, Raxus Prime 2.0 (POV-Ray) and Raxus Prime 2.0 (OpenGL). Note that y-axis has log scale

6. Parameter Estimation

Even after making the assumption that the orbit, shape and inertia tensor of the target are known, there are still 6 free parameters which describe the rotation state of the target:

- 3 parameters describing the initial angular velocity in inertial space: ω_x, ω_y, ω_z
- 3 parameters describing the initial attitude in inertial space $A_{0,1}$, $A_{0,2}$, $A_{0,3}$

This is a very parameter large space to search for the correct values. Assuming a simulation can be carried out, consisting of 100 frames rendered at at 100 frames per second. Then to find an optimal solution to $\pm 1^\circ$ by brute force would take on the order of 270,000 years. Clearly this approach is not feasible and the problem must be reformulated.

6.1. Reducing the Dimensionality of the Problem

The determination of the rotation period from the light curve allows the magnitude of the angular velocity to be known. If this is done then one parameter can be eliminated by expressing the rotation axis by two angles in the body frame of the satellite (only two are required because this is only a direction, not an orientation). This reduces the problem to 5 angles: 3 bounded between 0° and 360° and 2 bounded between 0° and 180°.

As discussed in section 4.2.5 if we see a constant frequency in the light curve, then we know the object is in a non-tumbling rotation. As mentioned in section 4.2.5 a fair guess for this rotation axis is the major principal axis of inertia (a flat spin state).

The assumption is made that the target is in a flat spin state. Although this is a big assumption to make, without a better understanding of the features of light curves this has to be made so that the number of free parameters can be reduced further.

With the Cosmos 2362 example, setting the rotation axis to be the lowest energy axis produces a light curve which shows similar features to the observed light curve. A similar result can be seen with Cosmos 1988. More data is needed to determine how valid the assumption of flat spin is, and for which kinds of objects it can be made.

At this point the problem is reduced to just three unknown parameters: the Euler angles which describe the initial attitude of the debris. Fortunately these parameters have limited ranges and this opens up the possibility of a grid search (brute-force fitting) in the parameter space. This form of the problem is investigated in the next section.

This could be further reduced to just two parameters if only the direction of the rotation axis is wanted (and not the phase of the rotation around this axis). The free parameters here are the first two euler angles describing the direction in which the rotation axis is pointing in inertial space. Since we have assumed that the body is in flat spin, the attitude of the spacecraft is determined for input into the simulation by aligning the body's principle momentum axis

to this rotation axis. Consider the situation where a fit had been carried out of these two parameters and the correct axis had been determined. Because the rotation around the axis is not included in the parameter search, the light curve which would be produced would not have the same phase as the real light curve but would have the same shape. The fitness function used previously might be very good, or very bad depending on the phase - this means that we can not use this fitness function for this fit. Instead the fitting could be done by determining the least squares error between the power spectra of the light curves - since this should be the same for light curves which have the same shape, even if they are out of phase.

6.2. A Possible Approach

A number of tests were conducted to estimate the 3 parameters of the attitude of COSMOS 2362 assuming the case of flat spin, using the reduced parameter situation, described above.

This was initially attempted by using the Levenberg-Marquardt algorithm for minimising the square of the differences between the real light curve y and the simulated light curve $f(\theta, \phi, \psi)$:

$$Err = \sum_{i=N}^{i=0} (y_i - f(\theta, \phi, \psi)_i)^2 \tag{6-1}$$

This approach, although it converged in all trials, only converges to local minima. This means that the result is highly dependent on the initial parameters passed to the fitting algorithm. Even using the improved Raxus Prime 2.0 ray tracing pipeline, each minimisation took many hours to complete. These early attempts provided the intuition that the error surface for this problem must be very *bumpy* - ie. that the 3D error surface has many local minima.

To gain a better insight into the shape of the error surface a brute force 'grid search' was conducted in parameter space. The 3 parameters were varied in an evenly spaced grid of 10 values between 0 and 360° (a resolution of 36°). This means that in total 1000 simulations were run, with each simulation rendering 407 images - a total of 407000 ray-traced images! This translated into 3 days of computation. It was realised, after the simulation had finished that because of symmetry in Euler angles, it would have only really been necessary to simulate the second Euler parameter in the range of 0 to 180° - so this simulation time could have been run in half the time.

The results of this search are shown in fig. 6-2. These figures show *Error Cubes* where the three spatial dimensions represent the three Euler angles used as input and the colour represents the sum of the squared errors between the output of the simulation and the real light curves of COSMOS 2362. Figure 6-2a show the error which occurs when only the real and simulated light curves of the UFO are considered. Figure 6-2a shows the error which results from comparing only the CAHA real and simulated light curves. Figure 6-2c shows the total error of the simulation - obtained by comparing both UFO and CAHA simulations to the real data.

Despite being low resolution, these 'error cubes' confirm the suspicion that the error surface

is extremely bumpy - with many isolated regions of low error. This explains why early attempts did not converge to a global minimum. It is interesting to see that despite the light curves for UFO and CAHA looking rather different, the error surfaces show a similar appearance.

The minimum values of error for these grids were established and the corresponding light curves are shown in fig. 6-3.

