OPTICAL SENSOR UNCERTAINTIES AND VARIABLE REPOSITIONING TIMES IN THE SINGLE AND MULTI-SENSOR TASKING PROBLEM

A Thesis

Submitted to the Faculty

of

Purdue University

by

Michael J. Rose

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

December 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF THESIS APPROVAL

Dr. Carolin E. Frueh, Chair

School of Aeronautics and Astronautics

Dr. Daniel A. DeLaurentis

School of Aeronautics and Astronautics

Dr. Bryan D. Little

Air Force Institute of Technology, Department Aeronautics & Astronautics

Approved by:

Dr. Greg Blaisdell

Associate Head of Gambaro Graduate Program of Aeronautics and Astronautics

This work is dedicated to my family and friends. I would have not been able to complete this work without their love and support.

ACKNOWLEDGMENTS

First, I want to acknowledge the support received in developing this thesis. I would like to acknowledge the support of this work via the Air Force Grant: AF16-AT05. Without funding, this research would have not been accomplished. I would like to thank and acknowledge the collaboration with Dr. Peter Zimmer and Dr. Mark Ackermann at J.T. McGraw and Associates, for their research and data that they provided for this research. Special thanks go to Dr. Bryan Little AFIT Ohio, who I owe much for in this thesis from his previous work. Finally, I would like to thank Nathan Houtz for collaborating with me by providing valuable data used in this thesis.

I would like to thank my advisor, Dr. Carolin Frueh. Her invaluable support in developing this research is difficult to describe in words, but this thesis would not be possible without her. She also provided me with the background knowledge in this field and connected me to other exceptional students working on incredible research. I would also like to thank the Aeronautical and Astronautical Engineering department and faculty. The support and knowledge they provided me will extend far beyond just helping me with this thesis, and will stay with me as I move into the workforce.

Finally, I would like to thank my family and friends. Though they did not understand what I was saying when I described my research, they always listened and made me feel heard. They made all the late caffeinated nights and long weekends working on this thesis worth it.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABSTRACT

Rose, Michael J. M.S., Purdue University, December 2020. Optical Sensor Uncertainties and Variable Repositioning Times In the Single and Multi-Sensor Tasking Problem. Major Professor: Carolin E. Frueh.

As the number of Resident Space Objects around Earth continues to increase, the need for an optimal sensor tasking strategy, specifically with Ground-Based Optical sensors, continues to be of great importance. This thesis focuses on the single and multi-sensor tasking problem with realistic optical sensor modeling for the observation of objects in the Geosynchronous Earth Orbit regime. In this work, sensor tasking refers to assigning the specific observation times and viewing directions of a single or multi-sensor framework to either survey for or track new or existing objects. For this work specifically, the sensor tasking problem will seek to maximize the total number of Geosynchronous Earth Orbiting objects to be observed from a catalog of existing objects with a single and multi optical sensor tasking framework. This research focuses on the physical assumptions and limitations on an optical sensor, and how these assumptions affect the single and multi-sensor tasking scenario. First, the concept of the probability of detection of a resident space object is calculated based on the viewing geometry of the resident space object. Then, this probability of detection is compared to the system that avoids the computational process by implementing a classical heuristic minimum elevation constraint to an electro-optical charged coupled optical sensor. It is shown that in the single and multi-sensor tasking scenario if the probability of detection is not considered in the sensor tasking framework, then a rigid elevation constraint of around 25°-35° is recommended for tasking Geosynchronous objects. Secondly, the topic of complete geo-coverage within a single night is explored. A sensor network proposed by Ackermann et al. (2018) is studied with and without the probability of detection considerations, and with and without uncertainties in the

resident space objects' states. (then what you have). For the multi-sensor system, it is shown that with the assumed covariance model for this work, the framework developed by Ackermann et al. (2018) does not meet the design requirements for the cataloged Geosynchronous objects from March 19^{th}, 2019. Finally, the concept of a variable repositioning time for the slewing of the ground-based sensors is introduced and compared to a constant repositioning time model. A model for the variable repositioning time is derived from data retrieved from the Purdue Optical Ground Station. This model is applied to a single sensor scenario. Optimizers are developed using the two repositioning time functions derived in this work. It is shown that the constant repositioning models that are greater than the maximum repositioning time produce results close to the variable repositioning solution. When the optimizers are tested, it is shown that there is a small increase in performance only when the maximum repositioning time is significant.

1. INTRODUCTION

Space Situational Awareness, or SSA, is a large field of research and operational use. While there is no common definition for SSA, The U.S. Strategic Command defines SSA as follows:

> "SSA is the requisite foundational, current, and predictive knowledge and characterization of space objects and the OE upon which space operations depend – including physical, virtual, information and human dimensions – as well as all factors, activities, and events of all entities, conducting, or preparing to conduct, space operations. Space surveillance assets include a mix of space-based and earth-based sensors." [17]

To conduct present and future missions in the Earth regime, the environment around Earth needs to be monitored to ensure safe operating conditions, and enable sustainable use of space. The total number of objects in the Earth regime, also known as Resident Space Objects (RSOs), has continued to increase, which complicates SSA. These RSOs can be defined by several different categorizations and classifications, the most notable two are operational satellites and debris. Space debris is a category classification of RSOs that are "non-functional with no reasonable expectation of assuming or resuming its intended function" [18]. Such objects include dead satellites, upper rocket stages, mission-related objects, fragmentation debris (which also vary in origin), etc. Most of the debris objects are tiny, less than ten centimeters in size, and it is estimated that there are over 100,000 objects that are unable to be tracked with current observation methods [19, 67–69]. The dynamic states of both the operational satellites, as well as debris, small and large, need to be understood to ensure safe operating conditions for current and future missions and for accurate SSA analysis to be conducted [1,66]. The states of the RSOs are generally estimated using sensor measurements, whether ground-based sensors or space-based sensors, such as the one operated by the Space/Air Force, that make measurements using a variety of

methods [1, 8, 71]. Currently, a prominent way ground-based electro-optical sensors record images of RSOs is with a Charged-Coupled Device, or CCD [31]. This work assumes that the sensors modeled and studied are electro-optical sensors with CCDs. These Ground-based sensors are limited by their location, which limits which RSOs can be in their Field of Regard (FOR), but also influences the observation conditions. Camera specifications affect the performance of the specific sensor depending on the quality of equipment. Attenuation effects from the atmosphere also limit the performance of the sensor, which affects the light coming in from the RSO to the optics of the sensor. These limitations reduce the chances of a sensor detecting an RSO, even when it is present in its FOR for a given time, requiring more sensors to be constructed to account for the increasing population of RSOs around Earth [1, 8, 71].

Several space agencies, countries, and private industries have sensor networks that track RSOs around Earth. Such entities include SMARTnet German Aerospace Center DLR, ExoAnalytics optical sensor catalog, LeoLabs radar sensor catalog, and many more [1, 7]. Other entities also have catalogs of known and currently tracked RSOs. The Vimpel Catalog and the U.S. Strategic Command's (USSTRATCOM) Two-Line Element (TLE) catalog are such examples. The most commonly known, and the one used in this thesis, one being from the USSTRATCOM [19].

The objects from the USSTRATCOM catalog are tracked by the Space Surveillance Network (SSN), which uses a combination of ground-based radar and optical sensors [71]. The size of an object that is able to be detected depends on how far away the object is from the ground-based sensor, or its orbital regime [1, 8]. Currently, for Low Earth Orbiting (LEO) objects, or objects with an altitude above the Earth's surface of around 2,000 kilometers or less, the SSN can track an RSO that has a ten centimeters or larger reflective surface relative to the sensor. For Geosynchronous Earth Orbiting (GEO) objects or objects with an altitude between 35,586 to 35,986 kilometers, the SSN can only track an RSO that has around one meter or larger reflective surface relative to the sensor [71]. It is estimated that if sensors could

track objects that are one centimeter or larger the number of cataloged objects would increase from about 20,000 to over 100,000 [1, 19, 69, 71].

The ground-based sensors are tasked with both tracking and maintaining cataloged objects, as well as surveying for new objects to be added to the catalog [1,19,71]. When an object is observed, its position and velocity state can be determined (assuming there is a priori information on its state), within a range of uncertainty. This uncertainty is due to several factors, some of which are imperfect measurement conditions, noise in measurements, and incomplete dynamics models of the RSO. [50]. If the objects are not regularly tracked and observed, their positional uncertainty will grow large enough such that the positional uncertainty is much larger than the field of view of the sensor, so it cannot be securely found again. Therefore, the object's state cannot be updated. The object's location cannot be determined then, so the object is lost [1, 50, 70]. If the object is tracked regularly, the uncertainty in the state of the object can be reduced and maintained to enable successful re-detection and re-discovery of the object. For successful collision predictions, however, with notice of several days in advance even tighter uncertainty bounds have to be established. USSTRATCOM's catalog of objects currently only has information about the mean state of the object, but no information of its uncertainty. This leaves a gap in information, and requires independent observations outside of the USSTRATCOM catalog [1, 19, 20, 22, 50]. Furthermore, the catalog is also not complete. Schildknecht et al. (1999) also showed that anywhere between 5%-20% of the objects detected in their work were not cataloged in the USSTRATCOM catalog in the GEO regime alone [38]. With the continual need to detect and track RSOs coming from a limited number of ground-based sensors, this creates the need for an efficient and effective sensor tasking scheme [1, 6, 24, 67].

Sensor tasking is a well-investigated topic [27,39–48,72]. Schildknecht et al. (1999) and Alfano et al. (2000) developed survey strategies based on the current catalog population without explicit a priori information on particular space objects [38, 48]. Follow-up observations are based on a current catalog that requires being updated, as

positional uncertainties grow over time. This can be formulated as an optimization problem: Hill et al. (2010) used a covariance-based optical sensor tasking approach, Sunberg et al. (2016) leveraged the advantages of an information-based sensor tasking [45,46]. Jaunzemis et al. (2016) used an evidence-based tasking scheme to understand how decision-makers in optimizers pose criteria [42]. Linares et al. (2017) and Little et al. (2018) investigated machine learning algorithms for solving the sensor tasking optimization [27,44]. Frueh et al. (2018) developed a scheme of sensor tasking fusing survey and follow-up in a single framework based on weighted viewing directions [24]. With the framework developed by Frueh et al. (2018), Little and Frueh (2019) investigated classical and AI-based optimizers for an efficient solution to the sensor tasking scenario [6]. Little and Frueh (2019) also presented a computationally efficient method for the single and multi-sensor tasking scenario in the absence of measurement feedback for both ground-based and space-based sensors [6]. In the multi-sensor tasking scenario, the location of each sensor also needs to be considered. Ackermann et al. (2018) optimizes the total number and location of ground-based sensors to view all GEO objects consistently every day, including local weather effects and light pollution at the sensor locations [28].

The CCD images can be used in a variety of applications in SSA [25, 29, 32, 34–36, 38]. Friedman and Frueh (2018) analyzed the observability of RSOs to determine shape characteristics based on CCD imaging [29]. Virtanen et al. (2016) used a segmentation method to determine what type of object is being observed (debris, operational satellite, etc.) [32]. In an ideal world, if there were no errors or noises in the CCD image, the true measurement of the RSO can be determined. In the real world, however, there is measurement noise, background light sources, and other effects that make determining the measurement state of the RSO in the CCD an important topic of research [1, 31, 37]. Schildknecht et al. (2018) used Gaussian Convolution to distinguish objects-of-interest from other objects in the CCD image, along with background noise [38]. Hagen and Dereniak (2008) studied 2D Gaussian estimation for determining the mean and covariance of an object's location in a CCD image

[35]. Sanson and Frueh (2018) assume that the brightest pixel is more significant in determining detection than any other part of the image [25]. They used this brightest pixel location to develop a probability of detection measure, useful in simulation and sensor tasking planning.

When considering the single and multi-sensor tasking problem, the time it takes for the sensor to move from one viewing direction to another, or the sensor's repositioning time (or slew time), changes depending on how far the repositioning distance is. Modeling this effect as a variable repositioning time however can be computationally more significant than just considering the repositioning time to be constant, though the model will more accurate. Because of the trade-off between computational time and simulation accuracy, choosing to model a variable repositioning time is not a trivial task [1, 6, 73, 74]. Frueh (2017) used two separate constant repositioning times depending on if the sensor was currently taking a stripe observation, or if the sensor was repositioning to a new stripe observation location [41]. Herzog (2013) used a pair of constant repositioning times (with some errors in the repositioning times considered) for specific stripe observation scenarios [70]. Daniel (2018) considered a constant slew rate when developing a modular neural network design for ground-based sensors [72].

When dealing with the loss of light near the local horizon in the sensor tasking problem, generally a heuristic elevation constraint is commonly used by sensor operators. However, having a rigid elevation constraint limits the FOR of a sensor, potentially limiting the performance. When attenuation effects and the probability of detection of an object-of-interest are included in the sensor tasking algorithm, a rigid elevation constraint may not be needed. In this work, the concept of the probability of detection of a resident space object is calculated based on the viewing geometry of the resident space object. Then, this probability of detection is compared to the system that avoids the computational process by implementing a classical heuristic minimum elevation constraint to an electro-optical charged coupled optical sensor. This is studied on the single sensor scenario and then analyzed in the multi-sensor

tasking scenario using the sensor system developed Ackermann et al. (2018) [28]. The multi-sensor system is studied with and without calculation of the probability of detection compared to the optical sensors that have a rigid elevation constraint and without a rigid elevation constraint. This first part assumes that the object's mean position and velocity state is the true state, to understand the relationship between the probability of detection a heuristic rigid elevation constraint in the sensor tasking problem. The uncertainty of the location of the RSOs is then considered in the full multi-sensor tasking scenario and the results are compared.

The conclusions that will be drawn above assume a constant repositioning time, but when a sensor repositions in the real world, the assumption of a constant repositioning time is not true. An analysis needs to be done on the validity of this assumption, and if the sensor tasking scenario does include a variable repositioning time, how can this be optimized to benefit the sensor tasking scenario. A model for the variable repositioning time is derived in this thesis from data retrieved from the Purdue Optical Ground Station. This model is then applied to a single sensor scenario. Optimizers are developed using the two repositioning time functions derived in this work to analyze the effects of considering the variable repositioning time in the sensor tasking problem.

1.1 Research Objectives

The formulation built upon by Little and Frueh (2019) is further developed to analyze various problems in the sensor tasking framework [6]. Building upon the sensor tasking algorithm developed by Frueh et al. (2018), this paper seeks to study the following objectives [24]:

1. Light that is passing through the atmosphere is attenuated. What is the effect of the probability of detection on the sensor tasking, especially when compared to rigid elevation constraints?

2. Can the framework from Frueh et al. (2018) be expanded towards complete coverage of the GEO regime [6, 24]? Are the assumptions of Ackermann et al. (2018) correct [28]?

3. How much of a benefit is there to modeling the computationally expensive variable repositioning time in the sensor tasking framework to achieve more accurate simulation models? If incorporated, can an optimizing strategy be developed to improve the performance of the sensor tasking problem?

1.2 Assumptions

To fully test the research objectives presented in section 1.1, several assumptions need to be made. Object-specific and sensor-specific assumptions are presented below [6, 28]:

1. Object Specific:

   (a) All objects are modeled as a one meter diameter sphere, with a constant diffusion coefficient [16].

   (b) Only Geosynchronous Earth Orbiting (GEO) objects are considered in the sensor tasking problem.

   (c) Orbital uncertainties are modeled initially as Gaussian.

   (d) Simulation times are assumed short enough, so the orbital uncertainties are assumed to remain Gaussian

   (e) All objects begin with an equal need to be observed.

   (f) All objects are correctly tagged with their correlated catalog object.

2. Sensor Specific:

   (a) Only optical sensors are considered.

   (b) Immediate feedback between each sensor is assumed.

(c) Sensor positions are perfectly known at any given time.

(d) All sensors in the multi-sensor tasking scenarios are assumed to have the same characteristics.

(e) Sensor pointing directions are perfectly known (no pointing errors).

Additional assumptions not presented above will be expressed as they arise in the paper.

1.3 Outline of Thesis

The organization of this thesis is as follows: Chapter 2 discusses the motion and driving dynamics of an object in the near-Earth regime. Conservative and non-conservative forces are presented and how the forces affect the motion of an object is presented. An overview of how optical sensors make measurements is also presented.

Chapter 3 introduces the main sensor tasking optimization equation that is used throughout the paper. Each part of this equation is analyzed and the effects of each are shown in the sensor tasking scenario. Further discussion on optimizers, FOR discretization, and immediate feedback are presented.

Chapter 4 introduces the uncertainties that arise in optical sensors, specifically in Charged-Coupled Devices, and how attenuation effects play a factor in the probability of detection of an object. The relationship between the probability of detection of an object and elevation is analyzed.

Chapter 5 presents a common assumption in sensor tasking simulations: a constant repositioning time. The constant repositioning time formulation is introduced in the sensor tasking problem, assumptions and limitations are discussed, and a new, variable repositioning time formulation is presented.

Chapter 6 introduces the computational challenges of the sensor tasking problem. Trade-offs between computational efficiency and simulation accuracy are shown in the single and multi-sensor tasking scenarios.

Chapter 7 analyzes several different sensor tasking scenarios. The relationship between the probability of detection of an object and elevation constraint on the optical sensor is analyzed in the single and multi-sensor tasking problem. The assumption of a constant repositioning time is studied, and an optimizer to consider variable repositioning time is presented.

Chapter 8 summarizes the thesis, and future research concepts are discussed.

2. BACKGROUND

To develop accurate computational models for the single and multi-sensor tasking scenarios, and understanding of the dynamical environment in space, as well as where and how the sensor's take measurements is vital. This chapter discusses the orbital motion of these resident space objects (RSOs), the method these objects are inputted into software for study, coordinate frames of both the RSOs and the observer, and how optical sensors take measurements.

2.1 Orbital Motion

To begin, the basic form of gravity derived from Newton for two objects is presented. Shown in Fig. 2.1, these bodies have an attracting pull on each other [23].

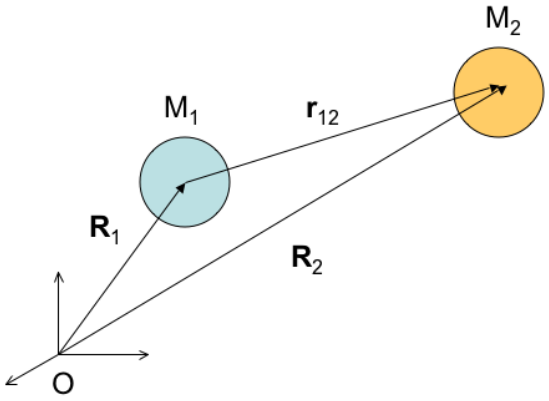


Figure 2.1. The two-body problem shown in the inertial frame [23].

Deriving the equations from the inertial frame using Newton's laws, one can arrive at the following equations for the equations of motion for both object 1 and object 2 [1–3]:

$$m_1\ddot{\overline{r}}_1 = -G\frac{m_1 m_2}{|\,\overline{r}_{12}\,|^3}\overline{r}_{12} \tag{2.1}$$

$$m_2\ddot{\overline{r}}_2 = -G\frac{m_1 m_2}{|\,\overline{r}_{21}\,|^3}\overline{r}_{21} \tag{2.2}$$

$$m_1\ddot{\overline{r}}_1 + m_2\ddot{\overline{r}}_2 = 0 \tag{2.3}$$

Where m_1 and m_2 are the masses of object 1 and object 2 respectively. G is the universal gravitational constant, and $\overline{r}_{12}$ is the position vector between each object, with $\overline{r}_{12} = -\overline{r}_{21}$. $\ddot{\overline{r}}_1$ and $\ddot{\overline{r}}_2$ are the accelerations of each object with respect to an inertial frame O, shown in Fig. 2.1 [1, 23].

These equations cannot be solved analytically as they are right now. If the arbitrary inertial frame O is moved to the center of mass (the center of mass for the system is moving at a constant velocity, so it is inertial [1]), then Eqns. (2.1)–(2.3) can be simplified and solved analytically. Additionally, the energy and momentum of the system are also conserved, which can be proved using the "Ten Known Integrals" of motion [1]. It is assumed that the secondary mass is significantly smaller than the primaries, which leads to the following equation:

$$\ddot{\overline{r}} = -\frac{\mu}{r^3}\overline{r} \tag{2.4}$$

$$\mu = Gm_{\oplus} \tag{2.5}$$

Now, because the system is referenced as the relative motion between the two objects, $\overline{r}$ represents the vector position from the primary body to the secondary body (in this case, from the Earth to the object-of-interest) in Eqn. (2.4). $\ddot{\overline{r}}$ is the relative acceleration between the objects, and μ is the Primary object's gravitational parameter. This work assumes the Primary object to be Earth, so Eqn. (2.5) is derived using Earth's mass [1, 3].

