DEPARTMENT OF SPACE & CLIMATE PHYSICS
MULLARD SPACE SCIENCE LABORATORY

Coursework Cover Sheet

Module Name: SPCE0014: Space Science and Engineering Individual Project

Tutor: Prof. Daisuke Kawata

Coursework number/title: Interim Report

Student Number: Max Hallgarten La Casta

DEPARTMENT OF SPACE & CLIMATE PHYSICS
MULLARD SPACE SCIENCE LABORATORY

AUTOMATED SKY SURVEY ANALYSIS METHODS: INTERIM REPORT

Max Hallgarten La Casta

Supervisor: Prof. Daisuke Kawata

Department of Space & Climate Physics, Mullard Space Science Laboratory,
Holmbury St. Mary, Dorking, Surrey, RH5 6NT, UK

Word count: 3000

1 INTRODUCTION

1.1 Background

The Japan Astrometry Satellite Mission for Infrared Exploration (JASMINE), illustrated in Figure 1.1, was selected by the Institute of Space and Astronautical Science (ISAS) and the Japan Aerospace Exploration Agency (JAXA) in May 2019 as one of JAXA's competitive M-class missions [1]. This class represents missions with a cost below ¥15 billion (£100 million) launched using the Epsilon rocket at a rate of approximately five missions per decade [2].

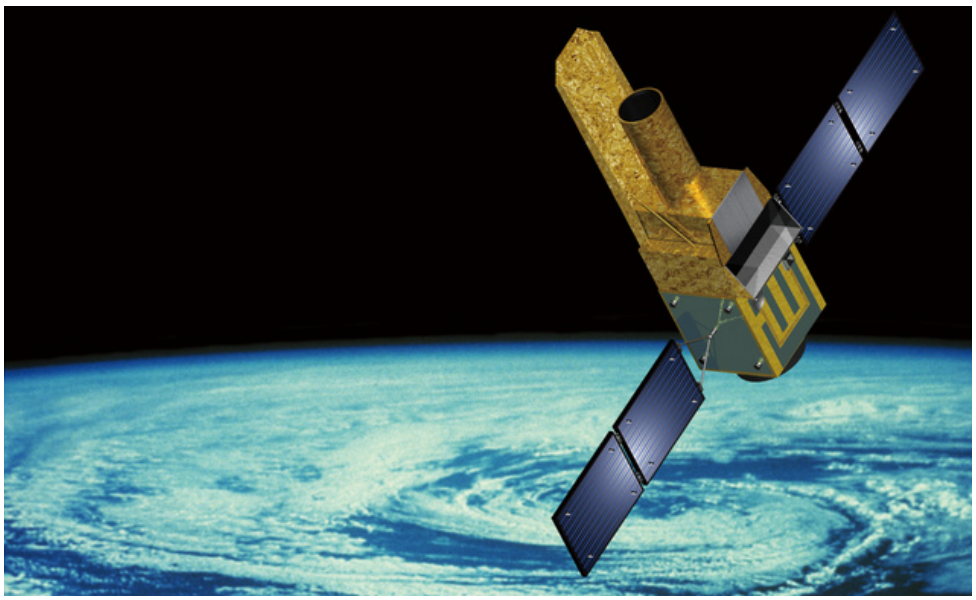


Figure 1.1: Artist's impression of the JASMINE spacecraft [3].

Designed initially for astrometry, JASMINE has three primary scientific goals [1, 4]:

1. Investigation of the structure of the Milky Way's central core.
2. Exploration of the formation history of the Milky Way.
3. Discovery of Earth-like habitable planets.

The first two scientific goals are conducted through observation of the Galactic centre and the Galactic mid-plane [1]. The third scientific goal is achieved by imaging M-type stars and conducting transit observations [1]. One of the main differences between these various targets, apart from the different science conducted by observing them, is their distribution: the targets grow more dispersed from the Galactic centre, to the mid-plane, and to the most dispersed, M-type stars, as illustrated in Figure 1.2.

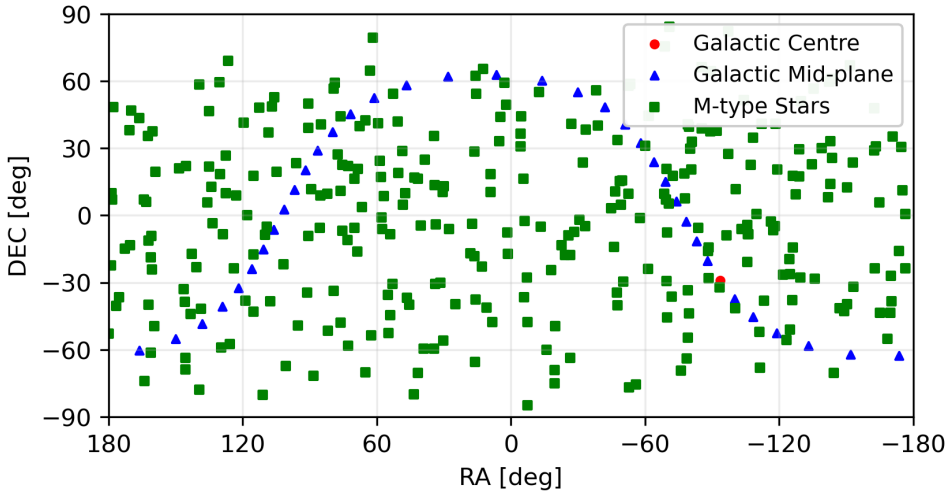


Figure 1.2: Distribution of targets in the sky, highlighting the higher dispersal of M-type stars when compared to both the Galactic centre and the Galactic mid-plane. It should be noted that the sizes of the markers do not correspond to the true sizes of the targets.

Observations of the Galactic centre in the visible spectrum are very difficult: the large amount of interstellar dust between the target and observers in the Solar System absorbs and scatters a large proportion of light in the visible band [5]. Consequently, JASMINE, as indicated by its full name, will observe in the infrared band, specifically in the near-infrared at 1.1 to 1.7 µm, to take advantage of the significantly lower extinction at these wavelengths [3].

The JASMINE spacecraft will be launched into Low Earth Orbit (LEO) at an altitude of around 550 km [1]. Furthermore, it will be a Sun Synchronous Orbit (SSO) with a Local Time of the Ascending Node (LTAN) of either 06:00 or 18:00 [1]. The spacecraft will be a pseudo-terminator-riding satellite with near constant illumination from the Sun.