The completed Raxus Prime 2.0 will be used to run simulations to determine the relationships between the input parameters and the resulting light curves in order to discover new techniques for the light curve inversion of real data. Until such relationships are discovered, a brute force search seems the most viable option to finding global minima and thus the true rotation state of the objects. A possible approach to parameter estimation which will be trialled in Raxus Prime 2.0 in the near future is outlined in fig. 6-1. It can be summarised in the following way:

1. Real light curves are analysed to determine the rotation rate of the object.
2. A fast grid search is carried out using Raxus Prime's OpenGL simulation pipeline.
3. A finer grid search around one or more of the lowest minima of this initial search is carried out using the more physically accurate POVRay rendering pipeline.
4. A Levenberg-Marquardt least squares parameter fitting procedure, using POV-Ray, is initialised with the parameters which produced the best fit in the above searches.

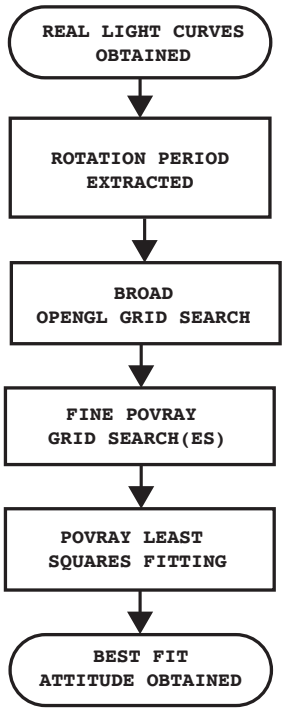


Figure 6-1: The planned automatic parameter estimation flow in Raxus Prime 2.0.

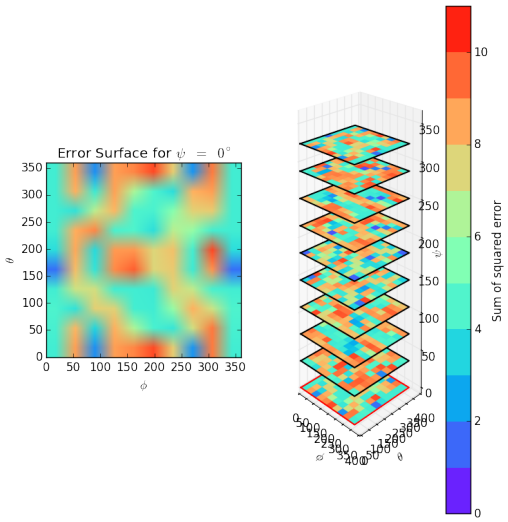
6.3. One vs. Two Observers

The results of the early fitting attempts, shown in fig. 6-3 are somewhat disappointing but illustrate an interesting point. In the parameter estimation which only considered the goodness of fit to UFO data a seemingly very good fit was found (fig. 6-3b). Had this only been a single observer experiment these parameters might have been considered the true rotation state of the debris.

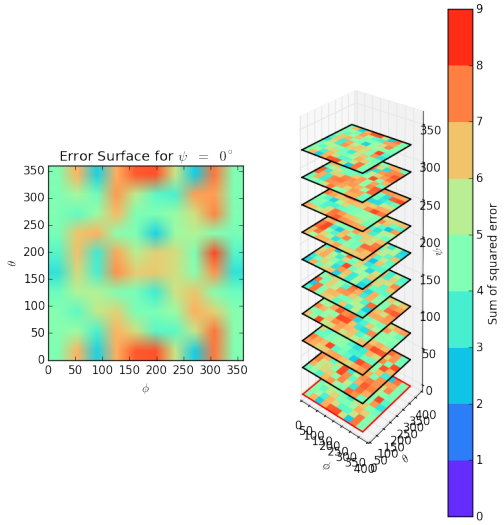
When these parameters, which were an excellent fit for UFO data are used to simulate a light curve for CAHA. These do not produce a light curve which is representative of the CAHA measured light curve. The second dataset has prevented us from falling into the trap of believing that a good fit for one observer is the absolute truth.

That being said, no simulation which was run as part of the parameter search particularly well matched the shape of the CAHA observed light curve. This could be an indication that the assumptions made are invalid but the surprisingly good fit to UFO data would seem to contradict that conclusion. The reason no good fit for CAHA was obtained is likely because the CAHA light curve for COSMOS 2362 is of a low quality due to the poor weather conditions on the night of the observation.

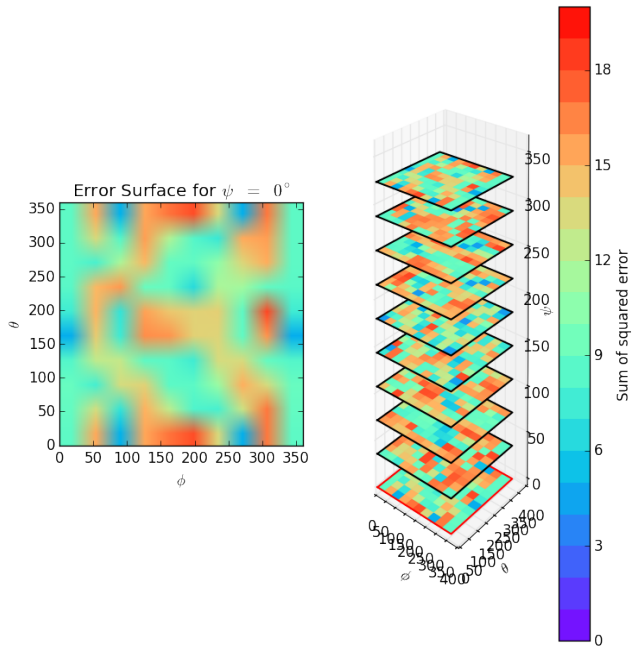
Much more, and better quality, data is needed so that this method can be fully tested.