The classical two-body problem is studied heavily, due to its analytical solution, but also is too simple when trying to study the motion of RSOs. Two other conser-

vative forces are often studied: n body gravitational forces, and a nonspherical shape of those bodies. These conservative forces are often studied because of the large effects they have on the object-of-interest's orbital state. n body gravitational forces generally have a larger effect on RSOs that are farther from the Earth (such as GEO objects), while a nonspherical shape model of the Earth has a larger effect on RSOs that are closer to Earth (such as LEO objects) [1].

When adding a third or more gravitational body to the system (such as the moon and/or sun), the formulation becomes more complex, and there is no analytical solution to the motion of the bodies. The general way of studying the n body problem is by formulating the equation of motion for the classical two-body problem, and then adding the extra acceleration from the other attracting bodies (there are more clever ways of studying the three-body problem, but those will not be focused on in this paper) [1, 2, 10]. The general equation of motion for the motion of the object of interest is as follows:

$$\ddot{\overline{r}} = -\frac{\mu}{r^3}\overline{r} - \sum_{i=1}^{i=n} \mu_i \left(\frac{\overline{r} - \overline{r}_i}{\mid \overline{r} - \overline{r}_i \mid^3} + \frac{\overline{r}_i}{\overline{r}_i^3} \right) \tag{2.6}$$

If the summation term in Eqn. (2.6) is ignored, then the classical two-body problem is arrived at, shown in Eqn. (2.4). μ_i is the gravitational parameter of object i, and $\overline{r}_i$ is the relative distance from object i to the object-of-interest. In the Earth regime, generally, the sun and the moon are taken into account as other perturbing bodies using Eqn. (2.6).

The classical two-body problem, and Eqn. (2.6) assumes that the attracting bodies are spherical and have uniform density throughout, which is not the case. The equation of motion for the object-of-interest from a single point mass attracting body can be written as either in Eqn. (2.4), or as follows [1]:

$$\ddot{\overline{r}} = \nabla U \tag{2.7}$$

$$U = \frac{\mu}{r} \tag{2.8}$$

Where U is the potential function. For the classical two-body problem, U simplifies to Eqn. (2.8). If it is then assumed that the attractive body is not spherical and/or does not have a uniform density, then that body can be considered to be a collection of many different point mass objects packed close together. This, in equation form, is written as the following:

$$U = G \int_{body} \frac{1}{\rho_q} dm_q \tag{2.9}$$

Here m_q is the infinitesimal element of the mass of the Earth that is acting on the satellite at a distance of ρ_q [1, 10]. Using spherical geometry and the Legendre Polynomials, the potential function of all the point masses (or the potential of the single, nonspherical, variable density) can be written as below:

$$U = \frac{\mu}{r} + \frac{\mu}{r} \sum_{l=2}^{l=\infty} \sum_{m=0}^{m=l} \left(\frac{R_\oplus}{r} \right)^l P_{l,m}[\sin \phi_{gc,sat}](C_{l,m} \cos(m\lambda_{sat}) + S_{l,m} \sin(m\lambda_{sat})) \tag{2.10}$$

Eqn. (2.10) accounts for the zonal and tesseral harmonics of Earth's Gravity. If $l = m = 0$, then the classical two-body equation is arrived at, as shown in Eqn. (2.8). $R_\oplus$ is Earth's mean radius, with $\phi_{gc,sat}$, $P_{l,m}$ is the Legendre Polynomial of the l and m element, and λ_{sat} being the geocentric latitude and longitude of the object-of-interest respectively [1, 2, 10].

Other forces that affect the motion of the RSO are not conservative. The two most considered non-conservative forces in the Earth system are drag and solar radiation pressure. They are considered the most in the Earth regime as they are the largest non-conservative forces that affect the state of the RSO. These non-conservative forces change the overall momentum of the system and depend on the size, shape, orientation, and location in space of the RSO.

For RSOs that are close to Earth, or in low Earth orbit (LEO), the primary non-conservative force is drag. Drag on the RSO will decrease the overall energy of the

orbit, causing the semimajor axis and eccentricity to change. If no orbital maneuver is done, the RSO will eventually re-enter Earth's atmosphere due to drag [1].

$$\overline{a}_{drag} = -\frac{C}{2}\rho(r)\frac{A_{obj}}{m}v'^{2}\frac{\overline{v}'}{\mid v' \mid} \tag{2.11}$$

Eqn. (2.11) shows how drag affects the motion of the RSO, and how drag is affected by the size, shape, orientation, and location of the RSO. $\overline{a}_{drag}$ is the acceleration term acting on the object-of-interest due to Earth's atmosphere. C is the coefficient of drag for the object-of-interest. For Spherical objects, $C = 2$, and in general $2 \leq C \leq 2.5$. A_{obj} is the relative area of the object-of-interest to Earth's atmosphere, m is the mass of the object, and $\overline{v}'$ is the relative velocity of the object and Earth's atmosphere the ratio A_{obj}/m is usually defined as the object's area-to-mass ratio, or AMR [10].

When the RSO is located farther away from the Earth, other non-conservative forces become more dominant than drag. For RSOs in the GEO regime, solar radiation pressure (SRP) is more dominant and considered more than drag. SRP changes the energy of the system over shorter periods of time, and in the GEO regime generally changes the inclination to move between $\pm 15°$ [63].

$$\overline{a}_{srp,sphere} = -\frac{A_{obj}}{m}\frac{E}{c}\frac{AU^2}{\mid \overline{x} - \overline{x}_{sun} \mid^2}\tilde{C}\hat{S} \tag{2.12}$$

$\overline{a}_{srp,sphere}$ is the acceleration term acting on the object-of-interest due to the solar radiation pressure of the sun. A_{obj}/m is the AMR of the object with respect to the sun direction, $\hat{S}$. E is the solar constant at Earth's surface, and c is the speed of light. AU is the astronomical unit distance constant, $\mid \overline{x} - \overline{x}_{sun} \mid$ is the distance from the object to the sun, and $\tilde{C} = (\frac{1}{4} + \frac{1}{9}C_d)$ for spherical objects [1–3, 10].

As more and more terms are added in trying to model the motion of the object-of-interest, computational considerations need to be addressed. For this research, only the classical two-body problem is considered. The motion of the RSOs that is simulated in this research is generally less than a day, so the perturbing effects

beyond the classical two-body problem are small. Over longer periods of time, the effects beyond the two-body problem become larger and more impactful. Keeping the simulation to just the two-body problem significantly reduces computational times, and because this research studies the multi-sensor tasking problem, it is important to balance computational efficiency and simulation model accuracy [1, 6].

2.2 Two-Line Element Catalog

This work uses the states of objects collected in the publicly available Two-Line Element (TLE), published by the United States Strategic Command (USSTRATCOM), as the initial state to determine their orbits in the future times in the simulation. The TLE catalog is maintained and updated by the 18th Space Control Squadron (as of December 2020), and contains information on when the object was observed, which object is it, as well as its orbit [1, 19, 20].

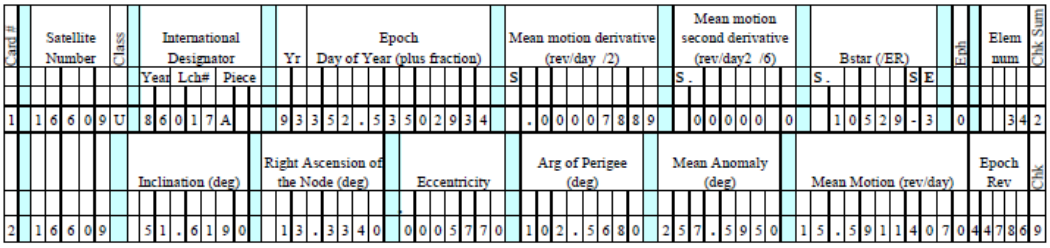


Figure 2.2. An example Two-Line Element with descriptions on each element in the TLE (from Ref. [1], Figure 2-19 page 107).

Fig. 2.2 shows an example TLE, along with its format. The first five digits (ignoring the card number) shows the Satellite Number (formatted in order of when the object was first observed), followed by if the TLE is unclassified (U) or not. The International Designator is another number that identifies the object but is based on the launch of the object. The first two numbers are the launch year (if it's greater than or equal to 57, it is assumed it's in the 1900s, if it's less than 57, it's assumed its in the 2000s). The next three number indicate the launch number for that year,

and finally, the last part is the "piece" information, which is what part of the launch was this object (example "pieces" are payload, debris, rocket bodies, etc.) [19]. The Epoch provides information on when the object was observed and its orbit updated. The year is the year it was observed, and the Day of the Year is the date in days. The Epoch information is in Universal Time Coordinated [1, 19]. The mean motion derivative and second mean motion derivative are used to propagate the states to future times, using models such as SGP and SDP. According to Vallado et al. (2006), these values are not used in the SGP4 model, which this research uses [20]. The Bstar term is a "drag-like" coefficient used in SGP4, and the element number is the number of times a new data set for the object is generated, though it is not strictly followed [7, 19, 20].

The second line describes the two-body assumed orbital motion of the object, in terms of keplerian orbital elements [1, 19]. The angles in the keplerian orbital set described in the TLE (inclination, right ascension of the ascending node, the argument of perigee, and mean anomaly) are all given in degrees [1, 19]. For the eccentricity of the orbit, it is assumed that all objects are Earth-orbiting objects ($e < 1$), so the decimal place is left off on the TLE. The mean motion is given in revs/day, where mean motion is related to the semimajor axis of an orbit by $a = \sqrt[3]{\mu/n^2}$. Finally, the epoch revolutions are counted from the ascending node of the orbit onward [7,19,20].

The propagation of the orbit in TLEs from one epoch to another is done using different models, such as SGP/SDP, SGP4, etc. This work uses the SGP4 model. For any orbit determination, the true state of the orbit is unknown, and generally, a mean and covariance is needed. The TLE does not provide any information on the covariance of the object, so research has been done to try to find this. USSTRATCOM asserts that the position covariance is "about a kilometer or so at epoch and it quickly degrades" (pg. 30), though generally different orbital regimes have different errors associated with it [1, 7]. Dong and Chang-yin (2010) studied the orbital uncertainties produced with the SGP4 model and found that orbit uncertainties change differently with different orbital regimes. They found that the errors between SGP4

propagation and higher-order propagations approached 40 kilometers for objects in the GEO regime after 15 days of propagation [21]. Other work tried to compare the initial covariance of these TLE objects. Flohrer et al. (2009) compared and combined the along-track and out-of-plane position errors of GEO regime objects. They found that when comparing these errors from optical observations to TLE objects, the position error could be as large as 70 kilometers [22]. Frueh and Schildknecht (2012) compared the accuracy of TLE objects with optical measurements in the GEO and HEO regimes. Their results showed that objects in the GEO regime compared to their TLEs had position errors of 25km in the along-track direction, and 10km of error in the cross-track direction on average [64]. The undetermined covariance in TLEs is considered and modeled in this work and will be presented. Little and Frueh (2019) used an initial covariance along with the TLE as the mean state of the object to model the RSOs. Their covariance information will be used in this work [6].

2.3 Observer Position and Coordinate Systems

When an observation of a RSO is made, such as the ones necessary to create the TLE catalog, the observer position needs to be known, as well as where the RSO is in the observer's working frame. It is assumed that the coordinate systems used in this work are orthogonal, but the frames in this work can either be right-handed (such as the ECI frame) or left-handed (such as the Local Meridian Local Horizon) sets [9,13].

2.3.1 Observer Position and Rotational Effects on the Working Frame

If the Earth was a true sphere, then the observer position would be simple: take the latitude and longitude of the observer location and multiply the constant Earth radius. But as shown before, the Earth is not a perfect sphere but is an oblate spheroid [1, 9]. The oblate spheroid changes the radius of the Earth at different locations on the Earth based on latitude, which can be shown below:

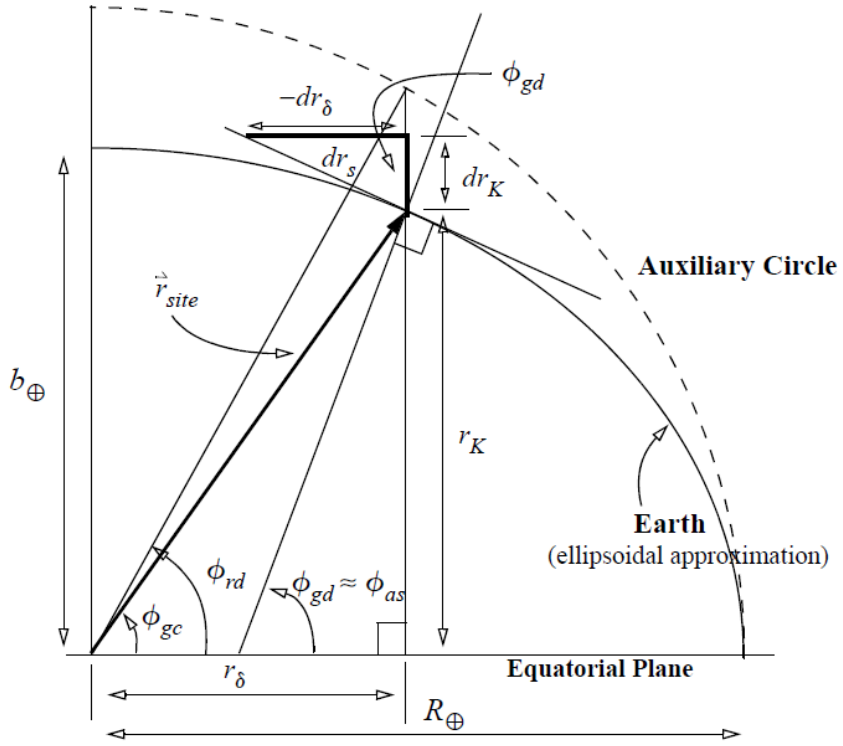


Figure 2.3. Visual on how the oblateness of the Earth changes the latitude, and the corresponding location of the observer. Accounting for this is important in achieving accurate measurements of RSOs (from Ref. [1], Figure 3-3 page 136).

With the concept in mind that Fig. 2.3 shows, a correction into the true latitude and radius of the observer position needs to be made. An approximation for the latitude and radius for the observer location can be made [1]:

$$\phi_{gc} = \phi_{rd} - 0.1924^\circ \sin(2\phi_{rd}) \tag{2.13}$$

$$R \approx 6378.14km - 21.38km \sin(\phi_{gc})^2 \tag{2.14}$$

Eqn. (2.13) and (2.14) are a first order approximation to correct for the observer's location difference due to Earth's oblatness using the uncorrected latitude (ϕ_{rd} from Fig. 2.3). With the new correct latitude and radius (ϕ_{gc} and R respectively), the observer position can be defined in vector form, shown below:

$$\overline{R}_{ECEF} = \begin{bmatrix} R\cos\lambda\cos\phi_{gc} \\ R\sin\lambda\cos\phi_{gc} \\ R\sin\phi_{gc} \end{bmatrix} \tag{2.15}$$

The ECEF frame is an earth fixed frame, which is useful for both keeping a constant location of the observer, and for calculating non-spherical harmonic effects on the satellite, as described in Eqn, (2.10) [1, 12]. For all the Earth fixed frames, a fixed direction needs to be defined for them to be considered "inertial." This direction is usually set to the vernal equinox. However, because the Earth is experiencing gravitational torques and other effects that change its orientation over time, the ECEF and other frames need to be regularly updated [1, 13]. There are three main perturbing effects on the Earth's orientation: Precession, Nutation, and Polar Motion, as shown in the figure below:

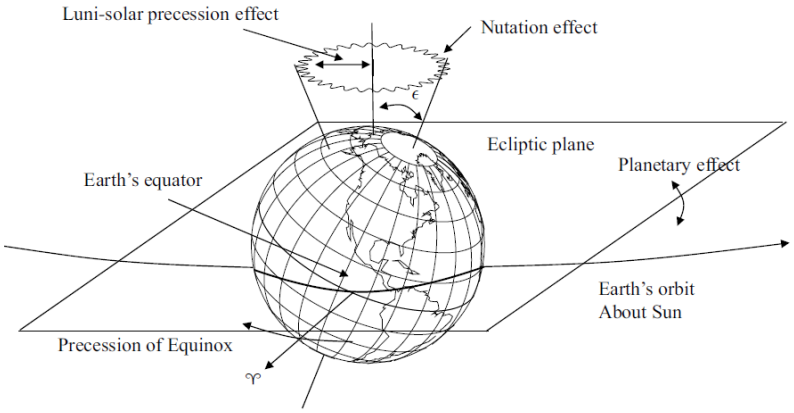


Figure 2.4. Because the Motion of Earth on its axes is not constant, the Precession and Nutation, along with the polar motion effect, need to be taken into account when formulating an accurate reference frame (from Ref. [1], Figure 3-25 page 206).

Precession, like nutation, is due to the torque from other celestial bodies, while polar motion is the motion of Earth's rotation axis with respect to Earth's crust [1].

Precession is the secular motion of the Earth's celestial north pole around the celestial north pole of the ecliptic. To calculate the effect of precession, the torque

on Earth needs to be calculated. This is done by treating the Earth as a rotationally symmetric body, then calculating the torque due to the sun and the moon on the axis of rotation. If the sun and moon were in the same plane as Earth's equator, then there would be no precession effects. But because the sun and the moon are inclined with respect to Earth's equator, the precession torques are presented, and calculated below [1, 9]:

$$\overline{D} = \frac{3}{2}(I - I_{eq}) \sin\epsilon \cos\epsilon (n_{sun}^2 + n_{moon}^2) e_x \tag{2.16}$$

The lunisolar torque ($\overline{D}$) on the Earth can be expressed in terms of the moments of inertia for the axially symmetric, oblate spheroid approximation of Earth (I, and I_{eq}), the obliquity of the ecliptic (ϵ), and the mean motion of the sun and moon (n_{sun} and n_{moon}).

The precession effects on the working frame can be expressed in the form of a 3-2-3 Euler angle rotation, or ζ, θ, and z shown below:

$$\zeta = 2306.2181''T + 0.30188''T^2 + 0.017998''T^3 \tag{2.17}$$

$$\theta = 2004.3109''T - 0.42665''T^2 - 0.041833''T^3 \tag{2.18}$$

$$z = \zeta + 0.79280''T^2 + 0.000205''T^3 \tag{2.19}$$

$$T = \frac{JD - 2451545}{36525} \tag{2.20}$$

Where JD is the current Julian date, and T is the epoch in Julian centuries. This precession defines the change of the coordinate frames at Mean Equinox of Date (MOD) for the epoch T, from the MOD at J2000 (J2000 is the frame calculation on January 1, 2000) [1]. From here, the classical Euler angle 3-2-3 rotation can be applied:

$$P = R_3(\zeta) R_2(-\theta) R_3(z) \tag{2.21}$$

The Nutation effects are periodic around Earth's rotational axis. This can be seen in Fig. 2.4 as the small indents creating the gear-like structure for the motion of Earth's rotation axis. Nutation is also due to the torque from the sun and moon, so Eqn. (2.16) is applicable and calculating the nutation changes [1]. Nutation is also can be expressed in the form of a 1-3-1 Euler angle rotation, or $-\epsilon - \Delta\epsilon$, $\Delta\Psi$, and ϵ shown below:

$$N = R_1(-\epsilon - \Delta\epsilon)R_3(\Delta\Psi)R_1(\epsilon) \tag{2.22}$$

$\Delta\epsilon$ and $\Delta\Psi$ are generally calculated from values computed in tables [9, 13, 14].