1.2 Problem Definition

JASMINE's scientific payload is comprised of a modified Korsch-type infrared telescope with a 30 cm primary mirror and 3.9 m focal length [1]. This optical system was designed to be highly compact to fit within the limited available volume [1]. The scientific payload and the observing strategy are designed to meet strict thermal requirements: the detector must be cooled below 180 K with a variation of less than 0.7 K during each observation, and thermal-structural coupling must be limited to deflections less than 10 nm on the focal plane during observations [1]. Telescope pointing is, therefore, heavily constrained to ensure that the thermal system operates correctly.

The dominant factors for determining whether observations can be conducted are the positions of the Earth and the Sun in relation to the potential targets, as shown in Figure 1.3 for the Galactic centre. Angular constraints are introduced by these bodies to ensure that the payload radiator and telescope Sun shield can operate effectively, both for thermal constraints and to avoid stray light from entering the telescope. Constraints are also introduced with the Earth due to the scattering of light in the upper atmosphere making observations infeasible [6].

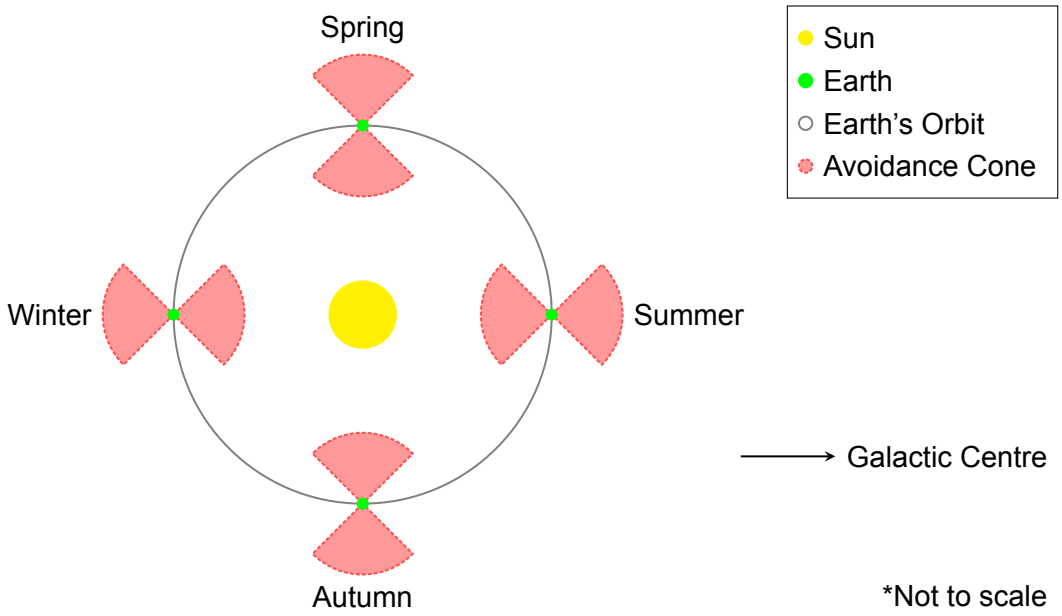


Figure 1.3: Observation constraints introduced by the Sun throughout the year showing that the Galactic centre cannot be observed during winter and summer due to thermal and pointing constraints.

The problem can be summarised, therefore, as JASMINE requiring sky survey plans which are heavily constrained by strict thermal and pointing requirements, further compounded by the large number of potential targets (over 300 M-type stars alone) and the long mission duration of over three years.

1.3 Project Objectives

The primary objective of the Automated Sky Survey Analysis Methods (ASSAM) project is to evaluate potential sky surveys for JASMINE to maximise the scientific return of the mission given the multiple targets and constraints.

Automated scheduling of space telescopes has been used to great effect in the past three decades: for example, the Hubble Space Telescope uses a form of artificial intelligence to process the high number of requests made by astronomers for telescope time [7, 8]. Automation is an important tool for optimising the use of the telescope as it can be difficult and labour-intensive to schedule observations manually, particularly over extended time periods such as the several years needed for extended surveys.

Access to many of the sky survey software packages developed for various missions is typically extremely limited. An aim of the ASSAM project, therefore, is to develop a flexible, modular tool which can be used for various different missions, under an open source licence to enable greater accessibility to space telescope mission planning.

The objectives for the ASSAM project can be summarised under three main branches:

1. Generation of target observation visibility to aid manual scheduling.
2. Full automation of sky survey scheduling.
3. Creation of a lightweight software package with high computational performance to enable rapid, iterative, mission planning.

2 COMPLETED WORK

The main software flowchart for ASSAM is presented in Figure 2.1. Three main modules have been implemented: the propagation module, the visibility module, and the visualisation module. The current implementation of software represents sufficient capability to conduct manual scheduling of space telescope observations. This satisfies the primary objective of ASSAM as outlined in Section 1.3.