(a) Sum of squared error between a grid of simulated UFO light curves and the real UFO light curve

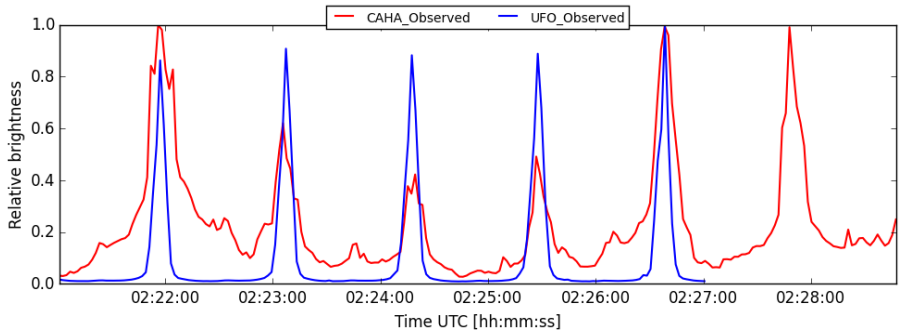


(b) Sum of squared error between a grid of simulated CAHA light curves and the real CAHA light curve

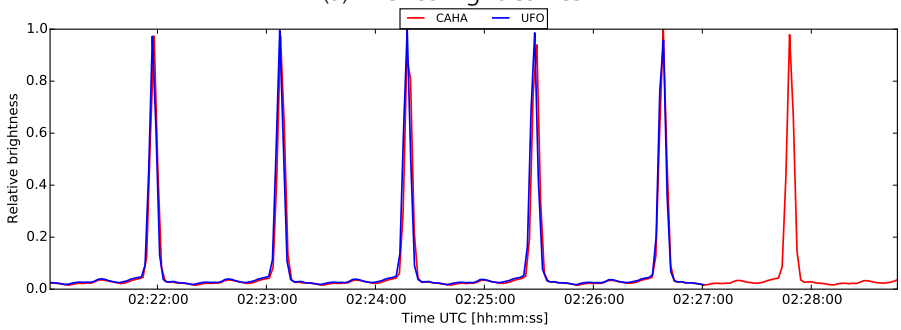


(c) Sum of combined error when both light curves are used for the fit

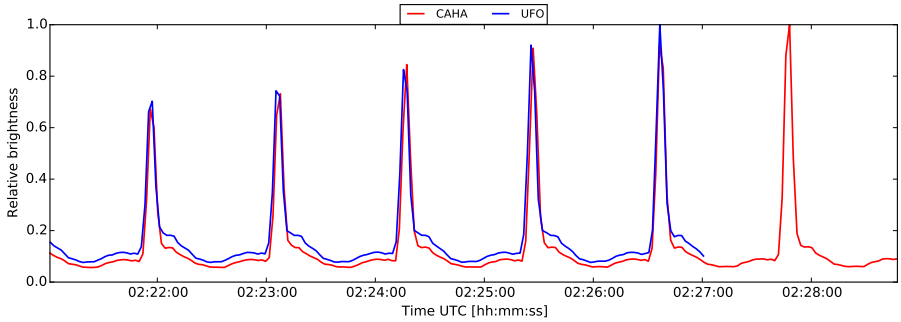
Figure 6-2: 3D 'Error cubes' showing the sum of squared differences between simulated and real light curves of COSMOS 2362 for a grid of simulated initial attitudes.



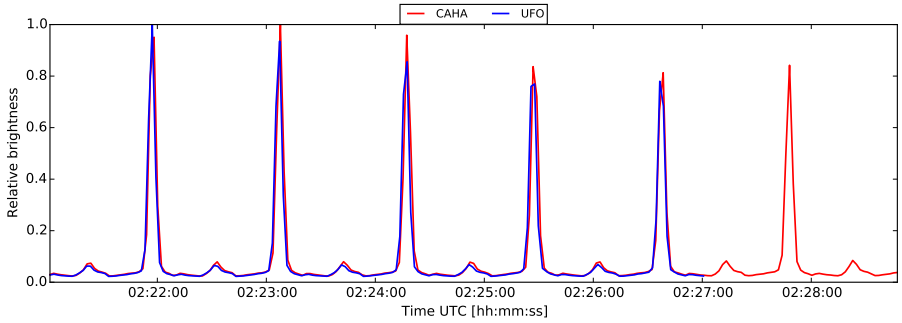
(a) The real light curves



(b) The light curves which result if the parameters obtained by only fitting to UFO data are used



(c) The light curves which result if the parameters obtained by only fitting to CAHA data are used



(d) The light curves which result if the parameters obtained by fitting to both CAHA and UFO data are used

Figure 6-3: Light curves obtained through early attempts at parameter estimation.

7. Summary

This thesis has documented the expansion of the light curve analysis tool Raxus Prime 1.0 into Raxus Prime 2.0, a well structured and maintainable research tool for the analysis of stereoscopic light curves. The program which was produced is modular in its design and the flow of information within is logical and hierarchical.

Raxus Prime 2.0 will act as a platform for the future investigation of stereoscopic light curves.

Due to reasons of limited time allotted and poor weather conditions the attempts at capturing stereoscopic light curves were quite limited in their success. From the 18 nights of observations planned between 2017-05-27 and 2017-07-20, only 2 successful stereoscopic light curves were obtained. These two light curves were of defunct GLONASS objects in MEO, of COSMOS 1988 on 2017-06-01 and COSMOS 2362 on 2017-07-19. Neither of these datasets were of particularly good quality because, in both cases one observer was observing through cloud cover. In the future more work should be done to calibrate the instruments to one another so that light curves can be compared on an absolute scale, rather than a normalised one.