The last effect is the polar motion. Polar motion is calculated by treating Earth as an axially symmetric body without any torques applied. Polar motion cannot be determined by analytical formulas but is determined through observations. Because of this, the values x_p and y_p are from look-up tables and the effects of polar motion on the working frame are computed as follows [1, 12]:

$$\Pi = R_2(-x_p)R_1(-y_p) \tag{2.23}$$

2.3.2 Object Position Relative to the Observer

The Keplerian orbital elements found in the TLE (shown in section 2.2 of this paper) are in "inertial" centered frames, described in section 2.3.1. however, for ground-based sensors, it is often useful to find the position or other locating quantities of the object relative to the observer. The position vector of the object-of-interest is generally written either in the Earth-Centered Inertial (ECI) or the Earth Centered Earth Fixed (ECEF) frames. The ECEF frames are related to the observer's sidereal time, or Θ, as well as Precession, Nutation, and Polar motion effects described in section 2.3.1. These effects relate the reference frames as shown in the equation below [1]:

$$\overline{R}_{ECI} = \Pi(t)\Theta(t)N(t)P(t)\overline{R}_{ECEF} \tag{2.24}$$

This resulting R vector is relative to the center of the Earth. It is often more useful to calculate the distance from the observer to the RSO. The distance from the observer to the object-of-interest can be written as $\overline{\rho}$, which in vector form is a function of the distance (ρ), the topocentric right ascension (β), and the topocentric declination (δ).

$$\overline{\rho} = \overline{r} - \overline{R} = \begin{bmatrix} \rho \cos\beta \cos\delta \\ \rho \sin\beta \cos\delta \\ \rho \sin\delta \end{bmatrix} \tag{2.25}$$

Where $\overline{r}$ is the position vector of the object, and $\overline{R}$ is the position vector of the observer. It is important to note that the topocentric frame relative to the observer is no longer "inertial" and rotates with the observer [9]. The unit vector distance from the observer to the object-of-interest can be calculated as follows:

$$\overline{u} = \frac{\overline{\rho}}{\rho} = \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} \tag{2.26}$$

From Eqn (2.26), several useful frames, and their corresponding quantities for the location of the object-of-interest, can be calculated. Of special interest is the Local Meridian Equatorial (LME) reference frame. This is a left-handed reference frame, where the reference plane is parallel to the equator but is located at the observer location. This is also a rotating frame, which rotates along with the observer. Little and Frueh (2019) found that using this frame as the working frame for making observations in the sensor tasking problem decreased computational times and had a minimal error from transforming the states and covariances of the RSOs from ECI [1, 6, 9]. The objects position vector can be described by two angles and $\overline{\rho}$; the two angles are described below:

$$\tau = \tan^{-1}\left(\frac{u_x \sin\theta - u_y \cos\theta}{u_x \cos\theta + u_y \sin\theta}\right) \quad (2.27)$$

$$\delta = \sin^{-1}(u_z) \quad (2.28)$$

Where τ and δ are the hour angle and declination of the object for the specific sensor respectively, u_x, u_y, u_z are the x, y, and z range components from the sensor's location to the object, and θ is the sidereal time of the sensor [9].

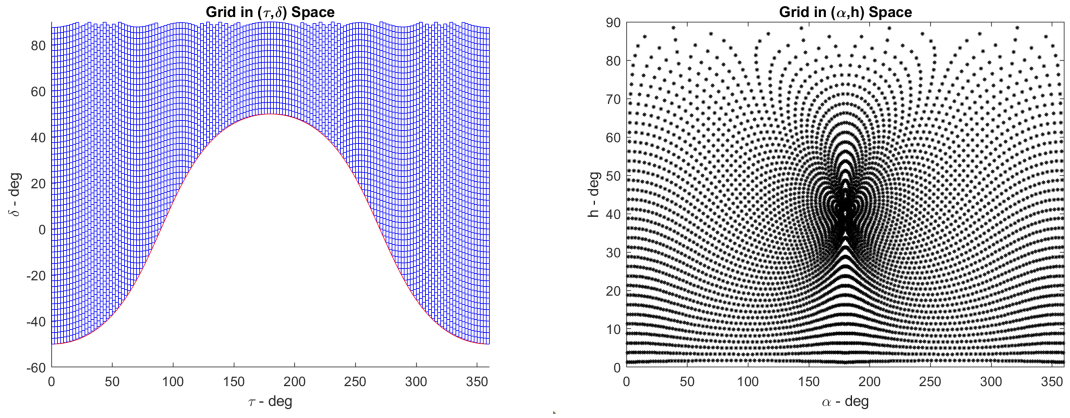
When the sensor makes an observation, it is made with respect to the Local Meridian, Local Horizon (LMLH) frame. This frame shifts the reference plane to be along the observer's south direction, which is the local horizon plane. The LMLH plane is a left-handed system, which is beneficial because the stars appear to move positively throughout the observation period, wherein a right-handed system would appear to move negatively [1]. The τ and δ frame transforms from the LMLH observation frame in azimuth and elevation (α and h, respectively) as the following:

$$\cos(h)\cos(\alpha) = \sin(\phi_{gc})\cos(\delta)\cos(\tau) - \cos(\phi_{gc})\sin(\delta) \quad (2.29)$$

$$\cos(h)\sin(\alpha) = \cos(\delta)\sin(\tau) \quad (2.30)$$

$$\sin(h) = \sin(\phi_{gc})\sin(\delta) + \cos(\phi_{gc})\cos(\delta)\cos(\tau) \quad (2.31)$$

Using Eqns. (2.29) – (2.31), the sensor tasking results can be transformed from the working frame (LME) to the observation frame (LMLH). An example of this transformation is shown in the figure below:



(a) τ, δ working frame

(b) α, h working frame

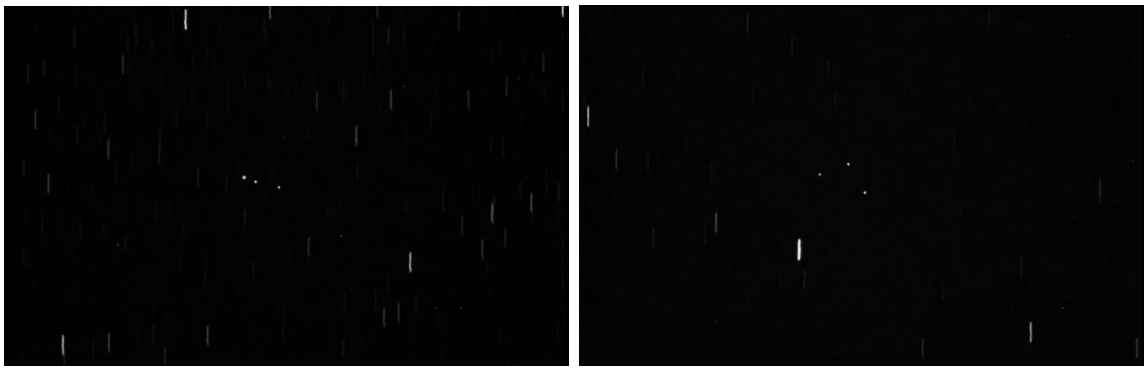
Figure 2.5. Using Eqn. (2.29) – (2.31), the working frame of the sensor can be transformed from τ, δ frame to α, h frame, and vice-versa.

Fig. 2.5a is an example of the field of regard (FOR) of a sensor in the working LME frame, and then is transformed into the observation frame (LMLH) in Fig. 2.5b. This is useful in visualization, as the LMLH frame represents the night sky for the observer [6].

2.4 Optical Sensors

The way TLEs are made are generally from ground-based sensors. These sensors are different and have different functions for them. such examples are optical sensors (usually used for GEO regime objects), radar sensors (usually for LEO objects), etc. For this work, optical sensors are used, and the working coordinate frame used is derived above [1, 8].

When an optical sensor takes an image, there are two distinct objects in the image: points and streaks [1, 8]. This is shown in Fig. 2.6:



(a) Image of three GEO satellites at the beginning of the night

(b) Image of three GEO satellites in the middle of the night

Figure 2.6. Example images created by a Charged-Coupled Device (CCD). These specific images were created by the Purdue Optical Ground Station (POGS) tracking the GEO objects (dots) with stars (streaks) in the background

In Fig. 2.6, the dots are shown to be GEO objects, and the streaks are stars (all pointing in the same direction. Streaks pointing in other directions are other objects). From these images, the mean and covariance states of these objects can be determined [1].

The optical sensors modeled in this work are assumed to be Charged Coupled Devices (CCD). The CCD collects electrons from the objects, stars, background noise, etc. from the image and measures its strength [8, 25]. To formulate this, the object's irradiance and signal function going into the optics of the sensor must be calculated:

$$I_{obj}(\bar{\lambda}) = I_{0,Sun}\frac{A}{\rho^2}(\frac{2C_d(\bar{\lambda})}{3\pi}(\sin\alpha + (\pi - \alpha)\cos\alpha)) \tag{2.32}$$

Here the irradiance $I_{obj}(\bar{\lambda})$ for spherical objects (also referred to as object brightness in this paper) is in units of watts per meter squared. $I_{0,Sun}$ is the irradiance of the sun, and A is the area of the object. This work is assuming all objects are spheres, so A remains constant. The true size and shape of the RSOs vary significantly. Operational satellites usually are attitude stabilized, thus the assumption of a sphere (constant area A) can be justified for the purpose of radiation effects.

Debris objects usually do have significant attitude motion, allowing for an averaged shape-attitude profile. Albeit usually the attitude motion is not uniform, but as a first-order approximation the canon ball model can be justified [16]. Setting all objects to a constant spherical shape allows for easy computation of the irradiance of the objects, and general assumptions about the probability of detection can be made [6]. $C_d(\bar{\lambda})$ is the diffuse reflection parameter at an average wavelength (assumed to be 600 nanometers in this work), and α is the Sun-Object-Observer phase angle [8].

$$S_{sig,obj} \approx (D-d)\frac{\bar{\lambda}}{hc} exp(-\tau(\lambda)R(\zeta))I_{obj}(\bar{\lambda})L \tag{2.33}$$

This work assumes that the sensors are equipped with a charge-couple device (CCD) to capture the images. $S_{sig,obj}$ is the signal function of the irradiance passing through the optics. D is the primary radius of the sensor's optics, and d is the secondary radius. λ is the average wavelength of light reflected off and h and c are Boltzman's constant and the speed of light constant respectively (for the remaining of the paper, h will refer to the elevation in the sensor's azimuth, elevation frame). $\tau(\lambda)$ is the atmospheric extension coefficient, $I_{obj}(\bar{\lambda})$ is the irradiance described in Eq. (2.33) and L is the loss function for the specific sensor. $R(\zeta)$ is an atmospheric model used based on the zenith angle of the object. It is expected that as elevation increases, the probability of detection of an object increases on average as the light from the object has less attenuating air-mass to pass through [8, 25]. The atmospheric effects are taken into account by $R(\zeta)$. This work is using a simple atmospheric model, known as the Van Rhijn factor, to improve computational time, shown below [8]:

$$R(\zeta) = \frac{1}{\cos\zeta} \tag{2.34}$$

Where ζ is the zenith angle from the sensor to the RSO, or $\zeta = \frac{\pi}{2} - h$. As elevation increases, $R(\zeta)$ decreases, and therefore the P_d of that object decreases.

Once the signal function of the RSOs is calculated, the expected value of this can be approximated as follows [8]:

$$E(S_{obj}) \approx E(S_{sig,obj})Q(\bar{\lambda})\Delta t \tag{2.35}$$

$Q(\bar{\lambda})$ is the loss function for the sensor, and Δt is the amount of time the sensor is collecting photons. For an object to be observed, the signal strength must be above the noise level at the detector in the sensor. This does not guarantee detection though. Detection is guaranteed if the object's Signal-to-Noise ratio (SNR) is sufficiently high [25]. There are several different sources for noise, some examples include background stars, errors in the readout process, sensor gains, etc. Calculation of the SNR value can account for these noise sources and others in the following Merline equation modified for CCDs:

$$SNR_{Merline} = \frac{\sum_i^{n_{pix}} \lambda_{obj,i}}{\sqrt{\sum_i^{n_{pix}} \lambda_{obj,i} + n_{pix}(1 + \frac{1}{n_b})(\lambda_{S,i} + \lambda_{D,i} + N_{R,i}^2 + \frac{g^2}{24})}} \tag{2.36}$$

The modified Merline equation takes into account the number of electrons per signal ($\lambda_{obj,i}$), sky background ($\lambda_{S,i}$), dark noise ($\lambda_{D,i}$), readout noise ($N_{R,i}^2$), and gain value $\left(\frac{g^2}{24}\right)$ for the CCD. n_{pix} is the number of pixels the signal is spread over, and n_b is the total number of background pixels in the image [8].

If the image strength is not significantly greater than the noise, then the object's state cannot be determined accurately [8, 25]. To try to model the probability that the object will be detected in simulation, the probability of detection (P_d) was formulated by Sanson and Frueh (2018). The following equation is the equation for the probability of detection:

$$\begin{aligned} P_d =& 1 - \frac{1}{2} \sum_{n=-\infty}^{n=\infty} \frac{\Gamma(n - \frac{g}{2}, \lambda_{obj,i} + \lambda_{S,i} + \lambda_{D,i})}{n!} \\ & \cdot \left(\left(\frac{n + 1 - t - \mu_{B,i}}{\sqrt{2g(\sigma_{B,i}^2 + \sigma_{R,i}^2)}} \right) - \left(\frac{n - t - \mu_{B,i}}{\sqrt{2g(\sigma_{B,i}^2 + \sigma_{R,i}^2)}} \right) \right) \end{aligned} \tag{2.37}$$

Here t is the user-defined threshold SNR value set for each specific sensor based on its characteristics, $\mu_{B,i}$ and $\sigma^2_{B,i}$ are the mean and variance of the number of background pixels, $\sigma^2_{R,i}$ is the variance of the readout noise, and is the error function. In this work, t is defined as three standard deviations of the Merline noise or three times the denominator of Eq. (2.36) [6, 25]. P_d has a range from zero to one, with an object having a $P_d = 0$ meaning there is a 0% probability of detecting that object, and a $P_d = 1$ meaning there is a 100% probability of detecting that object. The P_d is calculated for each object at each time. Then, a random number is chosen from a uniform probability distribution $C = U[0, 1]$. If $P_d \geq C$, then the object is considered detected at that time, otherwise it is not detected. This equation will be analyzed more and how it affects the single and multi-sensor tasking problem in this work.

3. SENSOR TASKING FORMULATION

This chapter focuses on the formulation of the sensor tasking optimizer used in this paper. The optimizing equation is examined for each of its parts, and how they affect the performance of the single sensor scenario. Several optimizing strategies are presented.

3.1 Orbit Propagation

To understand the position and velocity of resident space objects, sensors must track and maintain a catalog of past states to predict where the next observation should be. This is usually done by some sort of filter that the sensor uses to achieve the states of the objects based on the optical measurements. There is an assumed initial mean and covariance for this work, so no initial orbit determination, nor update process to achieve an initial covariance needed to be done. To begin with, a dynamic model and measurement model need to be defined. The following is used for this work [11]:

$$\dot{x}(t) = f(x(t)) + M(t)w(t) \tag{3.1}$$

$$z_k = h(x_k) + L_k v_k \tag{3.2}$$

Where $\dot{x}(t)$ is the velocity and acceleration state vector of the object and $f(x(t))$ is the dynamical model for the object's motion (in this work, just the two-body dynamics is assumed). $w(t)$ is the process noise in the system, and $M(t)$ is the mapping matrix from the noise to the dynamics. Eqn. (3.2) is the measurement model, where z_k is the actual measurements the sensor takes (τ and δ in this work), $h(x_k)$ is the non-linear mapping of the state x to the measurement space, v_k is the measurement noise,

and L_k is the mapping from the measurement noise to the measurement space [11]. The two-body orbit dynamics is done with an Extended Kalman Filter (EKF) in this work.

Due to noise in both the dynamics, such as an imperfect dynamical model and in the measurements, such as sensor pointing errors, the true state of the system is generally unknown. Instead, the objects mean position and velocity ($m(t)$), and its covariance ($P(t)$) is calculated as follows [1, 11, 50]:

$$m(t) = E\{x(t)\} \tag{3.3}$$

$$P(t) = E\{(x(t) - m(t)) \cdot (x(t) - m(t))^T\} \tag{3.4}$$

Here $E\{...\}$ is the expected value of the variables inside the braces. The initial states $m(0)$ and $P(0)$ is required for the Kalman Filter. The above equations provide the mean and covariance definitions for the EKF (first and second moments of the object's probability density function, or PDF). The first two moments allow the construction of a Gaussian PDF but higher moments are lost. Gaussianity is not preserved in the non-linear orbit dynamics. However, linearity can be assumed over short propagation times but is not a good approximation over longer propagation times [51]. The propagation of the mean and covariance is done as follows:

$$\dot{m}(t) = f(m(t)) \tag{3.5}$$

$$\dot{P}(t) = F(m(t))P(t) + P(t)F^T(m(t)) + M(t)Q_s(t)M^T(t) \tag{3.6}$$

Eqn. (3.5) is the rate of change of the mean ($\dot{m}(t)$), and Eqn. (3.6) is the rate of change of the covariance ($\dot{P}(t)$). $F(m(t))$ is the State Transition Matrix (STM) of $f(m(t))$, which in this work is the linearized dynamics of the two-body problem. Q_s is the process noise covariance and is derived from $w(t)$ in Eqn. (3.1). Q_s increases the covariance to absorb dynamical uncertainties and can be changed by the user of the EKF to keep the uncertainty of the state large enough to contain the state of the

object within a reasonable probability. In this paper, no process noise is used. In the simulations, the true dynamics and EFK dynamics are identical.

To use the EKF properly, measurements need to be recorded and inputted in. The EKF takes the previous measurements, mean, and covariance, and uses those to calculate an expected measurement, an updated mean, and an updated covariance. If there are too few measurements, the covariance will become too large in the along-track direction of the orbit, and the object will be considered "lost." With each new measurement, the EKF uses the following equations to update the mean and covariance [11]:

$$\hat{z}_k = h(m_k^-) \tag{3.7}$$

$$W_k = H(m_k^-)P_k^- H^T(m_k^-) + L_k R_k L_k^T \tag{3.8}$$

$$C_k = P_k^- H^T(m_k^-) \tag{3.9}$$

$$K_k = C_k W_k^{-1} \tag{3.10}$$

$$m_k^+ = m_k^- + K_k(z_k - \hat{z}_k) \tag{3.11}$$

$$P_k^+ = P_k^- - C_k K_k^T - K_k C_k^T + K_k W_k K_k^T \tag{3.12}$$

In Eqns. (3.7) – (3.12), all variables with a "$-$" in the superscript are values before the Kalman filter update, while all variables with a "$+$" are values after the Kalman update. $\hat{z}_k$ is the expected measurement based on the mean state, R_k is the measurement noise covariance, and $H(m_k^-)$ is the measurement Jacobian at the mean before the Kalman update. W_k is referred to as the Innovations Covariance (which is useful in understanding the performance of the filter in the real-world), C_k is the cross-covariance, and K_k is the Kalman gain. These values are used to calculate the updated mean and covariance, shown in Eqn. (3.11) and Eqn. (3.12) [11]. With the new mean and covariance, once a new measurement is done, the process repeats itself, starting back again with Eqns. (3.5) and (3.6).

As stated before, the EKF needs an initial mean and covariance to begin. For this work, the following initial conditions are assumed for all GEO objects [6, 22, 64]:

$$m_0 = \overline{r}_{TLE} \tag{3.13}$$

$$P_0 = \begin{bmatrix} 50^2 km^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 50^2 km^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 50^2 km^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1^2 \frac{m^2}{s^2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1^2 \frac{m^2}{s^2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1^2 \frac{m^2}{s^2} \end{bmatrix} \tag{3.14}$$

$\overline{r}_{TLE}$ is the position state derived from the keplerian orbital elements from the TLE at the starting epoch. Because the TLEs do not have any information on the covariance of each object, a constant covariance is assumed, shown in Eqn. (3.14) [6, 22]. Frueh et al. (2012) showed that for GEO objects, the size of the covariance from TLEs tends to range from 30km-50km, so the initial covariance assumed in Eq. (3.14) is a valid assumption [64].

3.2 Sensor Tasking formulation

In this work the sensor tasking formulation developed by Frueh et al. (2018) is used [24]. The equation derived in their work is very comprehensive, containing optimization factors for most sensor tasking cases. In this work, several simplifying assumptions are made about the sensor tasking scenario to reduce complexity. The Equation derived by Frueh et al. (2018) does not assume these equations to remain applicable to most all sensor tasking scenarios.