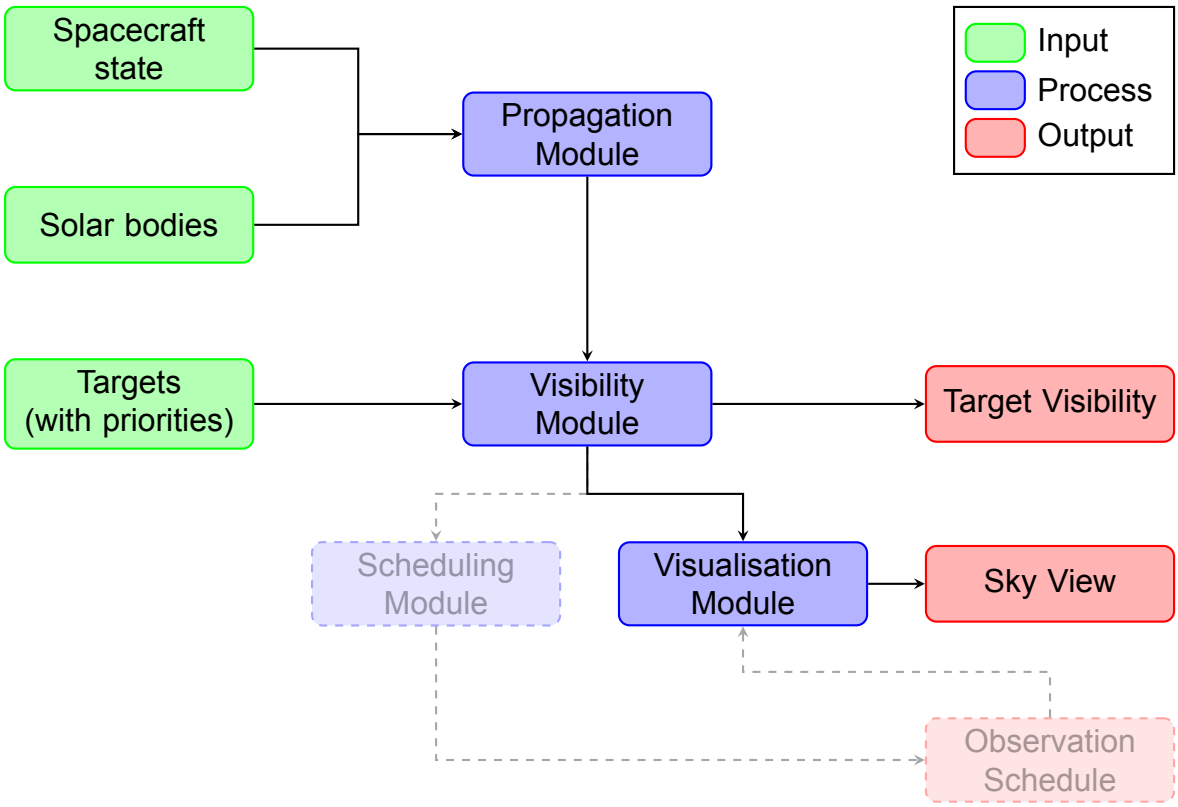


Figure 2.1: Software flowchart for ASSAM illustrating the currently implemented main inputs, processes, and outputs. The scheduling module and its output have not been implemented in the current version of the software.

Python [9] was chosen as the programming language for ASSAM due to its growing popularity, both in general use and specifically for astronomy, and ease of use as a very high level programming language [4]. ASSAM is able to benefit from the significant number of third-party libraries developed for Python such as Astropy [10, 11], a community-developed astronomy package for Python which is heavily leveraged by ASSAM. Other software used by ASSAM includes the General Mission Analysis Tool (GMAT) [12], an open source astrodynamics package developed by the National Aeronautics and Space Administration (NASA).

GMAT was initially chosen during prototyping due to the perceived limitations of alternative software [4]. Combined with the author's previous experience with GMAT, this significantly reduced the time between the project commencing and the first results being generated. Nevertheless, the modular architecture of ASSAM means that implementation of other orbital propagators should not pose a major challenge.

Standard version control practices have been implemented during the development of ASSAM to keep track of the software's development. Git was chosen "due to its widespread popularity, and ease of use with multiple collaborators" [4]. In addition to the local repository, an online "remote" repository* was created on Github, both to act as a backup but also to "enable future collaboration" [4]. This repository is currently private, however it is planned to make the code public in the near future.

2.1 Reference Systems and Frames

One of the primary concerns during development was ensuring compatibility between the different software packages used in ASSAM. This was particularly important when passing the spacecraft state between GMAT and Astropy, therefore defining a common reference frame between the software was critical.

The primary reference system chosen for use in Astropy was the Geocentric Celestial Reference System (GCRS). This system is aligned with International Celestial Reference System (ICRS) in rotation, however with a geocentric origin in contrast to the barycentric ICRS [13]. These systems are realised as the Geocentric Celestial Reference Frame (GCRF) and the International Celestial Reference Frame (ICRF) respectively through observations of several thousand extragalactic radio sources [13].

GMAT supports output in a variety of different formats and frames with two main base theories: IAU-1976/FK5 and IAU-2000 [12]. Coordinates can be exported from GMAT in ICRF as it is based on IAU-2000. Although this term refers specifically to the barycentric frame, GMAT supports output in the geocentric form by translating the origin of the frame to the geocentre. GCRF was, therefore, suitable as the common frame between Astropy and GMAT.

The spacecraft state, both position and velocity, calculated by GMAT in the common frame was used to create a spacecraft-centred reference frame offset from the Earth's centre but aligned with ICRS/GCRS. This offset form of GCRF was used as the main reference frame for all consequent calculations. The geometry of this spacecraft-centred frame is presented in Figure 2.2.

*Access available upon request.

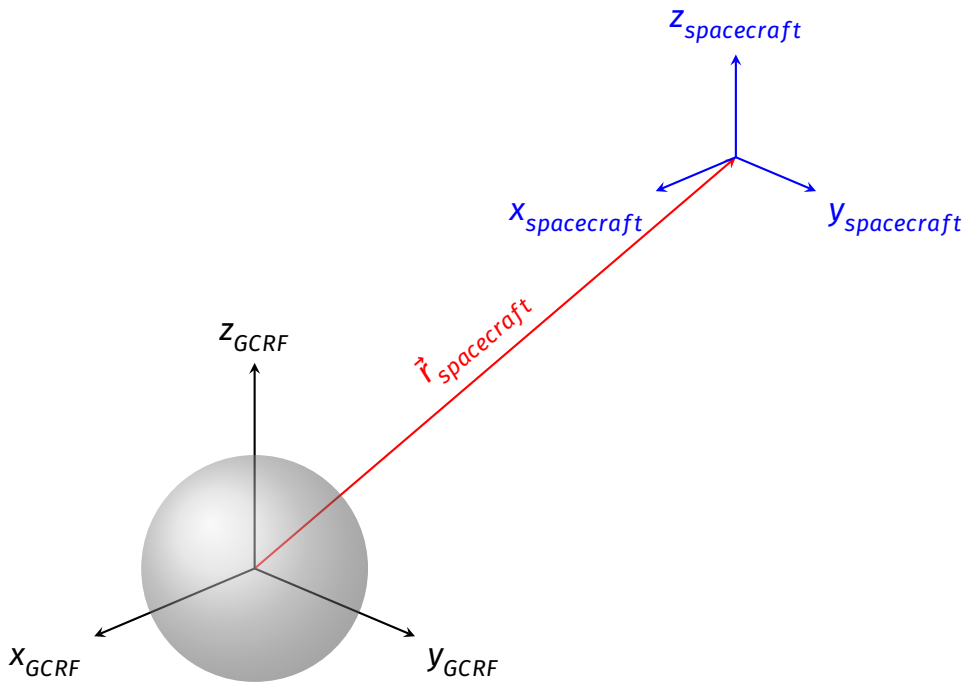


Figure 2.2: Illustration of the spacecraft-centred reference frame. The spacecraft frame (shown in blue) is aligned with GCRF (shown in black), but with an offset from the geocentre by position vector $\vec{r}_{spacecraft}$ (shown in red).

One of the phenomena which must be accounted for with astronomical observations is aberration, an apparent angular movement of the observed object. The two main forms are light-time aberration, due to the time taken by light to travel from the object to the observer, and stellar aberration resulting from the relative motion of the observer [13]. The application of aberration corrections is inconsistent between the two software: Astropy applies both light-time and stellar corrections in GCRS [14], while GMAT does not apply any corrections, except optionally when using the contact and eclipse locators [12]. Nevertheless, these corrections are not considered by Astropy when defining an offset GCRS frame, therefore the two software are consistent for the purpose of creating the spacecraft-centred frame.