Nevertheless, the two datasets were analyses using the new Raxus Prime 2.0 software, including the newly implemented phase dispersion minimisation algorithm - which was found to be very well suited to light curve analysis. The objects were found to have observed synodic rotation periods of 8.21 s for both observations of COSMOS 1988, and 70.21 s and 70.50 s for the UFO and CAHA observations of COSMOS 2362. It is likely that this discrepancy is the result of the poor quality dataset rather than a real difference in frequency.

By taking this observed rotation period to be the true rotation period of the satellite it was possible to run early attempts at automatic parameter estimation of the attitude of COSMOS 2362. Simulations were able to accurately reproduce the shape of the light curve seen at the UFO but was not able to produce a similar light curve to the observed CAHA dataset. Although this was probably due to the poor quality of the data, it highlighted the fact that stereoscopic observations provide the opportunity to check if the results of parameter estimation are valid in the general case, and not just over-fitted to a single observer.

It is hoped that in addition to Raxus Prime being used as a future research tool, the record of the procedures carried out and developed in this thesis can act as a guideline for future stereoscopic observations - in terms of planning, image correction and analysis. Stereoscopic light curve observations are a relatively unexplored concept and require much more future research. These difficult first steps are necessary for the subject's potential to be explored further in the future.

One should absolutely remember, when selecting the schedule and locations of sites for joint observations, weather absolutely must be taken into consideration since it has a huge impact on the success rates of joint observations.

7.1. Future Research

7.1.1. Cooperative Target Light Curves

To assess the accuracy of the developed model it would be extremely useful to have access to real attitude data of an observed satellite. Of even greater value would be to conduct a campaign of light curve measurements of a targets which would perform a series of planned manoeuvres, e.g. controlled rotation around various axes.

The opportunity exist for such a collaboration to take place between the University of Stuttgart's Institute for Space Systems (IRS) who have recently launched *Flying Laptop*, a small student-built satellite.

The IRS have supplied DLR Stuttgart with a high fidelity CAD model, photographs and information about construction materials which can be used to generate very accurate simulated light curves. Figure 7-1 show the light curve which would result from a coordinated roll around the $+\hat{z}_{ECI}$ axis.

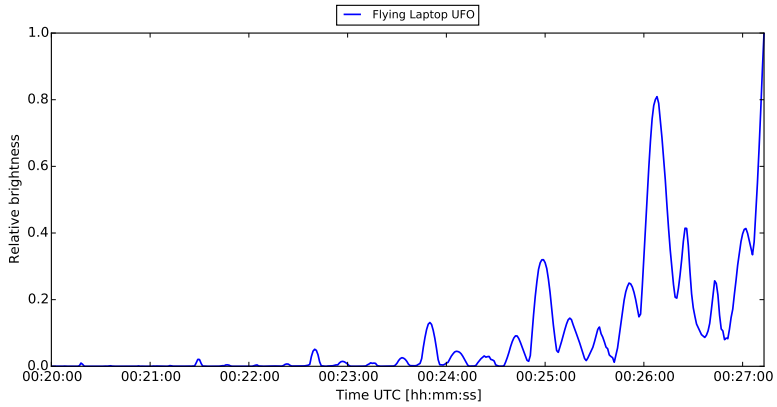


Figure 7-1: A light curve generated using a high-fidelity 3D model. The light curve shows a hypothetical coordinated roll around the $+\hat{z}_{ECI}$ axis.

7.1.2. Many-Observer Light Curves

Although the theory was not complete enough for inclusion in this thesis, it has been determined, that a phase shift between the light curves of two observers (if they have relatively similar phase angles) allows the angle between the rotation plane and the baseline plane (the plane on which the observer and the satellite lie) to be determined. This constrains the rotation axis to a cone but does not fully define it. A third observer would constrain this further, providing that they do not lie in the baseline plane of the other two observers.

7.1.3. Single Photon Detector LCs

One of the limiting factors in being able to extract useful information from stereoscopic light curves is the low frame rate of the imaging sensor used, and the associated dark time between exposures.

It is possible to obtain light curves through the use of a single photon detector [21]. These sensors work by recording the precise the time of arrival of incident photons. The output of these sensors is a sequence of timestamps recording these events.

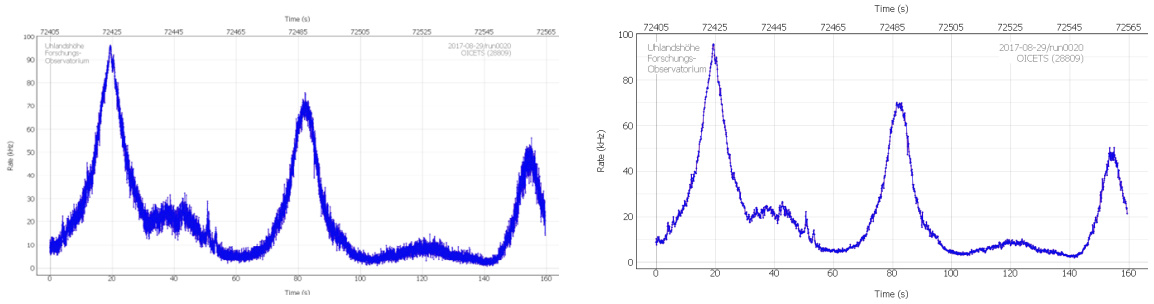
This data can be used to generate a light curve by binning the data in time. A binning width is chosen and the data is split into sections with this length. The number of counts in each bin is then determined - this produces a dataset which is equivalent to a light curve with no dark time.