The first assumption specific to this paper is that every sensor available is used for their entire time, i.e. the sensors are not shared by multiple users. Secondly, it is assumed that the observation takes place in the middle of the total number of

images taken by the sensor, regardless of how many images are taken. Finally, it is assumed that the sensor tasking scenario consists only of tracking, and the sensors are not surveying [6]. With these assumptions specific to this research, the following simplified equation is derived:

$$\max A = \sum_{g=1}^{l} \sum_{f=1}^{m_g} \left(\sum_{i=1}^{n} \mu(\tilde{X}_i) \cdot \ P_d(h_{f,g}, \tilde{X}_i) \cdot \ d(h_{f,g}, \tilde{X}_i, P_i) \right) \tag{3.15}$$

A is the total number of RSOs to be observed. l is the total number of sensors observing in the time-frame, and m_g is the number of observations (or viewing directions) taken by each sensor g. n is the total number of RSOs to be observed in the time-frame. $\mu(\tilde{X}_i)$ is an object i specific value. This value can either be increased depending on facts like observability of object i [29], if object i has been observed or not (set as a 1 and 0 respectively), etc. In this paper, all objects are initially set with $\mu(\tilde{X}_i) = 1$ and then changed to 0 once the object has been observed. $P_d(h_{f,g}, \tilde{X}_i)$ is the probability of detection of object i. More detail on $P_d(h_{f,g}, \tilde{X}_i)$ is shown in Eqn. (2.37). $d(h_{f,g}, \tilde{X}_i, P_i)$ is the probability that object i falls within the FOV grid choice centered at $h_{f,g}$. For this paper, scenarios including and ignoring covariance are considered [6, 24].

Eqn. (3.15) can be broken up to show how each variable affects the sensor tasking problem. Fig. 3.1 shows the simplest case.

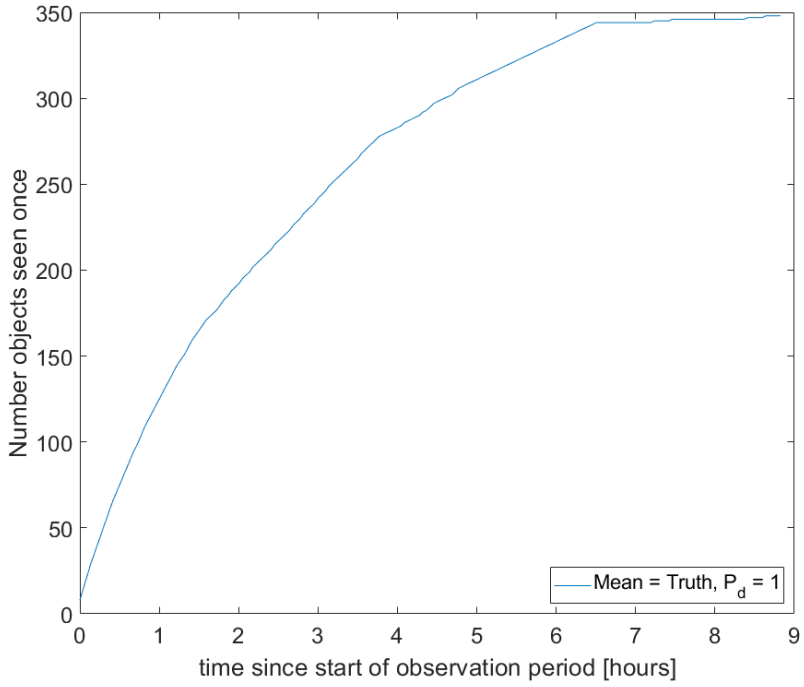


Figure 3.1. The Performance of a single sensor using Eqn. (3.15). These results assume that the probability of detection for each object ($P_d(h_{f,g}, \tilde{X}_i)$) is 1 at all times, and that the mean state of the object is the true state.

This case assumes only one sensor is making observations and is using a simple greedy algorithm to solve for the optimization (more information in section 3.3.1) [6]. The $P_d(h_{f,g}, \tilde{X}_i)$ for each object at each time is assumed to be one, and $d(h_{f,g}, \tilde{X}_i, P_i)$ for each satellite at each time is ignored (i.e., the mean state of the object is the true state). This simplifies Eqn. (3.15) to the following:

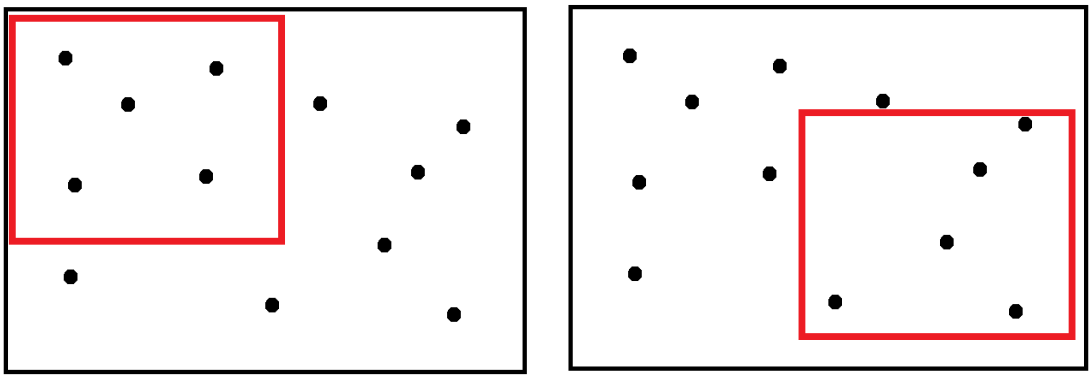
$$\max A = \sum_{f=1}^{m_g} \left(\sum_{i=1}^{n} \mu(\tilde{X}_i) \cdot \ P_d(h_{f,g}, \tilde{X}_i) \right) \tag{3.16}$$

This is an ideal case for the single sensor tasking problem, where at all times, for all objects, the probability of detection is one. The overall performance of a single sensor generally cannot be greater than what is shown in Fig. 3.1, as a real sensor using Eqn. (3.15) must consider the probability of detection and the covariance of an object, reducing the performance of the sensor. The following sections will build

from Eqn. (3.16), including more effects on the object and sensor tasking formulation, until Eqn. (3.15) is reached for the single sensor case.

3.2.1 Optimization of the Field of Regard

Before the sensor tasking optimization can occur, the Field of Regard (FOR) of the sensor must be analyzed. In fact, the FOR is discretized to apply an optimizer based on the sensor's field of view (FOV), which Eqn. (3.15) is.



(a) Choosing five objects on the left side of local sky

(b) Choosing five objects on the right side of local sky

Figure 3.2. If there is no discretization of the Field of Regard (FOR), then there are an infinite number of viewing directions possible. An example of this is shown, where multiple viewing directions can see five objects. The algorithm cannot determine the optimal solution without some discretization in place.

As shown in Fig. 3.2, if there is no discretization of the FOR, then there are infinite possibilities for viewing directions. Because of this, there can be multiple FOV choices that yield the "maximum" number of objects observed at this time-step. The optimizer cannot choose a specific max in this case, and therefore breaks the algorithm.

To avoid this issue, the FOR in the working frame is discretized based on the sensor's FOV. The working frame chosen is the δ, τ frame. Little and Frueh (2019)

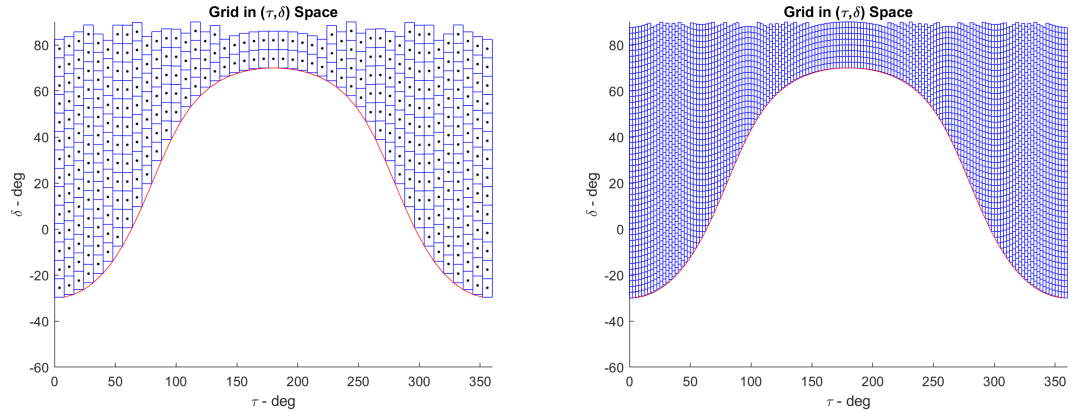
showed that this working frame both had the minimum errors when transforming the covariance, as well as had reduced computational time as compared to other working frames [6, 26, 52]. To discretize this working frame, first, the δ value in the working framework that is associated with the minimum elevation (h_{min}) needs to be found. The minimum elevation can either be the local horizon or some elevation restrictions around the sensor (trees, buildings, etc.). Eqns. (2.29) – (2.31) are rewritten to find this, shown below:

$$\delta_{min} = \sin^{-1}\big(\sin\phi_{gc}\sin h_{min} - \cos\phi_{gc}\cos h_{min}\cos\alpha\big) \tag{3.17}$$

Once the δ_{min} is found at the specific τ, then the working frame is built up based on the sensor's FOV, checking to make sure that the elevation constraint is not violated at the specific δ. Once the minimum δ is calculated at the specific τ, the working frame is discretized vertically based on the FOV of the sensor. At each new FOV viewing direction, the following equation is used to make sure the elevation constraint is not violated:

$$h = \sin^{-1}\big(\sin\phi_{gc}\sin\delta_{min} + \cos\phi_{gc}\cos\delta_{min}\cos\tau\big) = h_{min} \tag{3.18}$$

Finally, the top of the FOV is checked to make sure that the viewing direction is not above 90° [6]. Fig. 3.3 shows the result of the working frame of an example sensor with different size FOVs discretized.



(a) FOR space with FOV = 8° (b) FOR space with FOV = 2.5°

Figure 3.3. Comparison of how different Field of Views (FOV) affect the grid discretization in the Field of Regard (FOR). Fig. 3.3a has a larger FOV, and therefore less viewing directions, while Fig. 3.3b as a smaller FOV, and therefore more viewing directions. The sensors are located at a latitude of 40°.

In Fig. 3.3a, the sensor has a FOV of 8°, while in Fig. 3.3b, the sensor has a FOV of 2.5°. Both of these scenarios have an elevation constraint of 20°, which is shown in red, and both sensors are located at a latitude of 40°. With a larger FOV, the sensor has a fewer amount of possible viewing directions, while with a smaller FOV, the sensor has a greater amount of possible viewing directions. The transformation from the working frame τ, δ to the α, h working frame presented in Fig. 2.5b shows a significant amount of viewing directions clustered together. This is due to the FOV discretization in the τ, δ frame. The computational time can be decreased by setting a maximum elevation on the sensor's FOR, though the computational time does not decrease significantly, as the computational time is more of a factor of the object propagation, discussed in chapter 6.

With the FOR now discretized, the RSOs can be placed in the sensor's working frame, and the optimizer can choose which viewing direction is optimal (different at each time-step based on what type of optimizer is used). Fig. 3.4 uses Eqns. (2.27) and (2.28) to transform the RSOs position state into the sensor's working frame.

These objects are taken from the USSTRATCOM TLE catalog for the day of March 19th, 2019. Objects considered are ranged from $39000km \leq a \leq 44000km$ in semi-major axis, and $e \leq .5$ in eccentricity. the sensor is located at a latitude of 40° and a longitude of 75°, with an altitude of 2.221 kilometers and a FOV of 2.5°.

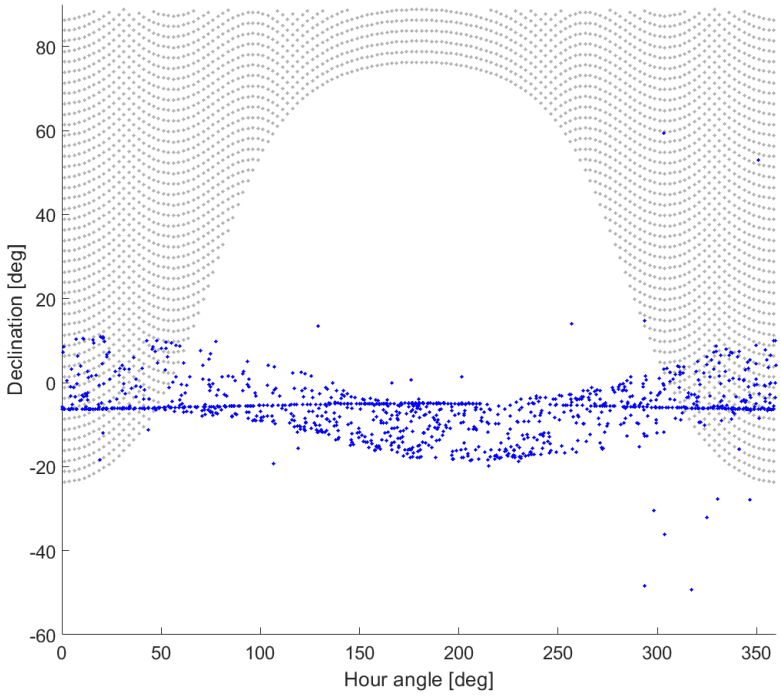


Figure 3.4. A visual representation of how the satellite position vector states transform into the sensor's working frame using Eqns. (2.27) and (2.28). The blue dots are the individual GEO satellites, while the light gray dots are the Sensor's possible viewing directions.

With the objects in the working frame, shown in Fig. 3.4, the sensor will choose the specific gray dot viewing direction that the optimizer determines is optimal. Depending on factors like the probability of detection, covariance, etc., the choice of viewing direction will change. Because the FOR is discretized though, the algorithm will determine only one viewing direction that is optimal, instead of multiple [6].

3.2.2 Probability of Detection in Sensor Tasking

Building from Eqn. (3.16), the probability of detection needs to be calculated at all times for all RSOs in the sensor tasking problem. $P_d(h_{f,g}, \tilde{X}_i)$, or also referred to as P_d in this paper, refers to the probability that an object will be detected by the sensor-based on the geometry of the sun-to-RSO-to-sensor, as well as noise properties (SNR) coming into the Charged Coupled Device assumed in this work. Details on the probability of detection in the sensor tasking problem are introduced in section 2.4, chapter 4, and Eqns. (2.32) – (2.37). This section's purpose is to introduce how P_d affects the sensor tasking equation presented by Frueh et al. (2018) and compare it with the case presented in Fig. 3.1.

When considering P_d in the single sensor tasking scenario, the sensor tasking scenario becomes more complex. It is expected, that as we move from the ideal case ($P_d = 1$) to a more complex, but closer to a real-world example, that the sensor tasking algorithm decreases in performance. This idea, and how it will affect the performance of the sensor tasking problem, is shown in Fig. 3.5.

This is still a simplified scenario of the overall sensor tasking problem. The mean TLE state of the object is still considered the truth. Fig. 3.1 can now be updated to include the results when calculating the P_d for each object, as well as assuming $P_d = 1$ for all objects at all times.

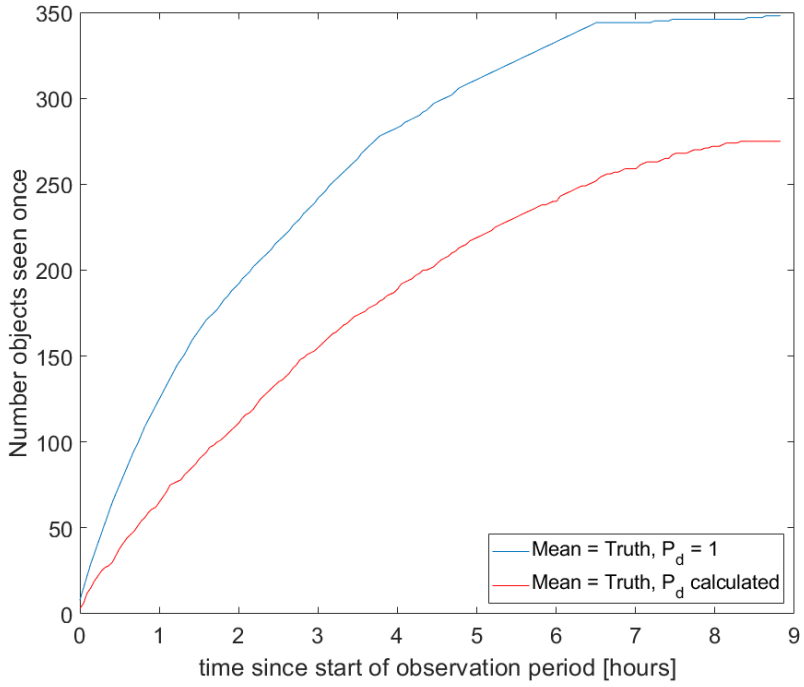


Figure 3.5. The Performance of a single sensor using Eqn. (3.15). These results now include the results if $P_d(h_{f,g}, \tilde{X}_i)$ is calculated for each object at each observation time. Both of these results still assume that the mean state of the object is the true state.

As expected, if the probability of detection is not considered to be one for all objects at all times (blue line in Fig. 3.5), then there is a decrease in performance, but the simulation of objects and making observations is more accurate to the real-world scenario. Though Fig. 3.5 does not include the covariance of each object, the results can be useful. Just using this simplified form can help isolate the effects of the probability of detection in the single and multi-sensor tasking scenario. These forms of the overall sensor tasking equation will be used in future sections of this paper.

3.2.3 Cumulative Distribution Function in Sensor Tasking

Finally, the last term needs to be considered in the sensor tasking algorithm, the cumulative distribution function (CDF) of each RSO, or the covariance of the RSOs in the working frame. Adding this term requires that the covariance of the object

needs to be transformed into the measurement space of the sensor. This is done using linearization, neglecting the apparent non-linearity of the true transformation through the measurement Jacobian H_k, described in section 3.1 [6, 11]. To do this, H_k needs to be written in terms of the measurement state, τ, and δ.

$$H_k = \begin{bmatrix} \frac{\partial \tau}{\partial x} & \frac{\partial \tau}{\partial y} & \frac{\partial \tau}{\partial z} & \frac{\partial \tau}{\partial \dot{x}} & \frac{\partial \tau}{\partial \dot{y}} & \frac{\partial \tau}{\partial \dot{z}} \\ \\ \frac{\partial \delta}{\partial x} & \frac{\partial \delta}{\partial y} & \frac{\partial \delta}{\partial z} & \frac{\partial \delta}{\partial \dot{x}} & \frac{\partial \delta}{\partial \dot{y}} & \frac{\partial \delta}{\partial \dot{z}} \end{bmatrix} \tag{3.19}$$

Once H_k is determined, then the covariance of each object can be transformed as follows [6, 26, 52]:

$$P_z = H_k P H_k^T \tag{3.20}$$

When the covariance P_z is found, the PDF of that object must be determined for all grid fields in the sensor's working frame. This allows the sensor to determine which grids have the highest probability of obtaining the true state of the object, shown below:

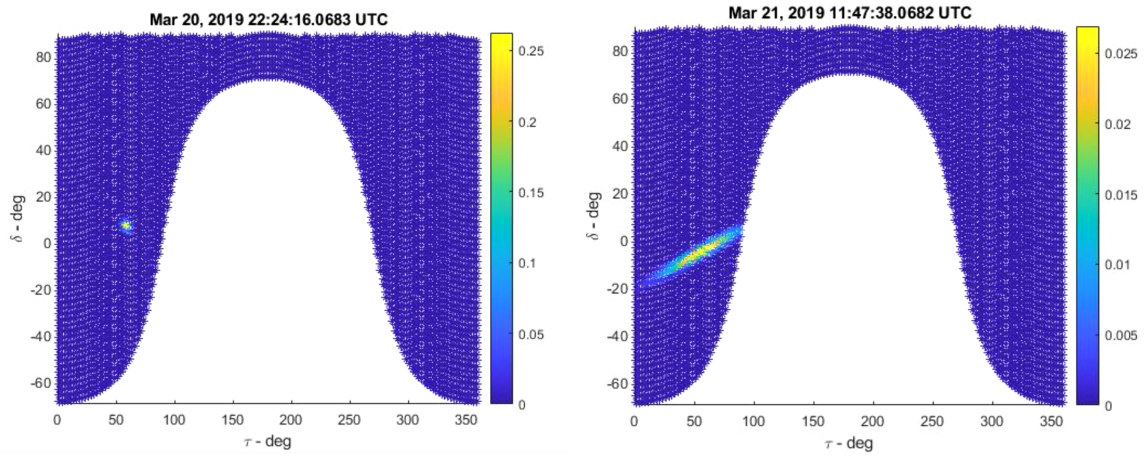
$$d(h_{f,g}, \tilde{Z}_i, P_{Z_i}) = \int_h \frac{1}{\sqrt{(2\pi)^2 \mid P_{Z_i} \mid}} \exp\left(-\frac{(Z_{i,h} - \tilde{Z}_i)^T P_{Z_i}^{-1} (Z_{i,h} - \tilde{Z}_i)}{2} \right) dh \tag{3.21}$$

Where $\tilde{Z}_i$ is the object's i mean state in the measurement space, P_{Z_i} is the transformed covariance of the object i using Eqn. (3.19) and (3.20), h is the grid field for which the integration is performed over, and $Z_{i,h}$ is the possible measurements of the object in the grid field [6]. Doing this for every object, and then summing up the CDF of each object will achieve the combined CDF of the grid in the working frame, as follows:

$$V(h_{f,g}) = \sum_{i=1}^{n} d(h_{f,g}, \tilde{Z}_i P_{Z_i}) \tag{3.22}$$

The summation term in Eqn. (3.22) represents each viewing direction's CDF for all objects at that specific time [6]. Due to the nature of the covariance in the orbit problem, the covariance grows and changes shape as the object moves around the

Earth [11]. Because of this growth, the CDF for each grid needs to be calculated at each time step. Specifically, an example of how the CDF for the working frame changes for a single object is shown. The sensor is located at a latitude of 20°, a longitude of -103°, and an altitude of 2.221 kilometers. There is no elevation constraint on the sensor, and the sensor's FOV = 2.5°. The object shown has the International Designator of 1997-029C from the TLE catalog taken on March 19^{th}, 2019.