2.2 Software Modules

2.2.1 Propagation Module

The propagator module currently interfaces with GMAT via script files and the commandline. Propagation using GMAT is executed in three main steps: script generation, script execution, and state importing. Script generation is conducted by importing a template file, modifying key parameters such as the start and end dates, and the Keplerian orbital elements, before exporting a modified script file. This modified script file, which includes the path to export the spacecraft state, is then executed via the commandline with flags to suppress GMAT's Graphical User Interface (GUI). The spacecraft state is then loaded, and used to create the spacecraft reference frame.

The propagation model implemented in GMAT was relatively simple with a limited number of low order gravity field terms to capture the nodal precession seen with SSOs. Perturbations such as atmospheric drag, Solar Radiation Pressure (SRP), and other bodies were ignored: corrections for these perturbations would be conducted as part of the spacecraft's operations and modelling these would be outside the scope of this project. The Runge-Kutta 89 numerical scheme was used due to its high accuracy and computational performance for LEO propagation [12]. GMAT uses a variable time stepping scheme to reduce the computational cost of propagation. However, it may be useful for scheduling to have a fixed time step and therefore the output is interpolated using a given fixed time step.

Solar body propagation was conducted using Astropy. The states were returned in GCRF as the apparent positions, including the previously discussed aberration effects. The DE430 ephemeris generated by the Jet Propulsion Lab (JPL) was used via the jplephem package [15]. This was selected over Astropy's default built-in ephemeris as it was found to have improved accuracy and computational performance. The calculated positions were then transformed into the spacecraft reference frame previously generated within the module.

The angular radius of each solar body is important for subsequent visibility calculations which check whether targets are visible or not. This property was calculated using the solar body's radius and apparent slant range from the spacecraft:

$$\rho_{solarbody} = \arcsin\left(\frac{r_{solarbody}}{s_{spacecraft,solarbody}}\right), \tag{2.1}$$

where $\rho_{solarbody}$ is the solar body angular radius, $r_{solarbody}$ is the solar body radius, and $s_{spacecraft,solarbody}$ is the slant range between the spacecraft and the solar body. The geometry is shown in Figure 2.3. The angular radius for each solar body was calculated for each time step as the angular radius could vary in time as the slant range varied.

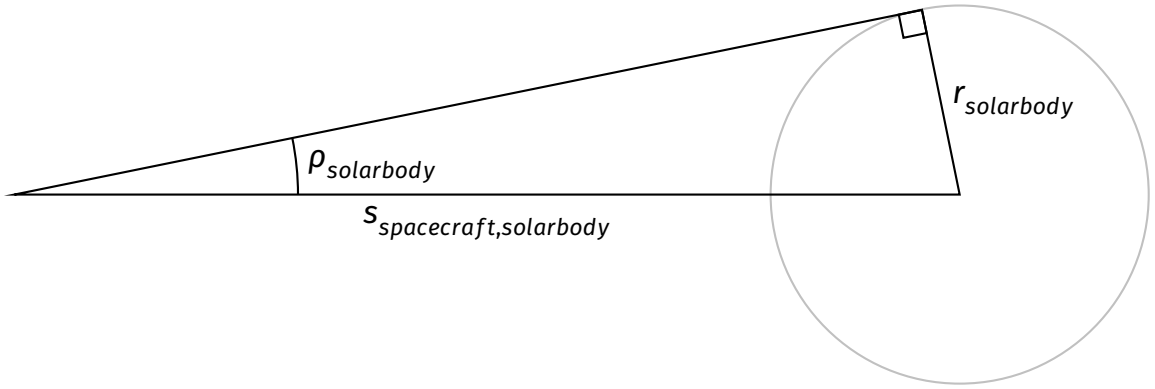


Figure 2.3: Solar body angular radius geometry.

The solar body coordinates transformed into the spacecraft reference frame, its angular radius, and other properties are stored in a list of solar body objects. This list of solar bodies and the generated spacecraft frame are then passed to the visibility module.

2.2.2 Visibility Module

The YAML Ain't Markup Language (YAML) format was selected for providing target information to ASSAM. This format has the advantage of being both human- and machine-readable, allowing for a large amount of flexibility both when creating and modifying the data, either manually or with software.

An example of the created format is presented in Listing 2.1, illustrating the main information fields. The key for each target block acts as the name for the respective target. Each target is given a category which will be used for filtering the list of targets into the different categories, for example, to filter for M-type stars. Additionally a target priority is provided which will be used during scheduling. For example, some targets such as the Galactic centre take the highest priority.

Each target can be comprised of multiple subtargets to capture complex geometries, such as JASMINE's Galactic centre survey field, as shown in Figure 2.4. Each of these subtargets are defined by their reference frame (any of the available from Astropy), the coordinates in the specified frame, and their geometry. Currently two options are implemented: circular or rectangular, using angular radius, or angular width and height respectively in degrees.

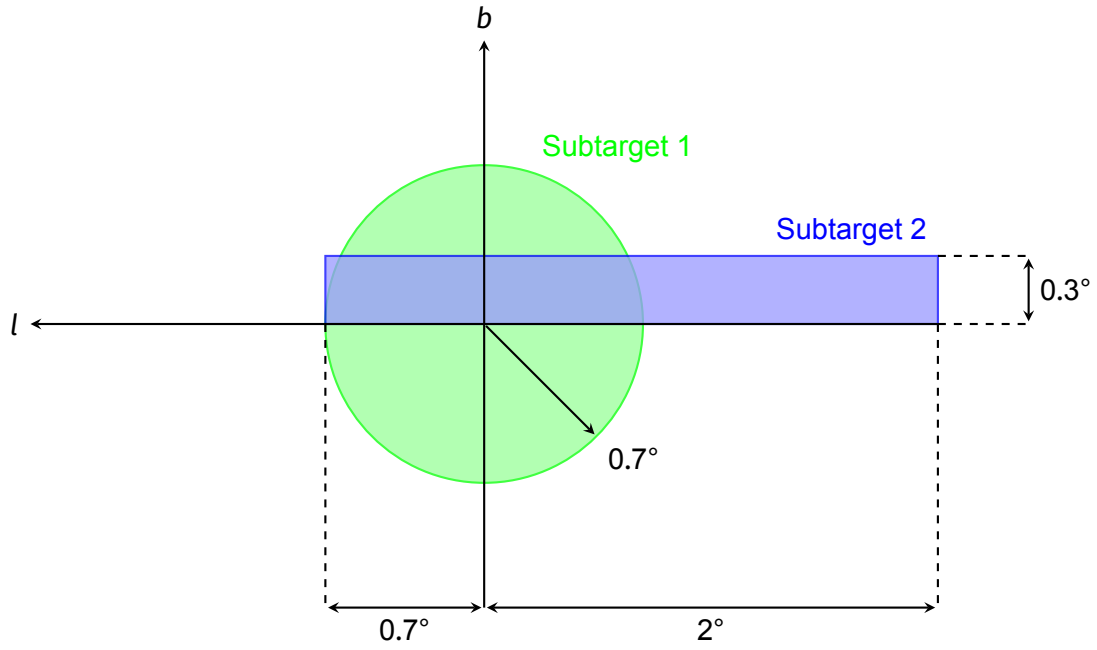


Figure 2.4: Galactic centre target geometry. Galactic longitude and latitude are indicated by l and b respectively. Redrawn and adapted from [3].