The choice of binning time is a trade off; longer binning times result in a better signal to noise ratio and a smoother light curve, whereas shorter binning times mean a higher 'frame rate'

The UFO is equipped with a Photonic Solutions ID400 Single Photon Detector to measure the time of flight of returns from a pulsed laser for laser ranging experiments. The light entering this detector is filtered to only pass the same near-infrared (NIR) light emitted by the laser.

As as extension of the work done in this thesis, on 2017-08-29 a series of experiments were carried out at the UFO to assess the suitability of this set-up for passive-optical measurements. Passes of the LEO debris object, OICETS, the MEO defunct GLONASS object, Cosmos 2362 and the MEO debris object MIDAS4 were observed.

The resulting light curves for OICETS showed clear structure down to a binning time of 0.01 seconds fig. 7-2a - a 100$\times$ improvement on the traditional method of obtaining passive optical light curves! The best signal to noise ratio to binning time compromise was found to be when using a binning time of around 0.1 seconds fig. 7-2b. The single photon detector light curve of MIDAS4 produced similarly good results.



(a) Single photon detector light curve of OICETS, binned to 0.01 s.

(b) Single photon detector light curve of OICETS, binned to 0.1 s.

Figure 7-2: Light curves of the LEO space debris object, OICETS, obtained with a single photon detector. The effect of binning with different bin widths is shown.

The single photon detector data from COSMOS 2362, however did not produce a visible light curve at any level of binning. This is probably due to the fact that it is a much more distant and thus fainter object. Interestingly though when this data was analysed by phase dispersion minimisation and reconstructed using phase folding the characteristic shape of a GLONASS object's light curve emerged, albeit with a lot of noise.

To summarise: Light curves produced with a single photon detector provide an opportunity

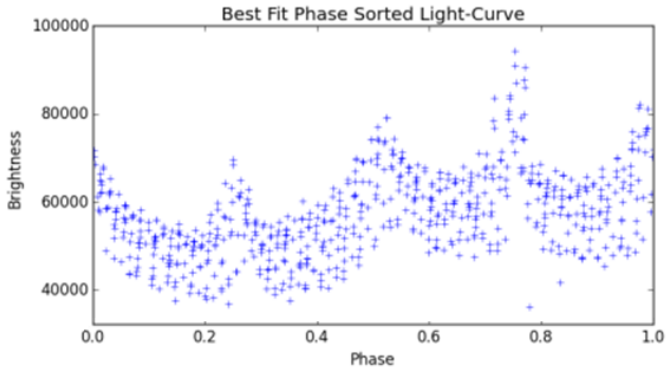


Figure 7-3: Single photon detector light curve of COSMOS 2362, binned to 1 s and phase folded to 70.30 s (result of PDM).

to study light curves at a higher time resolution than possible with traditional methods. In the context of stereoscopic light curves this would be extremely valuable since phase offsets in the light curves of multiple observers will give some indication of the objects attitude and rotation axis. However early results indicate poor performance on faint objects, such as those in MEO.

The fact that the SPD at the UFO is set up to measure NIR means that the majority of light is being filtered out before hitting the sensor. Removing this filter when recording light curves may produce better results for faint objects. On the other hand, comparing the NIR light curves to a full sprectrum light curve measured simultaneously by the CMOS sensor, may reveal different features. It is reasonable to expect this - the reflection properties of different materials are different across the spectrum and by determining the material involved in the reflection the attitude might be further constrained.

7.1.4. Space-Based Space Debris Observation

Another avenue of research which is being considered by the DLR Institute for Technical Physics' space debris group, is whether it would be possible to measure the light curve of space debris objects from space.

As part of this, Raxus Prime 2.0 was used to simulate the light curve which might result from the previously mentioned Flying Laptop satellite observing the space debris object, Envisat. A simulation of a pass on the 24th November 2017 was run. The results of this simulation are shown in fig. 7-4. An interesting feature of this simulation is a very sudden and drastic change in phase angle.