(a) Single object's CDF at the start of the night

(b) Single object's CDF at the end of the night

Figure 3.6. Visual comparison of how an object's CDF evolves within the sensor's working frame. The color bar represents the CDF value for each grid (FOV = 2.5°). This shows an exaggerated initial of P_0 for visual purposes. The real P_0 used is much smaller and is given in Eqn. (3.14).

At the initial time, the covariance is P_0. In Fig. 3.6, this P_0 is equal to $\sigma_x^2 = \sigma_y^2 = \sigma_z^2 = 100^2 km^2/s^2$, and $\sigma_{Vx}^2 = \sigma_{Vy}^2 = \sigma_{Vz}^2 = 50 m^2/s^2$. This is a exaggerated P_0, to show the changing CDF, and the real P_0 is for the rest of this paper is shown in Eqn. (3.4) [22, 64]. Even though this is exaggerated, the effects of propagating the covariance and transforming it into the sensor's working frame can be shown. As the simulation progresses, each specific grid will decrease in CDF value as the covariance spreads out, though not as drastic as what is shown in Fig. 3.6.

With the CDF formulation presented in Eqns. (3.19) - (3.22), the sensor tasking algorithm can now consider in this effect for the single sensor. The single sensor results using Eqn. (3.16) and Eqn. (3.15) can be shown in fig. 3.7 to compare how each term affects the solution for this example case.

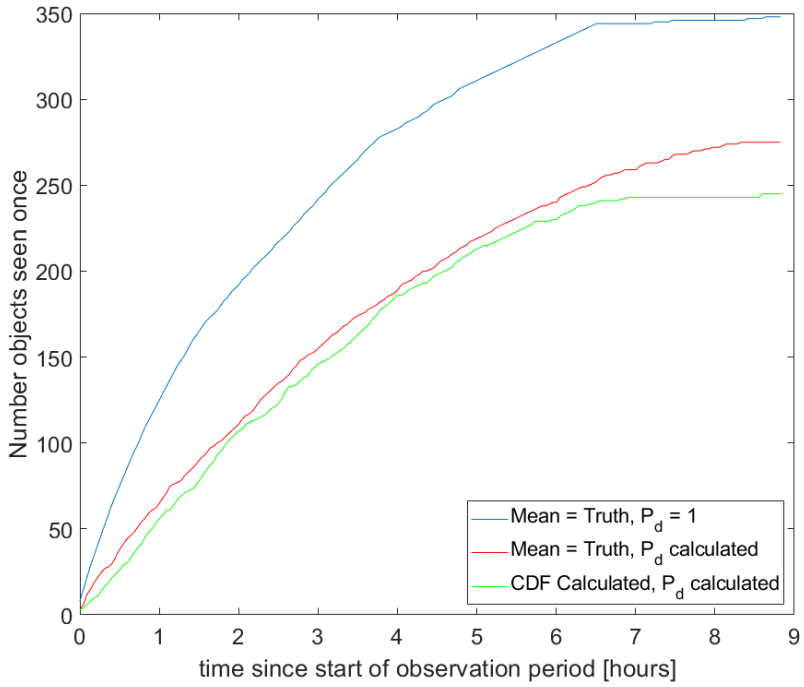


Figure 3.7. The performance of a single sensor using Eqn. (3.15). These results now include the results if $P_d(h_{f,g}, \tilde{X}_i)$ is calculated for each object at each observation time, as well as results for calculating $P_d(h_{f,g}, \tilde{X}_i)$ AND the CDF of each object $(d(h_{f,g}, \tilde{X}_i, P_i))$

Two main results can be determined from Fig. 3.7 when adding the CDF of all the objects. First, as expected, when more uncertainty is added into the system, the performance of the sensor decreases, but the model is closer to the real-world case. Secondly, as discussed before, initially the CDF model performs fairly close to the non-CDF model that includes P_d. But as the simulation continues and the covariance increases, the performance worsens. This is in agreement with Fig. 3.6. The case of no CDF (the mean state is the true state) and with P_d included, as well as various other initial covariances and P_d included is shown in Fig. 3.8, to emphasize this effect:

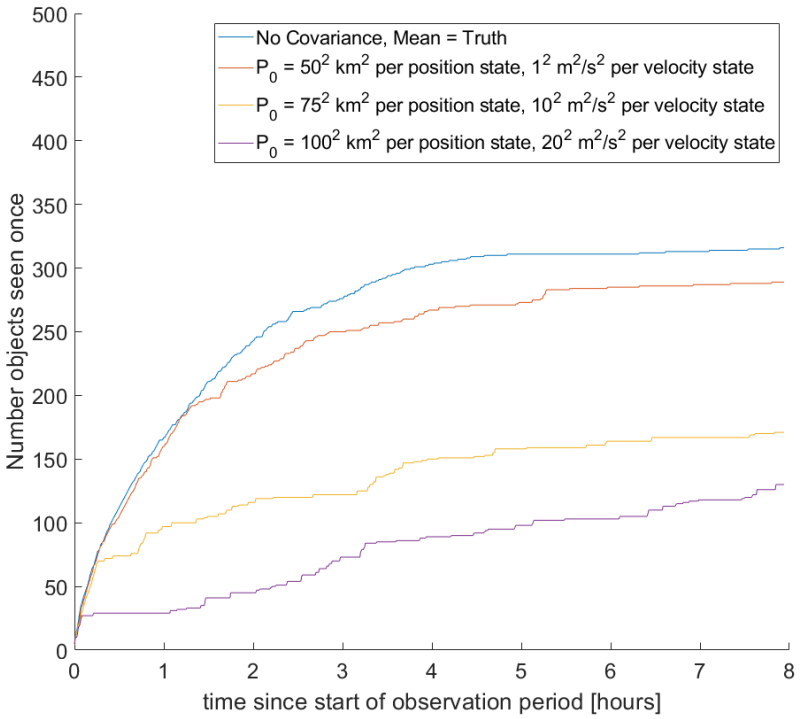


Figure 3.8. Comparison of how the different initial P_0 changes the overall Sensor Tasking solution for a single sensor system. The initial P_0 values are all along the diagonal, with zero values in the off-diagonal (like in Eqn. (3.14)). All the position states have the same P_{0i} value, and all the velocity states have the same P_{0i} value.

As expected, when the initial uncertainty is small (red line in Fig. 3.8), the performance is initially close to the no CDF case but diverges as the simulation continues. This is emphasized by the greater P_0 cases in Fig. 3.8.

It is also important to note that the CDF assumption assumes that the distribution is Gaussian at all times [11]. In the orbit problem, this is a good assumption initially, but as the propagation continues, this does not hold, and the distribution appears to look more like a banana shape [51]. For this work, the propagation time is small for the GEO objects, at most 30 hours, and the assumption that the uncertainty remains Gaussian is justified [6].

3.3 Sensor Tasking Optimizers

General optimizing problems can be classified into either Convex or Non-convex problems [55, 56]. Specifically for the orbit problem, Little and Frueh (2019) studied both convex and non-convex optimization strategies. They found that the orbit problem can be solved with both, though the problem itself is most likely non-convex [6]. In this section, four optimization strategies will be discussed: two convex, and two non-convex.

3.3.1 Greedy Optimizer

In the sensor tasking problem using Eqn. (3.15), the algorithm chooses the viewing direction that is maximal. This is shown below:

$$h^* = \text{argmax} \sum_{i=1}^{n} \mu(\tilde{X}_i) \cdot \ P_d(h_{f,g}, \tilde{X}_i) \cdot \ d(h_{f,g}, \tilde{Z}_i, P_{Z_i}) \tag{3.23}$$

Here, h^* is the chosen grid for the specific sensor at the specific observation time. The probability of detection of object i, $(P_d(h_{f,g}, \tilde{X}_i))$ and its specific CDF value in the grid viewing directions $(d(h_{f,g}, \tilde{Z}_i, P_{Z_i})$, affect the weight of each grid. Each object's weight in each grid is summed up, and the algorithm chooses the grid with the highest weight. Once the algorithm observes the specific object i, its $\mu(\tilde{X}_i)$ is set to zero. This informs the algorithm to not prioritize viewing this specific object and will affect future viewing directions [6].

The greedy algorithm is a very simple convex optimization strategy that only considers the local optimization, rather than the global optimization. If the problem the greedy algorithm is trying to optimize is convex, then the greedy algorithm will produce global optimal results [55–58]. The orbital sensor tasking problem is very likely non-convex, so in general, the greedy algorithm does not produce truly optimal results [6]. The greedy algorithm can be improved in the sensor tasking problem if

other factors are considered in the viewing direction choice, such as if the rise and set times of RSOs are considered rather than just positional states [24].

The benefit of using the greedy algorithm over other optimization strategies is its computational efficiency. Because the greedy algorithm chooses the local optimal solution and does not consider other time steps, the total simulation time is significantly reduced [6, 55]. This work uses the simple greedy algorithm for all simulations. Little and Frueh (2019) showed that though the greedy algorithm did perform slightly worse than non-convex optimizing strategies, the computational efficiency benefits outweighed the small increase in performance [6].

3.3.2 Other Optimizing Techniques

The other convex method that Little and Frueh (2019) studied is the Weapon Target Association (WTA) [6]. The basic form of WTA is to minimize the damage by the targets on the asset being defended [59]. WTA is classically set as a minimization problem, but in the sensor tasking scenario, it can be set as a maximization one, where the RSOs to be observed are the "targets" and the grid viewing directions available are the "weapons." Because WTA is a convex optimizer in the non-convex sensor tasking problem, other optimizers produce better results, though WTA is a fairly inexpensive optimizer in terms of computational cost [6].

For the non-convex optimizers, Little and Frueh (2019) analyzed Ant Colony Optimization, as well as Distributed Q-Learning. These optimizers are reinforced learning methods, and require some machine learning techniques to implement [6].

Ant Colony Optimization, or ACO, was first introduced by Dorigo et al. (1991) and has been used in several classical and complex optimization problems, such as the Travelling Salesman Problem [60]. The optimizer is built on the idea that, in the limit, the ants will find the shortest, or optimal, the path from their nest to the food source [61]. In the case of the orbit problem, the nest can be considered the sensors available, the path is the grid viewing directions, and the food source is the total

number of RSOs detected. Little and Frueh (2019) found that for the sensor tasking problem, ACO detected the most amount of unique RSOs, though the computational cost was extreme [6].

Distributed Q-Learning, or DQL, was developed by Mariano and Morales. Their method extends to the single-agent Q-Learning method [62]. Little and Frueh (2019) showed that DQL does improve the performance of the simple greedy algorithm (which is needed for DQL) in the sensor tasking problem, but like ACO, the computational cost is too expensive for practical use [6].

3.4 Immediate Feedback in Sensor Tasking

The simulations done in this paper assume that immediate feedback is available. This implies for a single sensor that once an object has been detected, that sensor knows immediately that the object has been observed, and can immediately update the μ_i for that object. In the multi-sensor case, this implies that all sensor's know which RSO each sensor has observed immediately [53]. Fig. 3.9 shows visually how immediate feedback plays a part.

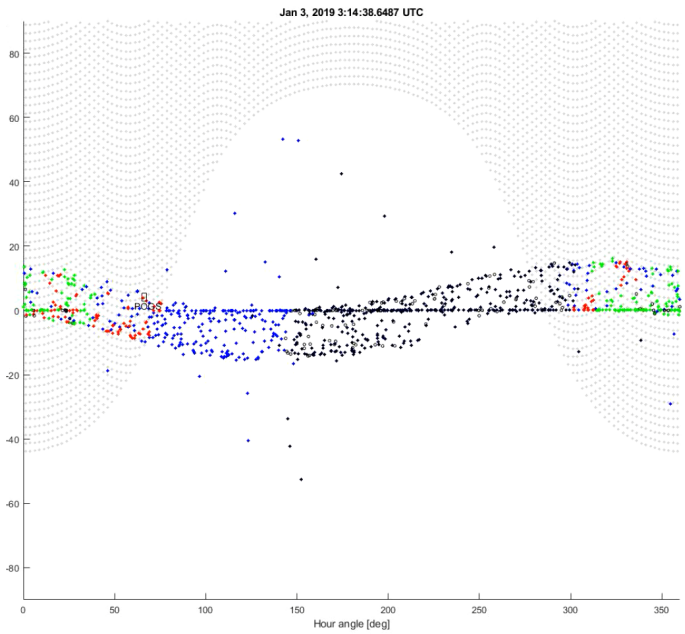


Figure 3.9. An example of how immediate feedback affects the multi-sensor tasking scenario. Sensor 'POGS' knows it has seen objects in red, as well as it knows the other sensors have seen objects in green and black. 'POGS' knows it needs to see blue objects immediately at each observation time-step.

Each of the three sensors in Fig. 3.9 is either currently making observations, or have already completed their observation period. Immediate feedback here means that each sensor already is aware of the μ_i value of the RSOs that they have observed, as well as the μ_i value that all other sensors have calculated, and vice versa [6,49,53,54].

Immediate feedback is an ideal case, as it is impossible to transmit data between sensors instantaneously. Immediate feedback also assumes that the orbits are processed and true detections are confirmed and matched to the objects (no mismatching of the objects). This process is never immediate in actual sensor tasking. If immediate feedback is not assumed, a Monte Carlo analysis is generally used to approximate the feedback. This produces accurate results but is extremely time consuming and computationally expensive. Little and Frueh (2019) developed the Predicted Measurement Probability method, or PMP, to try to account for scenarios without immediate feedback that is more computationally efficient. The PMP method seeks to estimate

the measurement of an object with PDF, like Monte Carlo. It then uses this estimate to provide feedback that predicts which RSOs were observed at the observation time. Little and Frueh showed that, when there is no immediate feedback, the PMP method can produce results fairly close to the Monte Carlo method (the theoretical maximum for this problem), with extremely low relative computational cost [6].

4. THE PROBABILITY OF DETECTION VS. ELEVATION CONSTRAINT

The optical sensors in this work are assumed to be made up of three primary parts: the mount, the optics, and the electronics to read the optical images. The mount describes how the sensor can move relative to its location, such as an Equatorial Mount geometry or an Alt-azimuth Geometry. The optics refer to how the sensor collects light from the object-of-interest. Some examples of optics generally used are refactors and wide-field telescopes [8]. As for the image processing, this work assumes that the optics of the ground-based sensors (GBS) read images using a Charge-Coupled Device (CCD). The CCD takes in electrons that reflect off the object-of-interest coming in through the optics and records the exposure to readout the object's measurement state. If there were no errors or background light sources, then a detector can read the object's state with high accuracy. In the real world system, however, this is not the case, and these effects need to be studied [8, 25, 31]. This chapter will present what attenuation effects are, and how they affect the images produced by the CCD. Then, the probability of detection for GEO objects will be analyzed in the sensor's FOR throughout an observation period with elevation constraints considered.

When the light from the object-of-interest passes through Earth's atmosphere, this leads to a number of offsets, such as refraction, scintillation, but also attenuation. The attenuation effects are best illustrated in Fig. 4.1:

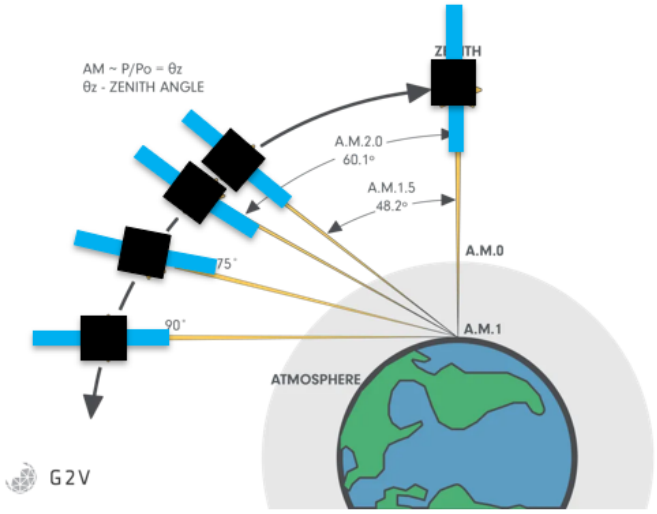
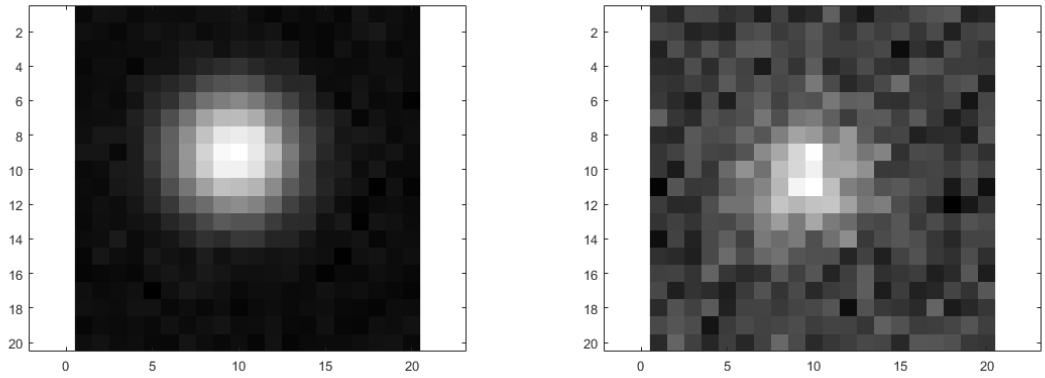


Figure 4.1. A visual representation of attenuation effects on ground-based sensors. If the Object-of-interest is at high elevation values, the light traveling from the object to the CCD has less atmosphere to travel through, vs. if the object is at low elevation values (Image is taken from G2V Optics [65]).

Depending on where the object-of-interest is relative to the GBS, the attenuation effects will change. If the object-of-interest is close to the local horizon of the sensor, then the light reflecting off the object has more of Earth's atmosphere to travel through. Traveling through more atmosphere leads to less light from the object-of-interest being collected, which decreases the SNR. Having a smaller SNR results in an image which is harder to determine the location, or centroid, of the object, leading to more uncertainty in the object's location [8,25]. If the object, however, is far from the local horizon, the amount of atmosphere the light travels through is much less, and therefore the results from the CCD have a higher accuracy [8,25,37]. Therefore, there is a clear relationship between the elevation of the object relative to the sensor, and the accuracy of the image taken. This will affect the sensor tasking problem, which has already been shown in section 3.2.2.

Other effects increase the noise in the CCD image. These effects include background light sources, readout and dark noise measurements, attenuation (as men-

tioned), etc. Depending on how strong or weak these effects are, they will change the quality of the CCD image. Two examples are shown in Fig. 4.2 emphasize the importance of taking these effects into account.