The full set of targets are stored in a list of target objects which are created when the targets are imported from their definition files. These objects contain the information from the definition files, and allow for further information to be stored as required by later processes both in the visualisation module and in others. Each of the target objects includes a list of subtarget objects to store the relevant information.

Listing 2.1: Example excerpt from a target definition file.

```
galactic_centre:
    category: galactic_centre
    priority: 1
    subtargets:
        region_1:
            frame: galactic
            centre: [0, 0]
            shape: circular
            width:
            height:
            angular_radius: 0.7
        region_2:
            frame: galactic
            centre: [-0.65, 0.15]
            shape: rectangular
            width: 2.7
            height: 0.3
            angular_radius:
00045753-1709369:
    category: m_dwarf
    priority: 3
    subtargets:
        star:
            frame: icrs
            centre: [1.2398172, -17.1602658]
            shape: circular
            width:
            height:
            angular_radius: 0.3
00064325-0732147:
    category: m_dwarf
    priority: 3
    subtargets:
        star:
            frame: icrs
            centre: [1.679958, -7.5380623]
            shape: circular
            width:
            height:
            angular_radius: 0.3
```

Target visibility calculations have been implemented between circular targets and solar bodies. For rectangular targets, these are currently approximated using a bounding circle. However, the error which is introduced is minor due to the small target sizes considered for JASMINE. The bounding circle provides a conservative estimate of the true visibility therefore this approximation is acceptable for initial calculations.

Under the implemented scheme, a target is visible if it does not intersect any occulting body such as a solar body. This geometry is presented in Figure 2.5. Expressed mathematically, the target is visible if the following equation is satisfied [4]:

$$\theta \geq \rho_1 + \rho_2, \tag{2.2}$$

where θ is the angular separation between the region centres, and ρ_1 and ρ_2 are the regions' angular radii. This was considered a "hard" radius constraint, where the solar body itself would occult the target.

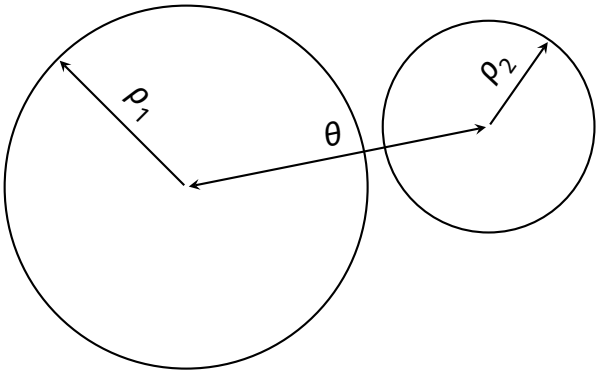


Figure 2.5: Separation geometry for two circular regions. Redrawn from [4].

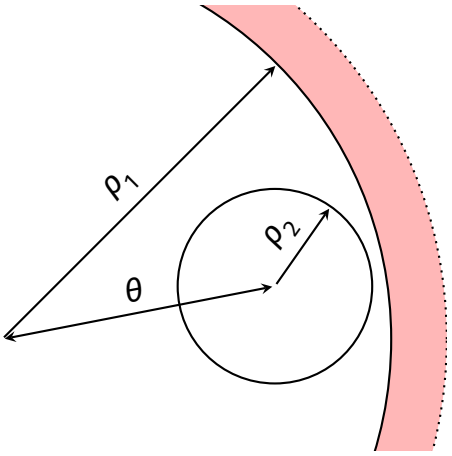
The concept of a "soft" constraint was introduced for pointing restrictions in regions where a solar body was not directly occulting the target. An example is restrictions near Earth: the angular radius of the Earth as seen from JASMINE's orbit is ~67°, but to avoid atmospheric scattering, an additional buffer zone must be added around this. The buffer was implemented as a "soft" constraint of 90° from the Earth's centre.

The "soft" constraints were implemented as annuli centred on a solar body with defined inner and outer radii. The region between these radii are considered a "no-go" zone, where targets are not visible even if there is no occulting solar body present. Each solar body can have an arbitrary number of these "soft" constraints. For example, the Sun was given two such constraints: 0 to 45° and 135 to 180° to capture the thermal and optical limitations when pointing both towards and away from the Sun.

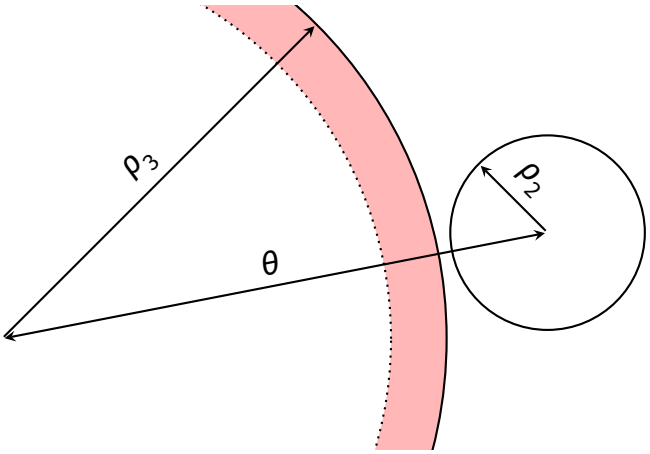
The separation geometry for "soft" radius constraints is presented in Figure 2.6. For this case, it is simpler to consider the conditions under which the target violates the constraint, and then invert this result to determine whether the target is visible. The constraint is violated when the following equation is satisfied:

$$\rho_1 - \rho_2 < \theta < \rho_2 + \rho_3, \tag{2.3}$$

where θ is the angular separation between the constraint centre and the target, ρ_1 and ρ_3 are the "soft" constraint's inner and outer angular radii, and ρ_2 is the target angular radius. These "soft" constraint parameters are provided to ASSAM through an additional YAML file which is used to define the properties of the solar bodies.



(a) Inner geometry.



(b) Outer geometry.

Figure 2.6: Separation geometry for "soft" radius constraints. The "no-go" zones, where targets would not be considered visible, are illustrated in red.