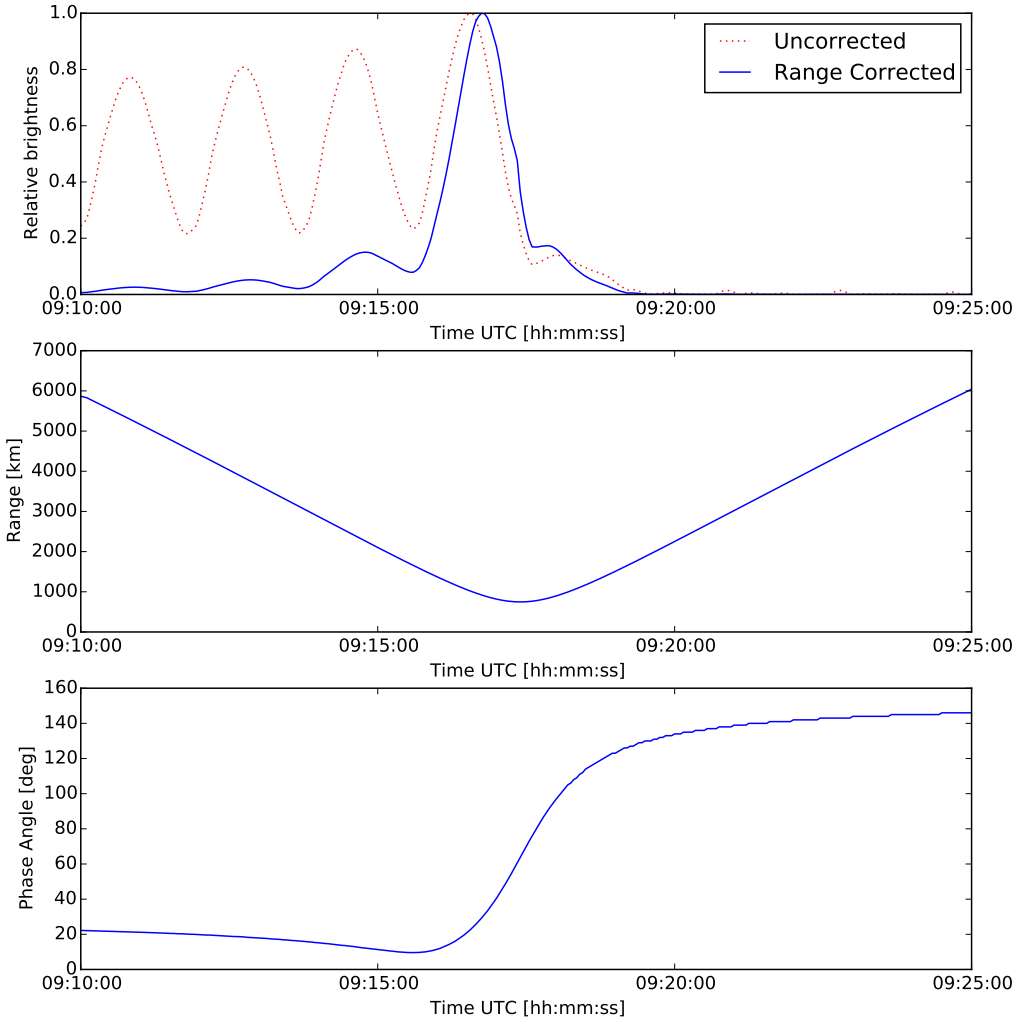


Figure 7-4: A simulated space-based light curve of Envisat as it would be seen by Flying Laptop.

Appendix A - Record of Observations

Stated here is a short summary of the planned campaign dates and resulting successful observations. A full record of observations is given in the tables overleaf.

Dates	Clear Nights UFO	Clear Nights UFO & CAHA	Single Observations UFO	Single Observations CAHA	Joint Observations
27/05/2017 - 02/06/2017	1	1	3	0	1
15/06/2017 - 18/06/2017	1	0	4	0	0
25/06/2017 – 30/06/2017	0	0	0	0	0
17/07/2017 – 20/07/2017	3	1	13	12	1
TOTAL: (18 Days)	**5**	**2**	**20**	**12**	**2**

Overleaf a complete list of observations.

Observation Period (UTC)	Target Name	NORAD ID	Observatory	Note
17/05/2017 20:36:42 - 20:42:11	Cosmos 2080	20620	UFO	
01/06/2017 20:36:01 - 20:49:58	Molniya 1-75	19807	UFO & CAHA	Unusable data: Weather cloudy at UFO
01/06/2017 20:46:01 - 21:09:10	Cosmos 1988	19750	UFO & CAHA	Partially cloudy at UFO
18/06/2017 21:04:53 - 21:34:41	Cosmos 2294	23398	UFO	
18/06/2017 21:59:08 - 22:14:47	Cosmos 2362	25595	UFO	
18/06/2017 23:21:14 - 23:30:34	Cosmos 2380	26989	UFO	
19/06/2017 00:16:31 - 00:24:42	Navstar 21	20830	UFO	
19/06/2017 00:26:15 - 00:39:18	SL-12 R/B	14984	UFO	
19/06/2017 00:46:03 - 00:51:42	Cosmos 2362	25595	UFO	
19/06/2017 00:57:24 - 01:13:42	CZ-2D R/B	37932	UFO	
19/06/2017 01:39:40 - 01:41:38	Cosmos 2363	25594	UFO	
19/06/2017 01:46:05 - 01:53:28	CZ-3B R/B	38253	UFO	
19/06/2017 01:59:21 - 02:18:57	Cosmos 2111	21008	UFO	
17/07/2017 22:18:30 - 22:30:17	Cosmos 2295	23396	UFO	
17/07/2017 23:04:12 - 23:17:29	Cosmos 2177	21853	UFO	
17/07/2017 23:44:27 - 23:59:58	Cosmos 2294	23398	UFO	
18/07/2017 00:24:38 - 00:33:59	Molniya 1-75	19807	UFO	
18/07/2017 01:35:03 - 01:44:16	Molniya 3-41	21706	UFO	
18/07/2017 01:48:08 - 02:12:42	Cosmos 2380	26989	UFO	
18/07/2017 02:31:28 - 03:03:52	Cosmos 2380	26989	UFO	
18/07/2017 23:35:09 - 23:51:55	Cosmos 2324	23735	UFO	
19/07/2017 00:27:17 - 00:29:25	CZ-2C R/B	36089	UFO	
19/07/2017 00:47:51 - 01:29:04	Cosmos 2362	25595	UFO	
19/07/2017 01:48:47 - 02:31:33	Cosmos 2362	25595	UFO & CAHA	
19/07/2017 02:30:40 - 02:52:28	Cosmos 2362	25595	UFO	
19/07/2017 20:04:25 - 20:16:16	Cosmos 2294	23398	CAHA	
20/07/2017 03:46:05 - 04:06:54	Cosmos 2235	22514	CAHA	
20/07/2017 03:22:55 - 03:43:54	Cosmos 2412	28510	CAHA	
20/07/2017 04:09:21 - 04:28:11	Navstar 43	24876	CAHA	
20/07/2017 02:57:49 - 03:20:29	SL-12 R/B	16968	CAHA	