(a) Image of an object with SNR = 50 (b) Image of an object with SNR = 8

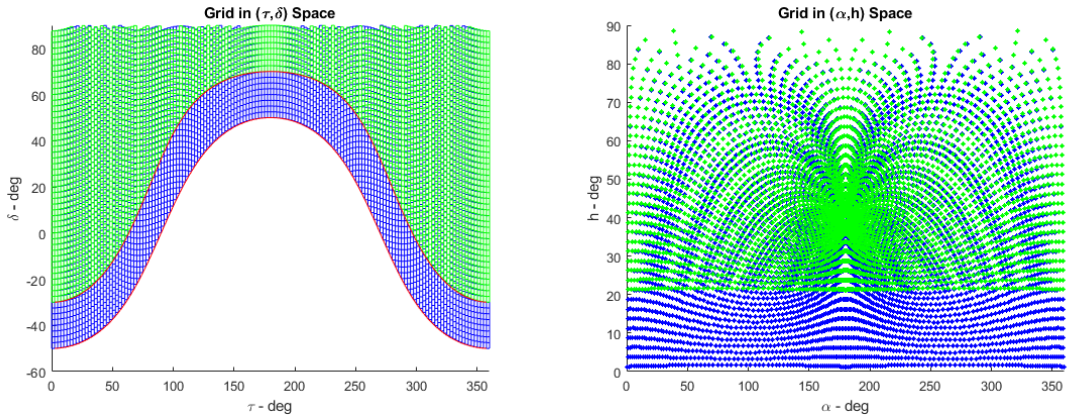
Figure 4.2. Comparison of different Signal-To-Noise ratios on the resulting image produced by a CCD. With a lower SNR, the state of an object becomes more difficult to determine accurately. If the SNR becomes too low, it will be impossible to determine if the object-of-interest is in the image or not.

Here, the signal-to-noise (SNR) ratio is calculated using the Merline equation, presented in Eqn. (2.36) [37]. If the SNR is high (the signal strength of the object is much greater than the noise), then the image is clear and the object's measurement state can be determined with greater accuracy, shown in Fig. 4.2a. If the SNR is low, however (the signal strength of the object is relatively close to the noise strength), then the image is hard to resolve and the object's measurement state is less accurate, as shown in Fig. 4.2b. Most of the noise effects are unchangeable, but there have been methods to reduce the effects of attenuation in the sensor tasking problem to increase the SNR [6, 25]. One such method often employed is by adding a heuristic minimum elevation constraint to the sensor. Adding this elevation constraint reduces the viewing directions that the sensor can view, but those viewing directions that were eliminated are at low elevations, where attenuation effects are highest [1, 8, 9].

Alternatively, the light attenuation can also be expressed via a rigorously defined probability of detection. This avoids introducing a rigid heuristic elevation constraint and also provides a wider framework, as also further effects, besides the atmosphere are reflected in the computation of the probability of detection, as discussed previously [25]. This paper will compare both methods and study their relationship in the single and multi-sensor tasking problem.

4.1 Minimum Elevation Constraint

This paper studies the outcomes of how a pre-defined fixed heuristic minimum elevation constraint for a ground-based sensor affects its performance in the sensor tasking problem. A changing minimum elevation constraint (h_{min}) will affect the FOR of the sensor (and a higher elevation constraint will make the FOR smaller, respectively), and consequently the total number of possible viewing directions. A comparison of how a pre-defined h_{min} changes the total possible number of viewing directions is shown in Figure 4.3.



(a) τ, δ frame with no elevation constraint (blue) and 20° minimum elevation constraint (green)

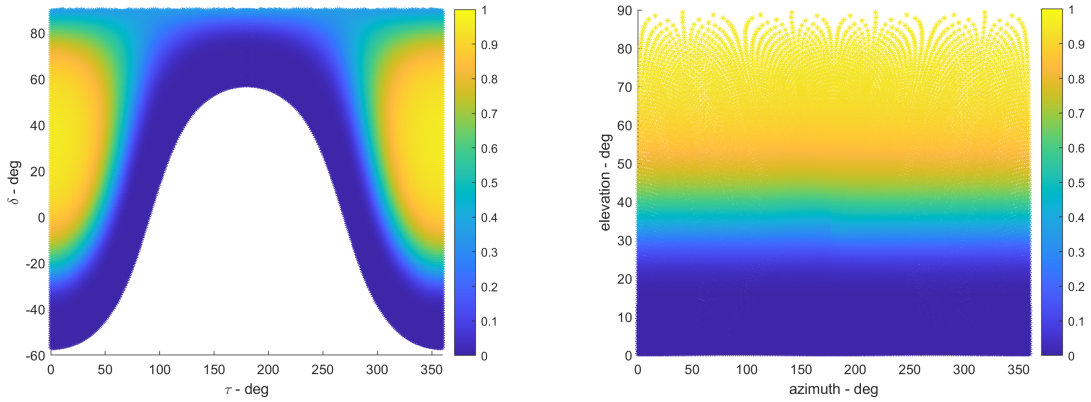
(b) α, h frame with no minimum elevation constraint (blue) and 20° minimum elevation constraint (green)

Figure 4.3. Comparison of the effects of an elevation constraint on the sensor's FOR. There is a significant decrease in possible viewing directions in the working frame when comparing no elevation constraint system (blue) versus a 20° rigid elevation constraint system (green).

The FOV for the grid-space created in Figure 4.3 is 2.5°. The sensor in Fig. 4.3 is located at a 40° latitude. The blue grids represent the viewing directions of a sensor with no elevation constraint ($h = 0°$), while the green grids represent the viewing directions of a sensor with a 20° elevation constraint. The blue sensor system of $h = 0°$ has a total of 5073 possible viewing directions. On the other hand, when increasing the minimum elevation constraint to $h = 20°$, or the green sensor system, the total number of possible viewing directions decreases to 3634. If no other factors are included when observing the RSOs, it is expected that increasing the minimum elevation constraint decreases the number of RSOs seen per observation period. This paper will investigate this scenario, as well as factoring in the probability of detection for the sensor tasking problem.

4.2 The Probability of Detection Dependence on Elevation

To illustrate the effect of elevation on the probability of detection a simulation is performed using Eqn. (2.37), presented in section 2.4. To isolate the Van Rhijn factor effect from Eqn. (2.34), the irradiance of an object ($I_{obj}(\bar{\lambda})$) is set constant. The object is modeled as a sphere with a diameter of one meter, and a diffusion reflection parameter (C_d) of 0.26 [6]. The sensor is located at a latitude of 30° and a longitude of 166.61° [28]. The sensor's FOV is a square 1.5°. This FOV was chosen to illustrate the entire FOR space. For the sensor's optic parameters, a primary radius of 0.3556 meters and a secondary radius of 0.165 meters are used. For the simulation of the sensor tasking scenarios in future sections, all of these variables will be used with the exception of the FOV being changed to a square 2.5°.



(a) P_d in the τ, δ frame with no minimum elevation constraint and constant I_{obj}

(b) P_d in the α, h frame with no minimum elevation constraint and constant I_{obj}

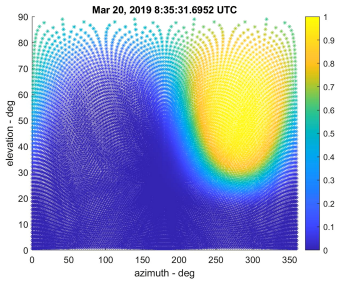
Figure 4.4. The probability of detection (P_d) in the τ, δ and α, h frames. The P_d of an object is calculated by Eq. (2.37) for each grid space with a constant $I_{obj}(\bar{\lambda})$ (from Eq. (2.32)) and shown visually by the color of the grid, with 1 being 100% probability of seeing the object, and 0 being 0% probability of seeing the object.

Figure 4.4 illustrates the Van Rhijn factor for a specific sensor. For this specific scenario, with an elevation constraint of 30°, all P_d lower than 0.245 are cut off from the FOR. If the elevation is just decreased by 5°, or an elevation constraint of 25°, then

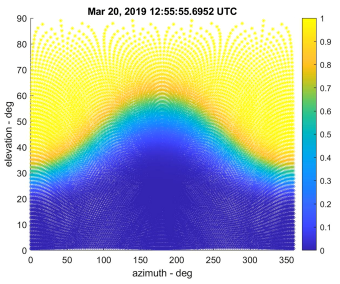
the P_d values are as low as 0.098. The following section will analyze the probability of detection with variable $I_{obj}(\bar{\lambda})$, depending on the location of the object in the grid solution space, and the location of the sun (or time of night), to try to model the real-world example as much as possible.

4.3 The Probability of Detection With Variable Irradiance

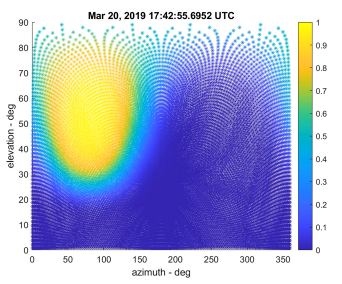
Similar to the previous section, the following results will try to analyze the probability of detection throughout the entire sensor's FOR. The difference in this section is that $I_{obj}(\bar{\lambda})$ is no longer assumed constant, and therefore will have to be calculated for each grid and at each observation time. To do this, an artificial RSO was placed at a range of 35788 km (around the range of GEO objects) from the sensor at every single grid space in the τ, δ working frame at each observation time. This was then transformed to the α, h space for a better visual representation of the correlation of elevation to P_d. The sensor used is the same as in section 4.2, and is located at a latitude of 30° and a longitude of 166.61° [28]. Fig. 4.5 shows the P_d of the sensor's solution space at the beginning, middle, and end of the observation period.



(a) P_d in the α, h frame at the beginning of the night



(b) P_d in the α, h frame in the middle of the night



(c) P_d in the α,h frame at the end of the night

Figure 4.5. Comparison of the probability of detection (P_d) as the observation nighttime changes in the α, h frame. The P_d of an object is calculated by Eq. (2.37) for each grid space with an object at a GEO range at each observation time. The object's $I_{obj}(\bar{\lambda})$ changes at each time and each grid due to the changing angle of the Sun-Object-Sensor.

Fig. 4.5b is halfway through the observation time-frame (or halfway through the night) of the sensor and is also the point in time when the average P_d of the solution space is maximum. It appears visually that there is a significant decrease in the probability of detection of a GEO object at elevation values of around 30°, with elevation values less than 30° having a much lower P_d. This is also visually shown at the beginning and end of the night in Fig. 4.5a and Fig. 4.5c. This conclusion is similar to the one in the much-simplified scenario in the previous section, though now is much more representative of the real-world observation of GEO objects.

5. REPOSITIONING TIME OF OPTICAL SENSORS

When a sensor repositions to a new viewing direction to make an observation, it must slew in the azimuth direction and in the elevation direction. The time it takes for this sensor to slew, or reposition, to the new viewing direction is called the slew time or the repositioning time. The repositioning time in this work also considers the time it takes for the sensor to fully decelerate from repositioning, which is called the settling time. When designing a sensor tasking formulation, repositioning time needs to be considered. If repositioning time is considered constant, then the problem simplifies. However, a constant repositioning time may not be a valid assumption depending on how far the sensor has to reposition to its next viewing direction. If the sensor only needs to slew by one degree, then a constant repositioning time may be over accounting for this, and more observations can be made if the sensor tasking problem did not assume a constant repositioning time. If the sensor needs to slew 170° however, a constant repositioning time may under account for this, and the RSOs that the algorithm calculated to be in that viewing direction may not actually be there, causing errors in the model. This chapter will analyze the benefits and drawbacks of both assuming a constant repositioning time, as well as calculating and using a variable repositioning time.

5.1 Constant Repositioning Time

When using the sensor tasking optimization equation developed by Frueh et al. (2018), the time between observations, or the time between m_g and m_{g-1} needs to be defined [24]. This time can be different for each sensor depending on characteristics

like the number of images taken, exposure time of those images, repositioning time, etc. The time between observations (t_{obs}) is given by the following [6, 24]:

$$t_{obs} = j \cdot t_{exposure} + (j-1) \cdot t_{readout} + t_{repos} \tag{5.1}$$

where $t_{exposure}$ is the exposure time for each image j. $t_{readout}$ is the readout time of each image, and t_{repos} is the time for the sensor to reposition to the specific viewing direction $h_{f,g}$ in the sensor's FOR. In this work, it is assumed that the readout time is smaller than the constant repositioning time, or $t_{repos} > t_{readout}$. This assumption allows for one of the images to be read while the sensor is repositioning, hence why in Eqn. (5.1) the readout time is multiplied by $(j-1)$ [24]. Eqn. (5.1) can be rewritten to separate the repositioning time from the image creation/processing time as follows:

$$t_{obs} = t_{img} + t_{repos} \tag{5.2}$$

where $t_{img} = j \cdot t_{exposure} + (j-1) \cdot t_{readout}$. Eqn. (5.2) is used regardless of if the repositioning time is constant or not, and for this work, it is assumed that t_{img} is constant, though can be different for each sensor.

If t_{repos} is set constant, then the simulation becomes computationally efficient. All the RSOs' states can be calculated beforehand at each m_g for each sensor, because of the constant repositioning time. However, if the repositioning time is not varying, then simulation errors can occur. This work assumes two cases when comparing the constant repositioning time:

1. **Case 1**: When the repositioning time is constant and is not being compared to the variable repositioning time, then the sensor tasking performance is the true performance in the simulation, but it creates a model mismatch when applied to an actual sensor.

2. **Case 2**: When the constant repositioning solution is directly compared to the variable repositioning time solution, which is the most realistic simulation to the actual sensor, the optimizer solution is analyzed further:

   a. **A**: If the constant assumed for the repositioning time is smaller than the actual variable repositioning time for that m_g, the sensor does not view the objects at the time. The optimizer knows this and will account for these "missed" objects in future viewing direction considerations.

   b. **B**: If the constant assumed for the repositioning time is smaller than the actual variable repositioning time for that m_g, the sensor does not view the objects at the time. The optimizer does not know this and assumes it detected the "missed" objects, even though they were never detected.

Case 1 will be used to analyze the results in section 7.2, while Case 2 will be used to analyze the results in section 7.3.

5.2 Variable Repositioning Time

The problem becomes significantly more complex if a constant repositioning time is not assumed. If the repositioning time is different for each viewing direction, then this increases the computational time of the simulation. Each RSO must now be propagated to the "pseudo" time, it takes to get to a specific viewing direction. The "pseudo" time for each viewing direction isn't the actual time that the observation is taken, but is the time the observation would be taken if the algorithm chooses this specific viewing direction. This is done for every single viewing direction because each viewing direction has its own repositioning time. To increase computational efficiency, RSOs that have a measurement uncertainty near the viewing direction being analyzed (a measurement standard deviation of 10) are propagated to that "pseudo" time instead of all RSOs for that specific viewing direction. Then, the viewing direction grid is chosen based on Eqn. 3.15, and the objects are actually

propagated to the true chosen time for this specific viewing direction. This process is repeated until the observation period is completed. With the variable repositioning time, the simulation computational complexity now increases with the size of the FOV of that specific sensor: a smaller FOV equates to more viewing directions and therefore more "pseudo" times that need to be calculated, and therefore more propagation times for the RSOs (these propagation times and computational considerations are discussed in chapter 6).

5.2.1 Repositioning Equation Formulation

A function to calculate the repositioning time based on the change in viewing direction also needs to be defined. The time for the sensor to reposition from grid at time $(f-1)$ to grid at time (f), or $t_{repos}(h_{f,g})$ is given below:

$$t_{repos}(h_{f,g}) = \zeta\sqrt{\max(\mid \Delta\alpha \mid, \mid \Delta h \mid)} \tag{5.3}$$

Here, the change in azimuth and elevation ($\Delta\alpha$ and Δh respectively) is given as the form of $\Delta\alpha = \alpha_f - \alpha_{f-1}$ and $\Delta h = h_f - h_{f-1}$ in units of degrees. ζ is a constant that scales how long the repositioning time takes. It is assumed that as the sensor is repositioning in the azimuth direction it is also repositioning in the elevation direction at the same time, hence why only the maximum the combined viewing direction change is considered. This model is used to try to account for the ramp-up and ramp downtime of the sensors slew. Once the sensor has reached maximum velocity, it is assumed that the speed will remain constant until the sensor arrives close to the new viewing direction, then slow down. The actual real-world repositioning system will be more complex. An example of the complexity deals with this ramp-up time. If there is a small repositioning angle, the sensor could spend the entire repositioning time either ramping up in slew rate and then ramping down in slew rate. This would also be a sensor-specific case, depending on how fast the sensor can ramp-up in slew rate. For this analysis, the simple variable repositioning

model shown in Eqn. (5.3) tries to model the ramp-up and ramp downtime in a computationally efficient way.

Eqn. 5.3 needs to be verified to see how well the repositioning time is modeled in an actual sensor. To do this, the repositioning time for the Purdue Optical Ground Station (POGS) was calculated for a single night. POGS on the night of May 19^{th}, 2020 was tasked with tracking and taking images of GPS objects throughout the night. The repositioning times were taken from the data, and all other data, such as following the object and image processing, has been removed to some extent. There were some factors, such as sensor rest time and amount of time between tracking and actually repositioning to a new viewing direction, that could not be removed from the analysis. Fig. 5.1 shows the actual repositioning data from POGS, the expected repositioning time using Eqn. (5.3), and a least-squares best fit model in the solution space of Eqn. (5.3).

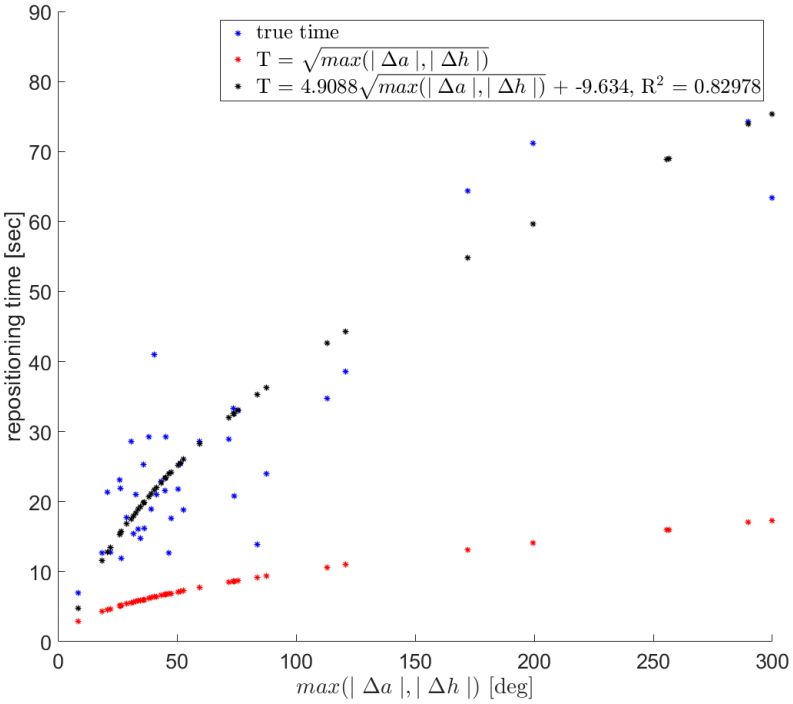


Figure 5.1. Repositioning time vs. the max slew angle for Purdue Optical Ground Station (POGS) tracking GPS objects on the night of May 19$^{\text{th}}$, 2020. There is a reasonable correlation between the actual repositioning time data and a repositioning equation in the solution space of Eqn. (5.3).

There are several takeaways from Fig. 5.3. The first one is that there is a solution that decently models the repositioning time for POGS on this night in the solution space of Eqn. (5.3). This model, shown in black, has a ζ value of 4.9 and a separate constant value of -9.634. The black least-squares best fit model also has a R^2 value for the blue true repositioning date of 0.82978 on this specific night. The separate constant, along with the lower R^2 than desired can be attributed to two main reasons. The first reason could be that there is a better model to account for the repositioning time, though it is unknown what this model could be. A second reason has to do with the fact that these blue repositioning times may hold other information besides just the repositioning times, such as time-tracking a specific object, and times where the sensor is sitting idle before moving to a new object. Though these errors need to

be noted, the actual R^2 for this best fit is large enough for the solution space from Eqn. (5.3) to model the repositioning time fairly well [76].

To confirm that Eqn. (5.3) is the optimal repositioning equation to use in this work, several other repositioning equations were tested similar to the process shown in Fig. 5.1. These equations are presented in Fig. 5.2 - 5.4.