It was initially intended to use the Astropy affiliated regions package to handle more complex geometry intersection calculations [4]. However, the package's use case differs from the intended role in ASSAM as the package was designed primarily for image annotation instead of intersection calculations [16]. The regions package in its current form is, therefore, not suitable for this application.

2.2.3 Scheduling Module

The scheduling module is not currently implemented as shown in Figure 2.1. Nevertheless, investigations are underway into dynamic linear programming and related methods for optimising the space telescope's observation strategy.

2.2.4 Visualisation Module

The purpose of the visualisation module is to render plots of the visible sky from the viewpoint of the spacecraft. These plots illustrate the positions of the targets and solar bodies, highlighting occurrences when the targets are obstructed.

The module generates raster images by calculating whether a given coordinate in the sky is obstructed by a target and/or a solar body. This is produced by creating a structured mesh of points from which separations to the targets and solar bodies are calculated, as shown in Figure 2.7. The visibility of each mesh point is calculated by comparing its separation to the both the hard and soft radii of the targets and solar bodies. This operates in a similar manner to target visibility, as outlined in Section 2.2.2, however producing a series of all-sky, two-dimensional spatial arrays for each time step, as opposed to a one-dimensional vector in time for each target.

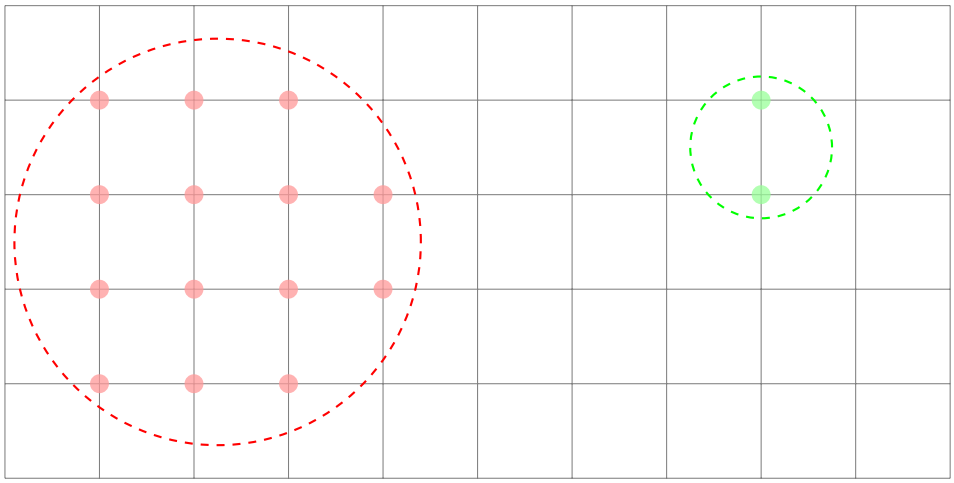
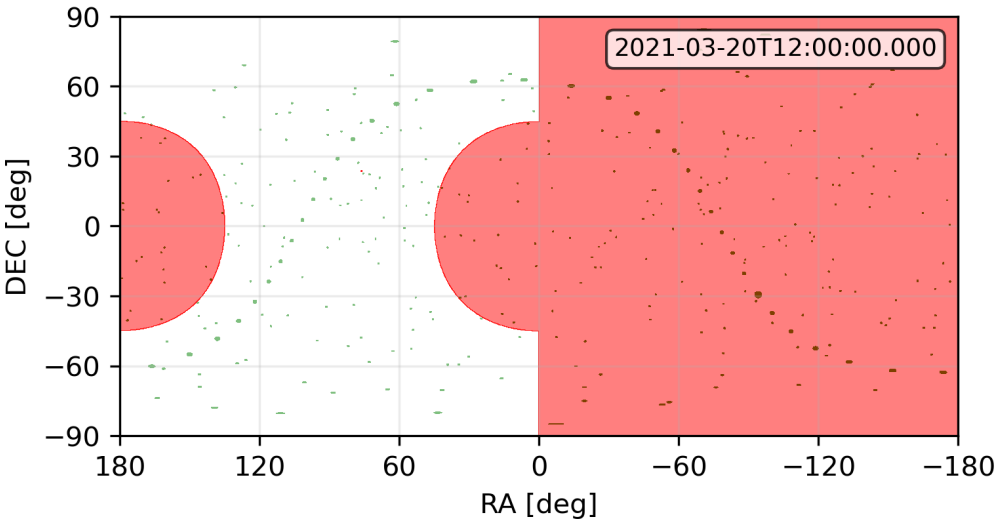


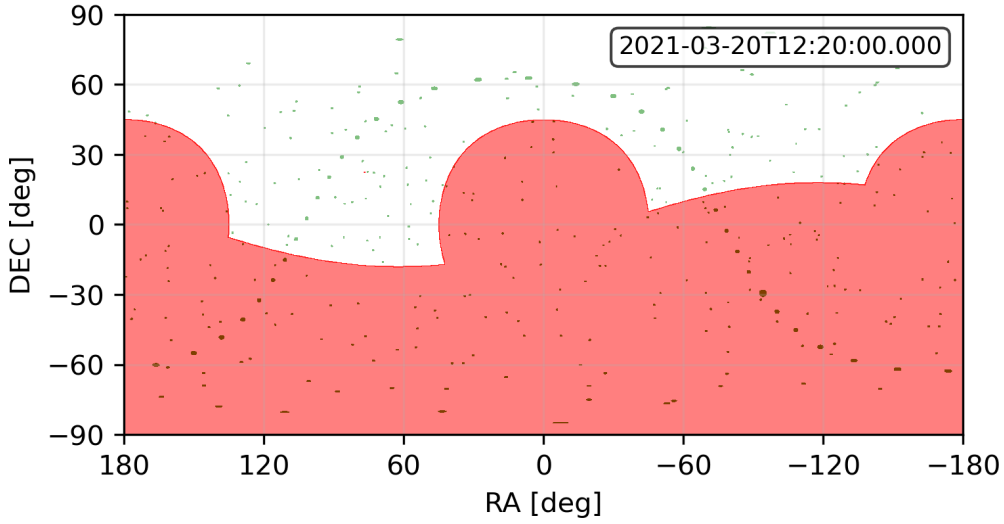
Figure 2.7: Example of a structured mesh overlaid by dashed regions. The value of a mesh point is true if it lies within one of the regions as illustrated by the coloured dots.

These raster images are plotted using Matplotlib [17] as a filled contour plot. This method was chosen as the raster generation was vertex-centred, which does not correspond to the pixel-centred values expected by the typical image plotting method. Furthermore, this method provides greater future flexibility for introducing alternative image projections: for example, by using packages such as Cartopy [18] which natively support filled contour plots with transformations between projections, requiring only minor implementation changes.