Observation Period	Target Name	NORAD ID	Observatory	Note
20/07/2017 21:50:08 - 22:04:09	Midas 6	00574	UFO	
20/07/2017 22:10:27 -22:31:43	Cosmos 2362	25595	UFO	
20/07/2017 22:31:47 - 22:33:33	Cosmos 1867	18187	UFO	Unusable data: cloudy weather at UFO
20/07/2017 22:44:41 - 22:48:24	Studsat	36796	UFO	Unusable data: cloudy weather at UFO
20/07/2017 23:45:42 - 23:49:21	Tisat	36799	UFO	Run too short - too long to acquire target
21/07/2017 00:31:28 - 00:35:21	SL-3 R/B	13771	UFO	
21/07/2017 00:39:43 - 00:56:44	Midas 6	00574	UFO	
21/07/2017 01:05:25 - 01:29:01	Cosmos 2323	23736	UFO	
21/07/2017 01:33:19 - 01:37:42	CSTB 1	31122	UFO	Unusable data: cloudy weather at UFO
20/07/2017	CZ-2D R/B	37932	UFO	Unusable data: cloudy weather at UFO
20/07/2017 02:06:37 - 02:12:16	Garuda 1	26089	UFO	
21/07/2017 02:23:54 - 02:46:17	Directv 3	23598	UFO	
21/07/2017 02:49:32 - 02:56:45	Cosmos 2402	28113	UFO	
21/07/2017 02:50:04 - 03:12:59	Cosmos 2179	21855	UFO & CAHA	Cloudy weather at UFO
21/07/2017 03:15:23 - 03:37:43	Cosmos 2318	23622	CAHA	
21/07/2017 04:04:59 - 04:26:12	Cosmos 2381	26988	CAHA	
21/07/2017 03:40:01 - 04:01:21	Navstar 39	24320	CAHA	

Appendix B - Axis/Angle Rotation

Sometimes it is convenient to talk about a rotation around a particular axis, $\mathbf{e}$, by a given angle θ. In this thesis the notation $\mathbf{R}(\mathbf{e}, \theta)$ is used for compactness to express this rotation. Which is calculated using Rodrigues' rotation formula [42]:

$$\mathbf{R}\left(\mathbf{e}, \theta\right) \; = \; \mathbf{I} + \tilde{\mathbf{e}} \sin\theta + \tilde{\mathbf{e}}^2 (1 - \cos\theta) \tag{7-1}$$

Where $\mathbf{I}$ is the identity matrix and $\tilde{\mathbf{e}}$ is defined as:

$$\tilde{\mathbf{e}} = \begin{bmatrix} 0 & -e_z & e_y \\ e_z & 0 & -e_x \\ -e_y & e_x & 0 \end{bmatrix} \tag{7-2}$$

The rotation matrix $\mathbf{R}(\mathbf{e}, \theta)$ can be written out in full as below:

$$\mathbf{R}\left(\mathbf{e}, \theta\right) = \begin{bmatrix} e_x^2(1-\cos\theta) + \cos\theta & e_x e_y (1-\cos\theta) - e_z \sin\theta & e_x e_y (1-\cos\theta) + e_y \sin\theta \\ e_x e_y (1-\cos\theta) + e_z \sin\theta & e_y^2(1-\cos\theta) + \cos\theta & e_y e_z (1-\cos\theta) - e_x \sin\theta \\ e_x e_z (1-\cos\theta) - e_y \sin\theta & e_y e_z (1-\cos\theta) + e_x \sin\theta & e_z^2(1-\cos\theta) + \cos\theta \end{bmatrix} \tag{7-3}$$

Bibliography

[1] Satellite Industry Association. State of the Satellite Industry Report 2017. [Online; accessed Nov 2017]. URL: www.sia.org/wp-content/uploads/2017/07/SIA-SSIR-2017.pdf.

[2] T. S. Kelso. SATCAT Boxscore. http://celestrak.com/satcat/boxscore.asp. [Online; accessed Nov 2017].

[3] DJ Kessler and Nl Johnson. "The Kessler Syndrome: Implications to Future Space Operations". In: 33rd Annual AAS guidance and control conference. 2010, pp. 1–15.

[4] National Aeronautics and Space Administration. USA Space Debris Envirmonment, Operations and Policy Updates. 2011.

[5] Holger Krag. "An overview of the IADC annual activities". In: Scientific and Technical Subcommittee United Nations Committee on the Peaceful Uses of Outer Spac February (2017).

[6] Caleb Henry. ExoAnalytic video shows Telkom-1 satellite erupting debris. [Online; accessed Nov 2017]. URL: http://spacenews.com/exoanalytic-video-shows-telkom-1-satellite-erupting-debris/.

[7] Jay Mcmahon and D Scheeres. "Improving Space Object Catalog Maintenance Through Advances in Solar Radiation Pressure Modeling". In: 38 (May 2015), pp. 1–16.

[8] Stefan Scharring, Jascha Wilken, and Hans-Albert Eckel. "Laser-based removal of irregularly shaped space debris". In: 56 (Aug. 2016), p. 011007.

[9] Matthias Wieser et al. e. Deorbit Mission: OHB Debris Removal Concepts. 2015.

[10] Ben K. Bradley and Penina Axelrad. "Lightcurve Inversion for Shape Estimation of GEO Objects from Space-Based Sensors". In: International Symposium on Space Flight Dynamics 1 (2014), pp. 1–20.