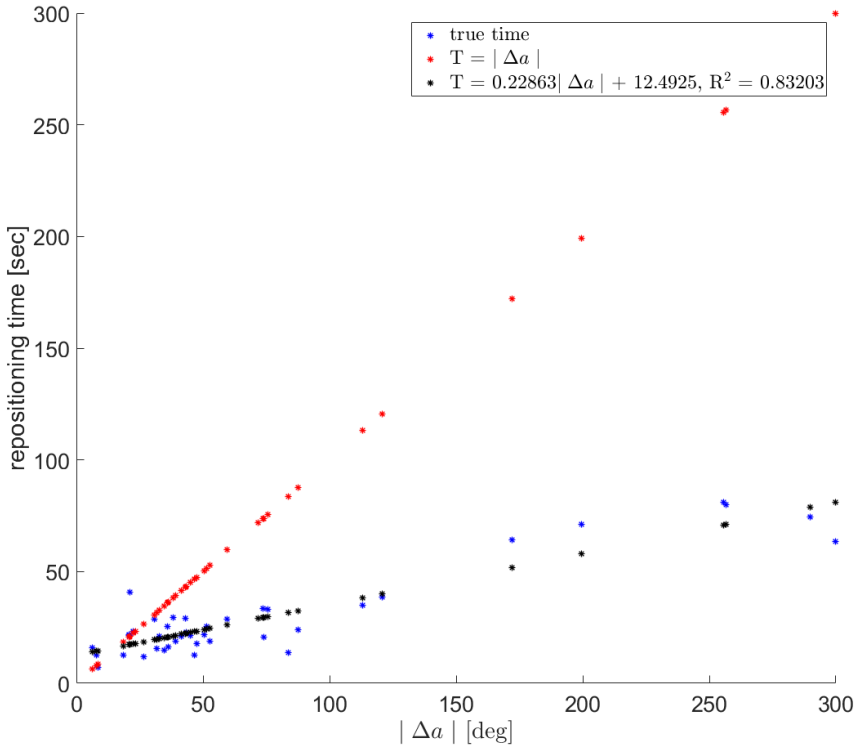


Figure 5.2. Repositioning time vs. the max slew angle for Purdue Optical Ground Station (POGS) tracking GPS objects on the night of May 19$^{\text{th}}$, 2020. The solution space equation for this analysis is of the form $t_{repos}(h_{f,g}) =\mid \Delta\alpha \mid$

Using the equation $t_{repos}(h_{f,g}) =\mid \Delta\alpha \mid$ as the solution space to model the repositioning results actually gives a slightly better R^2 value for the data at $R^2 = 0.83203$. This value is not significantly better, and if the viewing directions only changed in the elevation direction, then this model would assume that the repositioning time was zero, which is not physically possible. Fig. 5.2 does highlight the large dependence

on azimuth and the small dependence on elevation in the time it takes for the sensor to reposition.

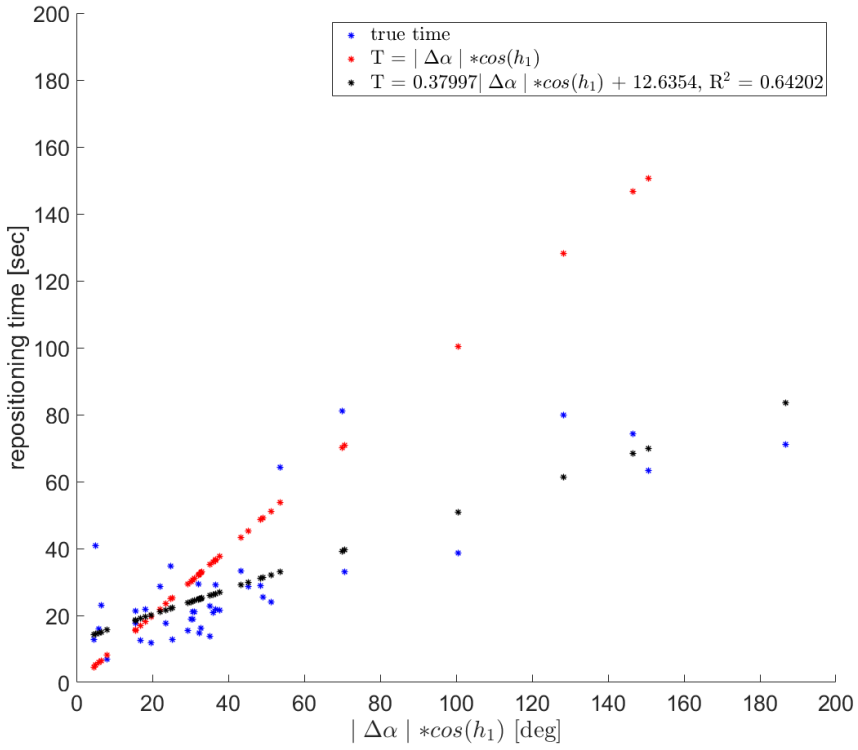


Figure 5.3. Repositioning time vs. the max slew angle for Purdue Optical Ground Station (POGS) tracking GPS objects on the night of May 19$^{\text{th}}$, 2020. The solution space equation for this analysis is of the form $t_{repos}(h_{f,g}) =\mid \Delta\alpha \mid *cos(h1)$

Fig. 5.3 tries to fold in some elevation dependence into the repositioning model. The equation used to model the repositioning time has a solution space form of $t_{repos}(h_{f,g}) =\mid \Delta\alpha \mid *cos(h1)$. The term $cos(h1)$ is added to account for the fact that at higher elevations, the time for the sensor to reposition in the azimuth direction should be shorter. This assumption is due to the smaller slewing distance the sensor has to travel in the azimuth direction at higher elevations. from Fig. 5.3 shows that assuming this fact does not produce more accurate results, with an R^2 value of 0.64202.

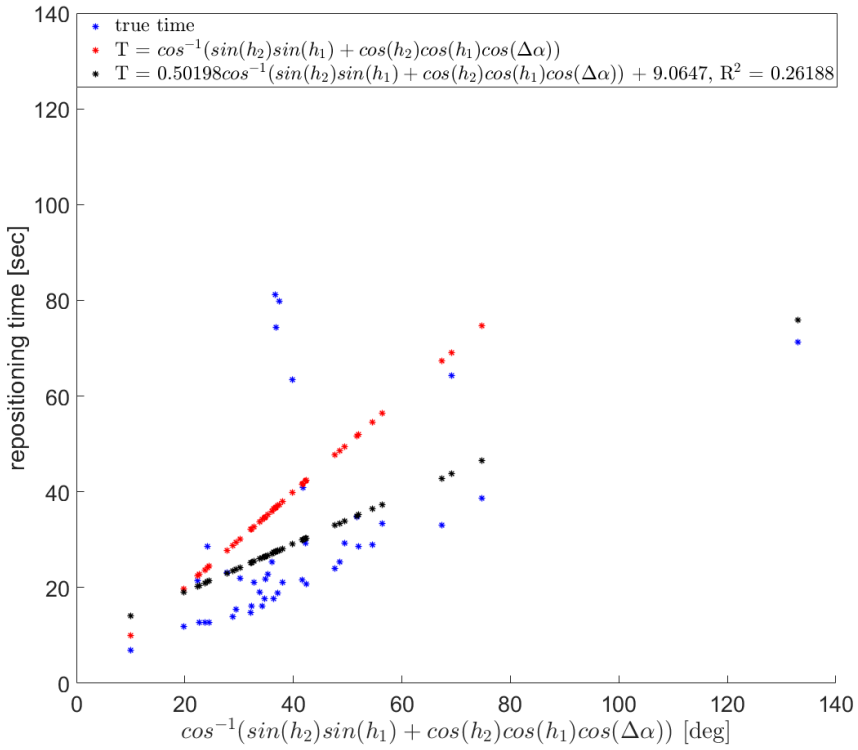


Figure 5.4. Repositioning time vs. the max slew angle for Purdue Optical Ground Station (POGS) tracking GPS objects on the night of May 19$^{\text{th}}$, 2020. The solution space equation for this analysis is of the form $t_{repos}(h_{f,g}) = cos^{-1}(sin(h_2)sin(h_1) + cos(h_2)cos(h_1)cos(\Delta\alpha))$

Finally, the great circle distance travelled to reposition is tested. The actual angular distance travelled by the sensor is given by $t_{repos}(h_{f,g}) = cos^{-1}(sin(h_2)sin(h_1) + cos(h_2)cos(h_1)cos(\Delta\alpha))$, and is shown in Fig. 5.4 shows that for this scenario, the great circle equation does not model the repositioning time accurately, with only a $R^2 = 0.26188$ value.

From analyzing Fig. 5.1 - 5.4, the model used in Fig. 5.1, or Eqn. 5.3, is used. Considering only the azimuth as a repositioning equation did produce a more accurate model, but leaves out repositioning scenarios that Eqn. 5.3 accounts for.

5.2.2 Optimizing for a Variable Repositioning Scenario

From Fig. 5.1, two scenarios will be tested in the variable repositioning analysis, shown in section 7.3. The first scenario will be using Eqn. (5.3), with $\zeta = 1$. This scenario is used to try to understand the affects of a variable repositioning time when the t_{repos} is not significantly larger than t_{img}, from Eqn. (5.2). The other scenario will be using Eqn. (5.3), with $\zeta = 5$. This ζ value was derived from Fig. 5.1.

The optimization strategy can easily be added into the formulation developed by Frueh et al. (2018). Modifying Eqn. (3.15) to consider the repositioning time weighting function $\gamma_{repo}(h_{f,g})$, the following is derived:

$$\max A = \sum_{g=1}^{l} \sum_{f=1}^{m_g} \left(\sum_{i=1}^{n} \left(\mu(\tilde{X}_i) \cdot \ P_d(h_{f,g}, \tilde{X}_i) \cdot \ d(h_{f,g}, \tilde{X}_i, P_i) \right) \cdot \gamma_{repo}(h_{f,g}) \right) \tag{5.4}$$

Where $\gamma_{repo}(h_{f,g})$ is the repositioning time weight function for all viewing directions in the FOR for sensor g. $\gamma_{repo}(h_{f,g})$ can be defined as any function that considers the repositioning time of the sensor. The following equation is used in this work to define $\gamma_{repo}(h_{f,g})$:

$$\gamma_{repo}(h_{f,g}) = \begin{cases} 1 & 0 \leq t_{repo}(h_{f,g}) < 1 \\ \dfrac{1}{C \cdot t_{repos}(h_{f,g})} + (1 - \dfrac{1}{C}) & 1 \leq t_{repo}(h_{f,g}) \end{cases} \tag{5.5}$$

$$C \geq 1 \tag{5.6}$$

$$\gamma_{repo}(h_{f,g}) = \begin{cases} 1 & 0 \leq t_{repo}(h_{f,g}) < 1 \\ \dfrac{-t_{repos}(h_{f,g}) + max(t_{repos}(h_{f,g}))}{max(t_{repos}(h_{f,g})) - 1} & 1 \leq t_{repo}(h_{f,g}) \end{cases} \tag{5.7}$$

$t_{repo}(h_{f,g})$ is calculated using Eqn. (5.3). Because an optimal $\gamma_{repo}(h_{f,g})$ function is unknown, several are tested in this work. C is a constant value that changes the penalty of the sensor choosing viewing directions that have a high repositioning time. A visual representation of Eqn. (5.5) - (5.7) is shown in Fig. 5.5 to highlight the affects of the constant C.

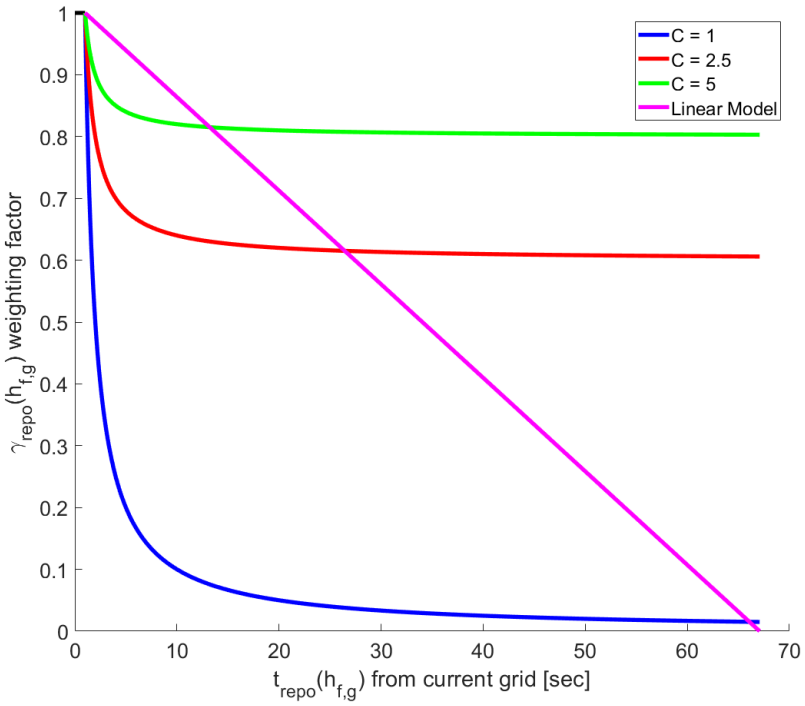


Figure 5.5. γ_{repo} Factor for Various C Values for $\zeta = 5$ from Eqn. (5.3) and Eqns. (5.5) - (5.7). Depending on the maximum repositioning time, this penalty can be significant and changes the linear model representation.

With a lower C value, the optimizer will prioritize repositioning times that are significantly shorter and viewing directions that are close to the current one. If the C value increases, the optimizer will still value shorter repositioning times over longer ones, but with less of an emphasis. If Eqn. 5.3 is used assuming $\zeta = 1$, then the max repositioning time for the system is around 13.5 seconds. However, if Eqn. 5.3 is used assuming $\zeta = 5$, then the max repositioning time for the system is around 67 seconds.

Because the linear optimizer model is derived using the maximum repositioning time, the linear model shown in Fig. 5.5 is for $\zeta = 5$, and would be different if $\zeta = 1$.

The varying C values and linear model are analyzed and tested for the two scenarios: $\zeta = 1$ and $\zeta = 5$. These are tested in this section using a simplified sensor setup to test model validity and then are analyzed in the actual sensor problem shown in section 7.3. Before the optimizers and variable repositioning can be tested, a few assumptions need to be defined. First, it is assumed that the sensor does not need to "unwind." This assumption means that the sensor will never reposition more than 180° in the azimuth direction. Fig. 5.1 shows that there are cases in the real world where the sensor needs to unwind, but for this work, it is assumed that the sensor can ignore this, as it is difficult to model the cases when the sensor needs to unwind. Another assumption that is made is that the sensor slews at the same rate for all directions of motion. This simplifies the problem significantly and allows for faster computational times, but in the real world, the sensor can slew at varying rates depending on the direction that it occurs.

As stated above, a simple variable repositioning problem is analyzed. A sensor is placed with a latitude of -30°, the longitude of 136.78°, and an altitude of 2.221 kilometers. To speed up computational time in this test case, two assumptions that deviate from the actual sensor used in section 7.3 are made. First, the FOV of the sensor is increased from the true FOV = 2.5° to a larger FOV = 6°. This significantly reduces the amount of viewing directions and the object states that need to be calculated before the optimizer chooses the actual viewing direction. The other assumption is a large t_{img} (presented in Eqn. 5.2). In this test case, t_{img} is significantly large to decrease the total amount of viewing directions per observation period and speed up computational time. For this test case, $t_{img} = 94$ seconds, while in the actual sensor results presented in section 7.3, $t_{img} = 4$ seconds. The relationship between t_{img} and t_{repo} is important, and will be discussed below.

First, the weighted functions are compared to the non-weighted repositoning time solution for this specific test case. Fig. 5.6 shows the results assuming $\zeta = 1$ from Eqn. (5.3).

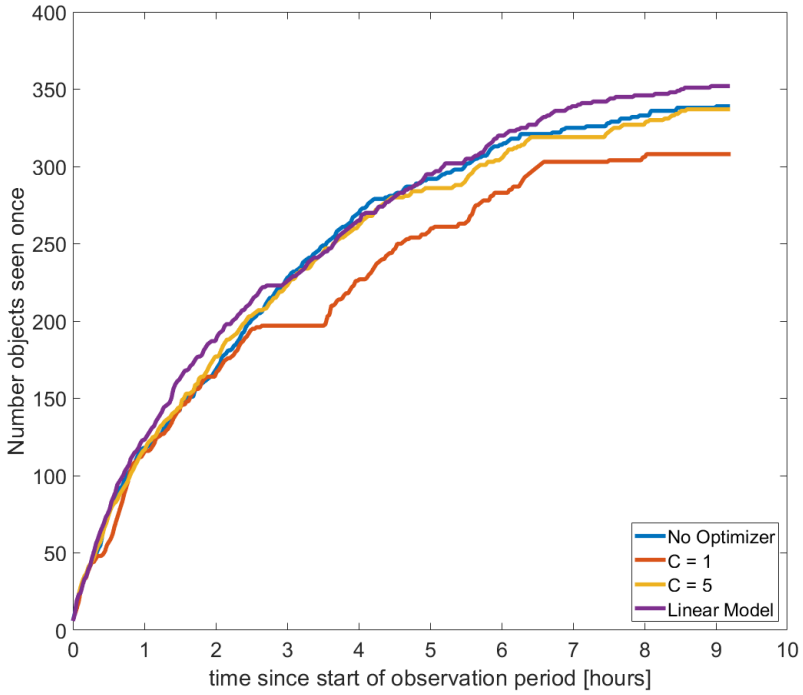


Figure 5.6. Various Repositioning Optimizers for $\zeta = 1$. Using Eqns. (5.5) - (5.7) on the sensor described above, the optimizers are compared to the non-optimized, variable repositioning model.

Fig. 5.6 shows a few results. First, it shows that an "optimizer" could produce worse results than the non-weighted solution if the weighted optimized solution is not developed accurately. This is the reason why several weighted factors are tested. Secondly, Fig. 5.6 shows that there is at least one optimizer in this test case that produces better results than the non-weighted solution, which is the linear model. This provides some validity to both the weighted optimizers that are used in this paper, as well as the need to optimize for the repositioning time in the first place. In this scenario where $\zeta = 1$, the maximum repositioning time is around 13.5 seconds. Compared to t_{img} in this case, that is not too significant. This decreases the impact that

is made when the optimizers choose a viewing direction that has a lower repositioning time and might be a reason as to why the optimizers are not performing better or only slightly better than the non-weighted solution. This factor will be considered when analyzing the true sensor scenario in section 7.3.

Now, $\zeta = 5$ is analyzed in this test case, and is shown in Fig. 5.7.

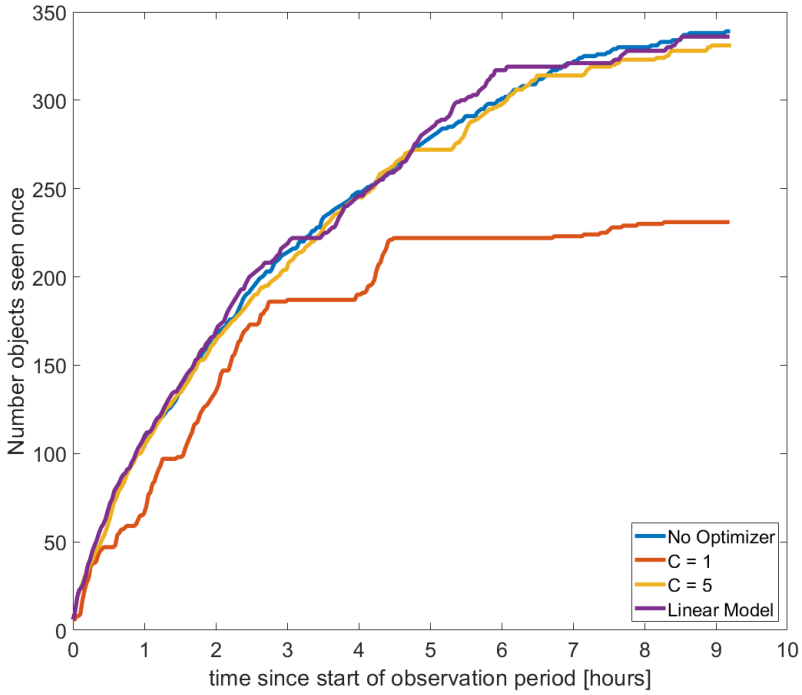


Figure 5.7. Various Repositioning Optimizers for $\zeta = 5$ on the sensor described above. The optimizer results change depending on how significant the repositioning time is, calculated by Eqn. (5.3).