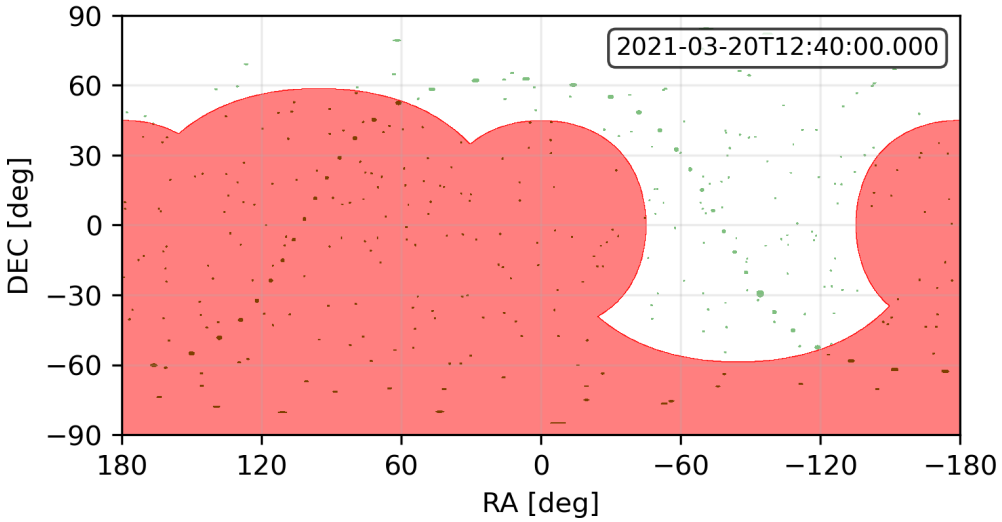
Examples of the images generated by the visualisation module are presented in Figure 2.8. The “soft” radius constraints of the Sun are visible as two circles with an angular radius of 45°, centred at (0,0) and (180,0). These represent the pointing restrictions towards and away from the Sun respectively, as previously discussed in Section 2.2.2. The orbit of the spacecraft can be seen by the motion of the Earth which appears to move below the spacecraft before rising on the opposite side. This behaviour was expected as the visualisation begins at the ascending node of the spacecraft’s orbit, where it passes from the southern to the northern hemisphere.



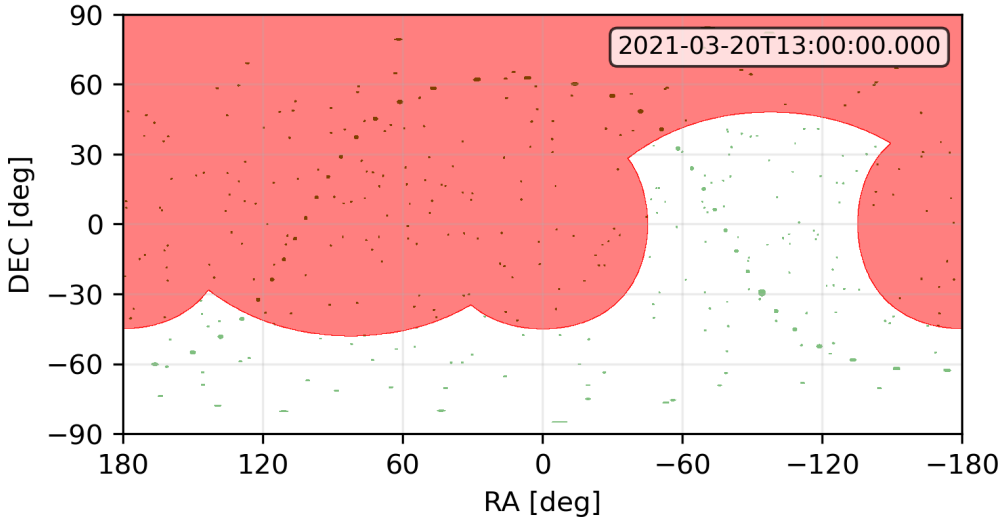
(a) 12:00 UTC



(b) 12:20 UTC



(c) 12:40 UTC



(d) 13:00 UTC

Figure 2.8: Examples of sky views generated by the visualisation module at various times during the vernal equinox. All of the solar bodies, the Galactic centre, the Galactic mid-plane at 10° intervals, and selected M-type stars are included in this visualisation. Targets are shown in green, and constrained regions resulting from the solar bodies in red, therefore targets in the white regions are visible. Unlike the markers in Figure 1.2, this visualisation is to scale, and subject to distortion at high declinations.

2.3 Preliminary Conclusions

In conclusion, an initial framework has been implemented which propagates both the spacecraft and solar bodies, and calculates whether specified targets are visible, subject to both “hard” and “soft” limits. The current implementation satisfies the first branch of the three main objectives outlined in Section 1.3 as manual scheduling is now possible with ASSAM. While automatic scheduling is not yet implemented, investigations into scheduling methods are underway, and the software is ready for this functionality to be added to address the second branch. Although the third objective, relating to computational performance, cannot be quantified, effort was made during initial development to ensure that optimisation of the software could be performed in the future. Several sections of the code, particularly within the propagation and visualisation modules, have already been parallelised, with many others ready to be optimised.