[11] Toshifumi Yanagisawa, Atsushi Nakajima, and Takeo Kimura. "Motion and Shape Determination of LEO Debris Using Optical Telescope". In: (2004).

[12] D Hall and P Kervin. "Optical Characterization of Deep-Space Object Rotation States". In: Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference. 2014, p. 33.

[13] Alex Willison and Donald Bédard. "Light Curve Simulation Using Spacecraft CAD Models and Empirical Material Spectral BRDFs". In: Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference. 2015.

[14] Jiří Šilha et al. "Comparison of ENVISAT's attitude simulation and real optical and SLR observations in order to refine the satellite attitude model". In: Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference (2016), pp. 1–10.

[15] Linder Esther et al. "Extraction of Spin Periods of Space Debris from Optical Light Curves". In: Proceedings of the 66th International Astronautical Congress, 2015, pp. 1–9.

[16] Jay W Mcmahon and Daniel J Scheeres. "Shape Estimation from Lightcurves including Constraints from Orbit Determination s ;) o ; ˆ o". In: (2016).

[17] L Shakun et al. "Monitoring of the Inoperative Envisat Satellite's Behaviour". In: Odessa Astronomical Publications 26 (2013), pp. 282–284.

[18] Daniel Burandt. Simulation der Lichtkurven orbitaler Objekte und Vergleich mit Messungen. Bachelor's Thesis, Universität Stuttgart. 2017.

[19] Persistance of Vision Raytracer Pty Ltd. Persistance of Vision Raytracer. [Online; accessed Apr 2017]. URL: http://povray.org.

[20] Daniel Hampf and Fabian Sproll. "OOOS: A hardware-independent SLR control system". In: ILRS Technical Workshop. 2017.

[21] M A Steindorfer et al. "Light Curve Measurements with Single Photon Counters at Graz SLR". In: ILRS Technical Workshow, 2015.

[22] Analytical Graphics Inc. Systems Tool Kit (STK).

[23] Daniel Hampf, Paul Wagner, and Wolfgang Riede. "Optical technologies for the observation of low Earth orbit objects". In: (2014). arXiv: arXiv:1501.05736v1.

[24] Michael Mommert. "PHOTOMETRYPIPELINE : An Automated Pipeline for Calibrated Photometry". In: (2017), pp. 1–8. arXiv: arXiv:1702.00834v1.

[25] Astrometry.net. http://astrometry.net. [Online; accessed Nov 2017].

[26] R. F. Stellingwerf. "Period Determination Using Phase Dispersion Minimization". In: The Astrophysical Journal 224 (Aug. 1978), pp. 953–960.

[27] Joint Functional Component Command for Space. Space-Track.org. [Online; accessed Nov 2017]. URL: www.space-track.org.

[28] R. Kanzler et al. "Space Debris Attitude Simulation - IOTA (In-Orbit Tumbling Analysis)". In: Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference (2015).

[29] Ramses van Zon and Jeremy Schofield.
"Numerical implementation of the exact dynamics of free rigid bodies".
In: Journal of Computational Physics 225 (2007), pp. 145–164. ISSN: 00219991.
DOI: 10.1016/j.jcp.2006.11.019.

[30] F.L. Markley and J.L. Crassidis.
Fundamentals of Spacecraft Attitude Determination and Control.
Space Technology Library. Springer New York, 2014. ISBN: 9781493908028.

[31] Nicholas L. Johnson. "GLONASS Spacecraft". In: GPS World (Nov. 1994), pp. 51–58.

[32] Gunter Dirk Krebs. Gunther's Space Page: Uragan GLONASS, 11F654.
[Online; accessed Nov 2017].
URL: http://space.skyrocket.de/doc_sdat/uragan.htm.

[33] Riverbank Computing Limited. PyQt Whitepaper. [Online; accessed Apr 2017].
URL: www.riverbankcomputing.com/static/Docs/PyQt4/pyqt-whitepaper-a4.pdf.

[34] Riverbank Computing Limited.
Qt — Cross-platform software development for embedded & desktop.
[Online; accessed Apr 2017]. URL: www.qt.io.

[35] G. van Rossum, B. Warsaw, and N. Coghlan. PEP 8 – Style Guide for Python Code.
www.python.org/dev/peps/pep-0008/. [Online; accessed Apr 2017].

[36] A. Patel et al. Google Python Style Guide – Revision 2.59.
google.github.io/styleguide/pyguide.html.
[Online; Accessed Nov, 2017].

[37] Robert Lauchie Scott, Alex Ellery, and Martin Levesque. "Non-resolved detection of objects performing On Orbit Servicing in Geostationary orbit".
In: AMOS Technical Conference (2011), p. 10.

[38] Eric E Becklin. "Stratospheric observatory for infrared astronomy (SOFIA)".
In: Advances in Space Research 36.6 (2005), pp. 1087–1090.

[39] Astropy Collaboration et al. "Astropy: A community Python package for astronomy".
In: Astronomy & Astrophysics 558, A33 (Oct. 2013), A33.
DOI: 10.1051/0004-6361/201322068. arXiv: 1307.6212 [astro-ph.IM].

[40] Eric Jones, Travis Oliphant, Pearu Peterson, et al.
SciPy: Open source scientific tools for Python. [Online; accessed Nov 2011]. 2001–.
URL: http://www.scipy.org/.

[41] PyA group. PyAstronomy. [Online; accessed May 2017].
URL: https://github.com/sczesla/PyAstronomy.

[42] Serge Belongie. Rodrigues' Rotation Formula. [Online; accessed June 2017]. URL:
http://mathworld.wolfram.com/RodriguesRotationFormula.html.