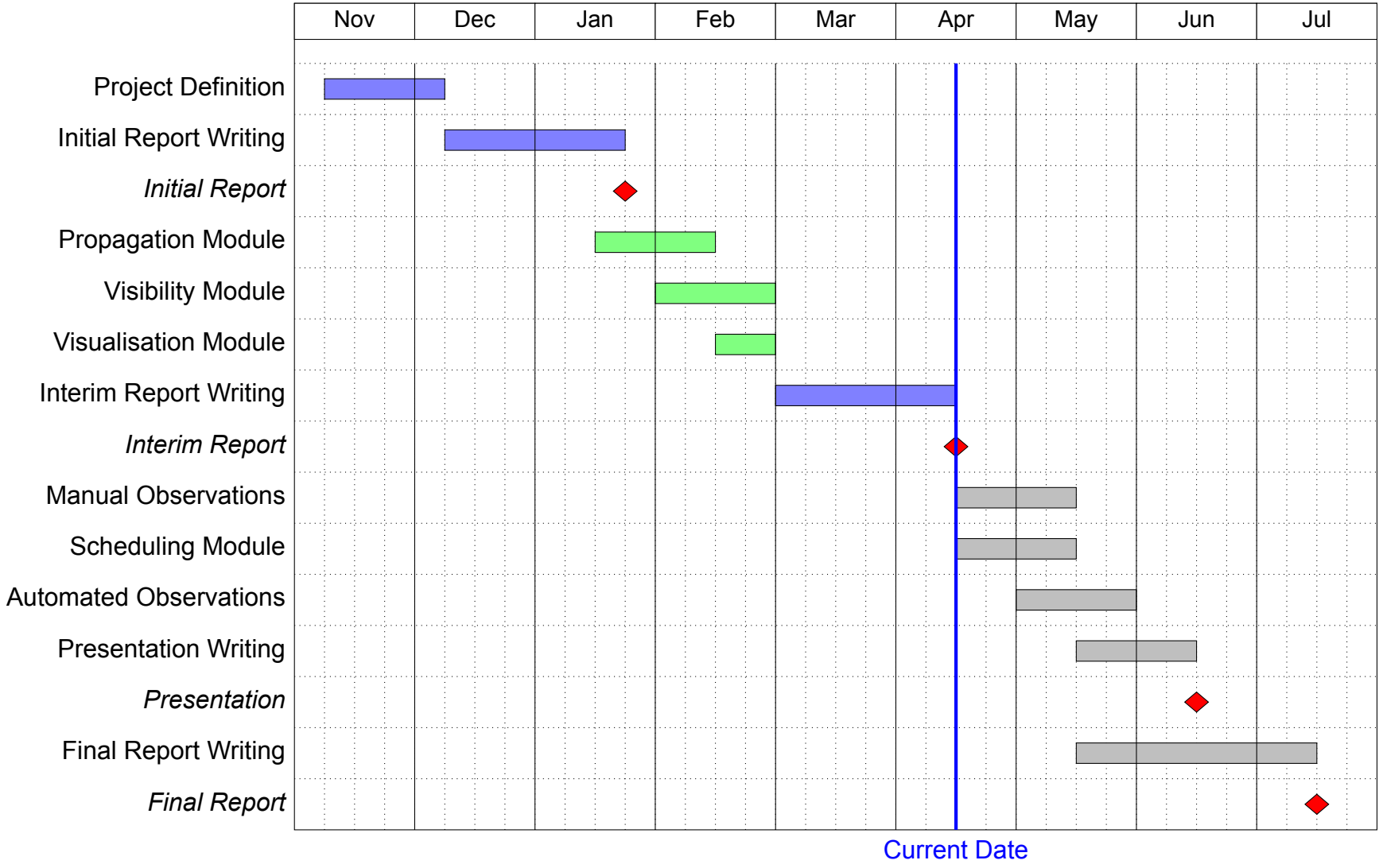
3 FUTURE WORK

A provisional plan for the contents of the final report is presented in Listing 3.1, broadly following the suggested report structure [19]. An updated project timeline is presented in Table 3.1 with changes including updated project milestone dates and the inclusion of the new visualisation module. Future work will concentrate on the implementation of the scheduling module, and creation of the final deliverables.

Listing 3.1: Provisional final report contents.

```
Title page
Abstract
Contents
    List of Sections
    List of Figures
    List of Tables
    List of Acronyms
Introduction
    Project objectives
Background
    JASMINE
    Sky Survey Software
    Schedule Optimisation
Methods
    Software Modules
    Verification/Validation
Results
Discussion
    Strengths/Limitations of the Software
    Future Work
Conclusions
References
Appendices
```

Table 3.1: Project timeline. Redrawn and updated from [4].



REFERENCES

[1] N. Gouda, 'JASMINE,' *Scholarpedia*, vol. 6, no. 10, p. 12 021, 2011. [Online] Available at: http://www.scholarpedia.org/article/JASMINE [Accessed 8th March 2021].

[2] T. Yamada, *Future Space Science Program of JAXA*, JAXA, 12th December 2019. [Online] Available at: https://www.nao.ac.jp/for-researchers/naoj-symposium2019/presentations/09.NAOJ_sympo20191212_Yamada.pdf [Accessed 8th March 2021].

[3] D. Kawata *et al.*, 'JASMINE: Near-Infrared Astrometry and Time Series Photometry Science,' *Astronomical Society of Japan*, in preparation.

[4] M. Hallgarten La Casta, *Automated Sky Survey Analysis Methods: Initial Report*, Internal Access Only, University College London, 2021.

[5] S. Nishiyama *et al.*, 'The Interstellar Extinction Law toward the Galactic Center. II. V, J, H, and K_s Bands,' *The Astrophysical Journal*, vol. 680, no. 2, pp. 1174–1179, June 2008.

[6] D. Kawata, Private Communication, 9th March 2021.

[7] M. D. Johnston, 'Spike: AI scheduling for NASA's Hubble Space Telescope,' in *Sixth Conference on Artificial Intelligence for Applications*, vol. 1, IEEE Computer Society, May 1990, pp. 184–190.

[8] D. S. Adler, D. K. Taylor and A. P. Patterson, 'Twelve years of planning and scheduling the Hubble Space Telescope: process improvements and the related observing efficiency gains,' in *Observatory Operations to Optimize Scientific Return III*, International Society for Optics and Photonics, vol. 4844, SPIE, 2002, pp. 111–121.

[9] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. CreateSpace, 2009, ISBN: 1441412697.

[10] Astropy Collaboration *et al.*, 'Astropy: A community Python package for astronomy,' *Astronomy & Astrophysics*, vol. 558, A33, October 2013.

[11] Astropy Collaboration *et al.*, 'The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package,' *The Astronomical Journal*, vol. 156, no. 3, p. 123, September 2018.

[12] The GMAT Development Team, *General Mission Analysis Tool (GMAT) User Guide*, NASA Goddard Space Flight Center, Greenbelt, Maryland, 2020.

[13] D. A. Vallado, *Fundamentals of Astrodynamics and Applications*, 4th ed. Microcosm Press, 2013, ISBN: 9781881883180.

[14] The Astropy Developers, *Astropy Documentation*, 2020. [Online] Available at: https://docs.astropy.org/en/stable/index.html [Accessed 24th February 2021].

[15] B. C. Rhodes, *PyEphem: Astronomical Ephemeris for Python*, 2020. [Online] Available at: https://pypi.org/project/jplephem/ [Accessed 28th February 2021].

[16] The Astropy Developers. (16th February 2021). ‘Astropy Regions Documentation,’ [Online] Available at: https://astropy-regions.readthedocs.io/en/latest/ [Accessed 31st March 2021].

[17] J. D. Hunter, ‘Matplotlib: A 2D graphics environment,’ *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[18] *Cartopy: A cartographic python library with a matplotlib interface*, Met Office, Exeter, Devon, 2010–2015. [Online] Available at: http://scitools.org.uk/cartopy [Accessed 24th February 2021].

[19] S. Matthews, *Individual research projects*, Internal Access Only, University College London, 2021.