Now, the maximum repositioning time for Fig. 5.7 is around 67 seconds. This is still below the t_{img} time in this test case, which is 94 seconds, but now the repositioning time is not significantly smaller, making it a more important factor. In the actual sensor in section 7.3, the maximum repositioning time for both $\zeta = 1$ and $\zeta = 5$ will be larger than the assumed t_{img}. For this test case, however, there are some interesting results. Like in Fig. 5.6, When $C = 1$, the sensor performs the worse, and with $\zeta = 5$, the performance is significantly worse. For $\zeta = 5$ in the test case,

however, no weighted optimizer performs better than the non-weighted solution. This might suggest that the sensor tasking solution does not need to consider the variable repositioning times in the optimizer part, but that conclusion will be fully explored in the real sensor in section 7.3, and could change when the maximum repositioning time is larger than t_{img}.

6. COMPUTATIONAL TIME VS. SIMULATION ACCURACY

The simulation of the RSOs and optimization algorithm used to compute Eqn. (3.15) are computationally expensive [6, 24]. Little and Frueh (2019) found that using the greedy algorithm produces good results in maximizing Eqn. (3.15), while also being very computationally efficient [6] for a single sensor system. When the problem's complexity is increased from a single sensor to the multi-sensor problem, or the 34 sensor system developed by Ackermann et al. (2018) to explore complete geosynchronous region coverage in a single day, finding a computationally efficient method is vital to solving the sensor tasking problem [28].

There are two main ways to improve the computational performance of the sensor tasking problem. Because of the way the sensor tasking algorithm presented by Frueh et al. (2018), parallelization of the simulation code is an easy way of improving the run-time of the simulation [24]. However, the work that is done by each core still is computationally expensive. Another method of improving run-time needs to be developed. This other method is a trade-off between simulation accuracy and run-time of the simulation.

6.1 Object's Dynamics Simulation

To simulate the dynamics of the objects, MATLAB's ODE45 function is used. This function uses the Runge-Kutta integration method to solve the dynamics of the problem, in this case, the two-body problem as well as propagating the mean and covariance of the EKF [75]. The relative tolerance and absolute tolerance for the integration steps can be set in ODE45. These tolerances inform the integrator of how accurate each integration step must be. Therefore, a smaller tolerance, for

example: Tolerance $= 1 \times 10^{-16}$, produces results closer to the "true" two-body solution than a solution integrated with a Tolerance $= 1 \times 10^{-8}$. However, because the smaller tolerances produce more accurate results, they generally take a longer time to integrate the object's state and therefore are more computationally expensive.

This section compares the computational time spent on simulating 1211 RSOs vs. the accuracy of said simulation for two scenarios: The first scenario is for a sensor that has $t_{obs} = 163.75$ seconds from Eqn. (5.1), or a total number of observations of $m_g = 194$ for the night, while the second scenario is for a sensor that has $t_{obs} = 17.5$ seconds from Eqn. (5.1), or a total number of observations of $m_g = 1946$ for the night. The large difference was chosen to try to understand the effects of rounding and truncation that occurs with a larger amount of observation times versus a smaller amount of observation times. All simulations were ran on the Space Information Dynamics (SID) group's Linux Server[1] using MATLAB®[2]. For both cases the relative and absolute tolerances are equal, and it is assumed that a Tolerance $= 1 \times 10^{-16}$ is the "true" state of the object, i.e. the accuracy of the simulation cannot be better than at this tolerance level.

6.1.1 Simulation Analysis for Sensor With $m_g = 194$

This first scenario analyzes the object simulation for a sensor that has a total of $m_g = 194$ observations a night. This scenario was chosen to see if there was a dependence on run-time and accuracy for a simulation that propagates the RSOs for a long period of time but does not stop and record the object's state as much as other sensors. Table 6.1 and shows the results for this case:

[1]Red Hat Enterprise Linux Workstation CentOS Linux release 7.8.2003, Copyright 2011-2020 Red Hat, Inc., x86_64, 32 Intel® Xeon® E5-2630 v3 (2.40GHz)
[2]R2018b Update 3, 64-bit (Copyright 1984-2018 The MathWorks, Inc.)

Table 6.1. Comparison of the simulation run-time and error vs. Tolerance (Tol) in the propagation of the RSOs, with a total of $m_g = 194$ viewing directions for the night. The total number of RSOs simulated was 1211. The errors shown are the total errors of the RSO's position and velocity states at the end of the observation period.

Tol Level	t_{run} per RSO	t_{run} per m_g	mean $\mid \overline{e}_{pos} \mid$	mean $\mid \overline{e}_{vel} \mid$
Tol $= 1 \times 10^{-16}$	9 seconds	3 hours	0	0
Tol $= 1 \times 10^{-12}$	.7 seconds	14.1 minutes	$6.5 \times 10^{-4}\ m$	$4.7 \times 10^{-8}\ \frac{m}{s}$
Tol $= 1 \times 10^{-10}$	.1 seconds	2 minutes	$3.5 \times 10^{-4}\ m$	$2.6 \times 10^{-8}\ \frac{m}{s}$
Tol $= 1 \times 10^{-9}$	.01 seconds	12.1 seconds	$5.4 \times 10^{-4}\ m$	$4 \times 10^{-8}\ \frac{m}{s}$
Tol $= 1 \times 10^{-8}$	.005 seconds	6 seconds	$2.3 \times 10^{-3}\ m$	$1.8 \times 10^{-7}\ \frac{m}{s}$
Tol $= 1 \times 10^{-7}$	.002 seconds	2.4 seconds	$.01\ m$	$5.1 \times 10^{-7}\ \frac{m}{s}$
Tol $= 1 \times 10^{-6}$	.001 seconds	1.2 seconds	$0.17\ m$	$6.7 \times 10^{-6}\ \frac{m}{s}$

Table 6.1 shows the simulation run-time and associated errors at the end of the observation period for the RSOs, as they change with the tolerance level. As expected, as the level of accuracy of the simulation decreases (or as the magnitude of the tolerance level increases), so does the simulation run-time. The decrease is not one-to-one, however. The benefit gained in simulation run-time from a higher accuracy simulation to a lower one decreases. If no other metric is considered in the problem, then choosing a Tolerance $= 1 \times 10^{-6}$ would be optimal. However, the errors for each of these tolerances need to be analyzed before choosing an optimal tolerance level. The error for this section and for section 6.1.2 are evaluated by running the RSOs states at Tolerance $= 1 \times 10^{-16}$ and then subtracting them from the desired response. In both sections, this error at the end of the observation period is averaged for each 1211 RSO and is shown. As stated above, Tolerance $= 1 \times 10^{-16}$ is assumed to be the best possible simulation, hence why the objects' mean state errors are zero. All other errors are compared to this tolerance level. The level of error increases as the accuracy of the simulation decreases. The max error at the end of the observation

period occurs at Tolerance = 1×10^{-6}, with the mean error in the 1211 simulated objects being around 0.17 meters. The RSOs that were examined here, as well as the rest of this paper, are in the GEO regime, so an error of 0.17 meters in position isn't too significant. From Table 6.1, setting a tolerance level of Tolerance = 1×10^{-7} or Tolerance = 1×10^{-6} is the optimal choice in simulation accuracy versus simulation run-time in this case.

6.1.2 Simulation Analysis for Sensor With $m_g = 1946$

Now, this scenario analyzes the object simulation for a sensor that has a total of $m_g = 1946$ observations a night. This scenario was chosen to see if there was a dependence on run-time and accuracy for a simulation that propagates the RSOs for a short period of time and stops and record the object's state often. This scenario is more representative of the sensors developed by Ackermann et al. (2018) [28]. Table 6.2 shows the results for this case:

Table 6.2. Comparison of the simulation run-time and error vs. Tolerance (Tol) in the propagation of the RSOs, with a total of $m_g = 1946$ viewing directions for the night. The total number of RSOs simulated was 1211. The errors shown are the total errors of the RSO's position and velocity states at the end of the observation period.

Tol Level	t_{run} per RSO	t_{run} per m_g	mean $\mid \overline{e}_{pos} \mid$	mean $\mid \overline{e}_{vel} \mid$
Tol= 1×10^{-16}	5 seconds	1.68 hours	0	0
Tol = 1×10^{-12}	1 second	20 minutes	3×10^{-4} m	2.7×10^{-8} $\frac{m}{s}$
Tol = 1×10^{-10}	.07 seconds	1.4 minutes	4.5×10^{-4} m	2.8×10^{-8} $\frac{m}{s}$
Tol= 1×10^{-9}	.01 seconds	12.11 seconds	1.6×10^{-3} m	1.2×10^{-7} $\frac{m}{s}$
Tol= 1×10^{-8}	.002 seconds	2.4 seconds	2.3×10^{-3} m	1.7×10^{-7} $\frac{m}{s}$
Tol= 1×10^{-7}	.001 seconds	1.2 seconds	6×10^{-3} m	3.8×10^{-7} $\frac{m}{s}$
Tol = 1×10^{-6}	.0009 seconds	1.1 seconds	.08 m	3.6×10^{-6} $\frac{m}{s}$

A similar trend is shown in Table 6.2 that was also present in Table 6.1 for the previous scenario: As the level of accuracy decreases, so does the run-time. The difference, in this case, is the rate of change of the simulation run-time. In the scenario presented in subsection 6.1.1, as the level of accuracy changed from a higher accuracy to a lower one, the amount of time the simulation took to propagate a single RSO decreased by at least a factor of two, if not more. However, the change shown in Table 6.2 stops following this trend when the tolerance is changed from Tolerance $= 1 \times 10^{-7}$ to Tolerance $= 1 \times 10^{-6}$, and only a small change in run-time is seen. As for the errors shown in this scenario, Table 6.2 has similar trends as in Table 6.1: decreasing accuracy of the simulation increases the error in the object's final state. The error in the state at Tolerance $= 1 \times 10^{-6}$ is also on the same order of magnitude. Looking back at Tolerance $= 1 \times 10^{-7}$ and Tolerance $= 1 \times 10^{-6}$, the increase in error is nearly an order of magnitude, while the run-time difference is significantly less. Because of this, a Tolerance $= 1 \times 10^{-7}$ is better suited for this scenario, as it reduces the errors in the simulation, while still maintaining decent computational efficiency.

Between the two scenarios, it was found that either Tolerance $= 1 \times 10^{-7}$ and Tolerance $= 1 \times 10^{-6}$ was optimal, shown for a sensor with $m_g = 194$ observations per night in section 6.1.1, or just a Tolerance $= 1 \times 10^{-7}$, shown for a sensor with $m_g = 1946$ observations per night in section 6.1.2, was optimal. Considering both of the scenarios, it was concluded that for this work, an absolute and relative Tolerance $= 1 \times 10^{-7}$ produces accurate and computationally efficient results.

6.2 Simulating the Probability of Detection

The other aspect of the simulation of the sensor tasking problem that can be analyzed for computational efficiency is the calculation of the probability of detection. The probability of detection or P_d is calculated for each RSO using Eqn (2.37). This equation assumes that the summation term n will tend towards infinity [25]. In this case, the probability of detection will be completely calculated for the object without

errors. However, it is impossible to let $n = \infty$, as the simulation will never complete. Therefore, some real value of n must be chosen to allow the calculation of P_d to be done in a reasonable amount of time. A maximum n value is chosen to calculate the true P_d value. Little and Frueh (2019) used $n = 8000$ in the calculation of P_d in the single and double sensor tasking problem, so $n = 8000$ is chosen as the maximum n. Further, lower values of n are analyzed, and the mean, standard deviation, and maximum error for all 1211 RSOs after each observation period are compared to $n = 8000$, shown in Table 6.3. These errors are then compared to the run-time for the simulation to propagate each RSO, using Tolerance $= 1 \times 10^{-7}$ for $m_g = 194$, and the run-time to calculate each object's P_d.

Table 6.3. Comparison of the simulation run-time vs. the accuracy of the P_d calculation, using Eqn. (2.37). The error is shown for the average, max, and standard deviation for all 1211 RSOs after each observation period.

P_d Analysis	t_{run} for 1211 RSOs	Mean Error in P_d	σ_d	Max Error in P_d
n = 8000	3.9 seconds	0	0	0
n = 6000	3.9 seconds	0	0	0
n = 4000	3.85 seconds	0	0	0
n = 2000	3.85 seconds	0	0	0
n = 1000	3.85 seconds	.35	.42	.98
n = 100	3.85 seconds	.35	.42	.99

Table 6.3 shows the accuracy of the P_d calculations using Eqn. (2.37) for each RSO. For each value of n, all RSOs have their P_d calculated, and then compared to the P_d calculations for $n = 8000$. Because P_d is a probability calculated with a range of zero to one, the maximum error for each RSO can only be one. The mean of this error, standard deviation, and the maximum error is computed.

As n decreases from 8000 to 2000, there is no error detected, while the run-time decreases by a minuscule amount. When n decreases further to $n = 1000$ and smaller, the error increases significantly, while the run-time remains the same. If n

was decreased further, the run-time is expected not to decrease significantly, while the errors are expected to increase.

From Table 6.3, it appears that changing the n value from Eqn. (2.37) has little affect on the simulation run-time. Table 6.3 suggest that $n \geq 2000$ is optimal. Because this work only analyzed this single case, this paper assumes $n = 8000$ to ensure accuracy with low computational penalty.

7. RESULTS

The sensor tasking optimization framework formulated by Frueh et al. (2018) and presented in chapter 3 is used to compare results from the single and multi-sensor tasking problems in this section [24]. A simple greedy algorithm introduced in Chapter 3 is used to optimize the sensor tasking framework equation in this chapter [6]. Finally, the non-weighted and the additional variable repositioning weight factor on the sensor tasking optimization framework formulated by Frueh et al. (2018) and introduced in chapter 5. The results of this chapter are split into two primary sections: A constant repositioning assumption and a variable repositioning framework.

In the first section of this chapter, or section 7.2, a constant repositioning time between viewing directions is assumed, regardless of how far the sensor has to slew. The results of this section are assumed to be the true results, and independent of if the constant repositioning time assumption is too small compared to the calculated repositioning time discussed in chapter 5. In this section, an analysis of the probability of detection versus a heuristic minimum elevation constraint is conducted. First, the single sensor scenario is analyzed with the probability of detection, assuming the mean state of the object is the true state. Then, the multi-sensor tasking scenario developed by Ackermann et al. (2018) is analyzed with the probability of detection and assuming the mean state is the true state of the objects [28]. The multi-sensor framework is compared with two different heuristic minimum elevation constraints to analyze the relationship between an elevation constraint and the probability of detection, and the multi-sensor tasking framework is tested to see if the framework can detect all GEO objects for the night. Finally, the same analysis is conducted, but now uncertainties in the object's states are considered in the analysis.

In the section section of this chapter, or section 7.3, the variable repositioning problem discussed in chapter 5 is analyzed. First, the single sensor solution is ana-

lyzed with a variable repositioning equation without any weight values applied to the repositioning time. This unweighted scenario is calculated for both a repositioning framework where the maximum repositioning time (t_{repo}) is not significantly larger than the image processing time (t_{img}), and a repositioning framework where the maximum repositioning time (t_{repo}) is significantly larger than the image processing time (t_{img}), represented in Eqn. (5.2). These results are compared to a solution where the repositioning time is assumed constant. Finally, the two repositioning frameworks are analyzed with several different weighted repositioning functions described in chapter 5.

7.1 Simulation Setup

Before the results can be presented, the sensor and simulation properties need to be defined. The following simulation constants are split into two sections: resident space object-specific constant, and sensor-specific constants.

7.1.1 Resident Space Objects Used for the Analysis

For the object population of the RSOs, a two-line element (TLE) catalog of the night of March 19$^{\text{th}}$, 2019 was chosen for the initial states of the RSOs to try to minimize the differences in observation periods between different latitudes. This catalog was then reduced to analyze just GEO objects, assuming that GEO objects exist with a semimajor axis of $39,000\text{km} \leq a \leq 44,000\text{km}$ and an eccentricity of $e \leq 0.5$. The initial mean state for the RSO is assumed to be the state given by the TLE set, and the initial covariance is defined for all RSOs from Eqn. (3.14). As for the RSO specifics, although there is a varying degree of size, shape, and reflection properties, a constant set is defined for all of them, as precise information is mostly absent [6, 30]. All RSOs are modeled as spherical objects with a diameter of one meter, and a diffusion reflection parameter (C_d) of 0.26. The average wavelength of light reflecting off the RSO is assumed to be 600 nanometers, and the extinction

coefficient is assumed to be 0.56. As for the Extended Kalman Filter, the process noise is assumed to be 1×10^{-6} and the measurement noise value is assumed to be one-arcsecond [6].

7.1.2 Ground-Based Sensor Constants

For the Ground-based Sensors, there are several constants that are assumed to be the same for all sensors. First, the FOV used in studying the multi-sensor setup from Ackermann et al. (2018) is assumed to be 2.5°. The readout time of the images is also assumed from the study to be zero seconds, the exposure time (or Δt from Eqn. 2.37) is set to 0.25 seconds, and the number of images taken for each observation time is 16 [28]. As for the optics, the primary radius is 0.3556 meters, the secondary radius is 0.165 meters, and the camera gain is set to 1.4. The pixel scale in the CCD is 0.72 arcseconds per pixel, and the Quantum Efficiency is set to 0.6. As for the noise properties for calculating the SNR of an image, the readout noise is set to 10 and the dark noise is set to 5. The total number of background pixels per image is assumed to be 2000, and the intensity of light for the background is assumed to have a mean value of 1000 W/m^2 [6, 25]. These values correspond with Purdue Optical Ground Station (POGS). The constant repositioning time and heuristic minimum elevation constraint used in section 7.2 will be presented as the results are shown, as they vary depending on the specific scenario.

The location for the multi-sensor scenarios is taken from Ackermann et al. (2018). This multi-sensor system was developed to observe all GEO objects at least once each night. They vary between a latitude of ± 40°, vary in altitude, and span across Earth. Table 7.1 shows the sensor name used in this work, their location (along with the continent/region they are located at for reference), and the altitude above sea level they are situated at.

Table 7.1. Sensor locations derived from Ackermann et al. (2018) [28]. The remaining sensor properties are presented above and are the same for all sensors.

Sensor Name	**Latitude [deg]**	**Longitude [deg]**	**Altitude [meters]**
Australia			
SNSR2	-33.112	138.52	1
SNSR3	-35.32	149.01	441.49
SNSR7	-36.168	144.97	98.958
SNSR8	-31.273	149.06	192.71
SNSR10	-30.931	136.78	140.63
SNSR15	-27	150	250
SNSR16	-30.313	149.55	151.04
SNSR17	-30.159	115.97	130.21
SNSR19	-21.82	114.17	1
SNSR23	-23.75	133.44	707.45
SNSR25	-24.75	117.7	430.85
Pacific			
SNSR1	19.29	166.61	1
SNSR4	20.583	-156.26	5.2083
SNSR6	21.349	-157.94	65
SNSR13	21.977	-159.75	7
North America			
SNSR5	38.522	-113.29	1618.4
SNSR9	39.111	-121.36	54
SNSR11	39.403	-118.73	1199
SNSR12	29.073	-110.96	205
SNSR18	38.549	-121.75	52.083
SNSR20	18.172	-66.592	1338

Continued on next page

Table 7.1 – continued from previous page

Sensor Name	Latitude [deg]	Longitude [deg]	Altitude [meters]
SNSR26	30.75	-115.22	1802.6
SNSR28	32.865	-114.39	135.42
South America			
SNSR21	-21.533	-64.733	3737.8
SNSR22	-31.51	-68.62	1618.4
SNSR24	-30.489	-68.62	2565.8
SNSR30	-29.015	-70.693	881.58
SNSR31	-23.019	-67.753	2486.8
Africa			
SNSR27	-22.561	17.066	1655
SNSR29	15.58	32.52	441.49
Europe			
SNSR14	37.175	-5.37	36.458
The Middle East			
SNSR32	26.236	50.039	1
SNSR33	29.859	30.252	140.63
SNSR34	27.505	33.982	301

To visually understand the locations of the 34 sensor system developed by Ackermann et al. (2018), presented in Table 7.1, a representation of these locations on the 2D Earth map is made [28]. This show presented in Fig. 7.1.

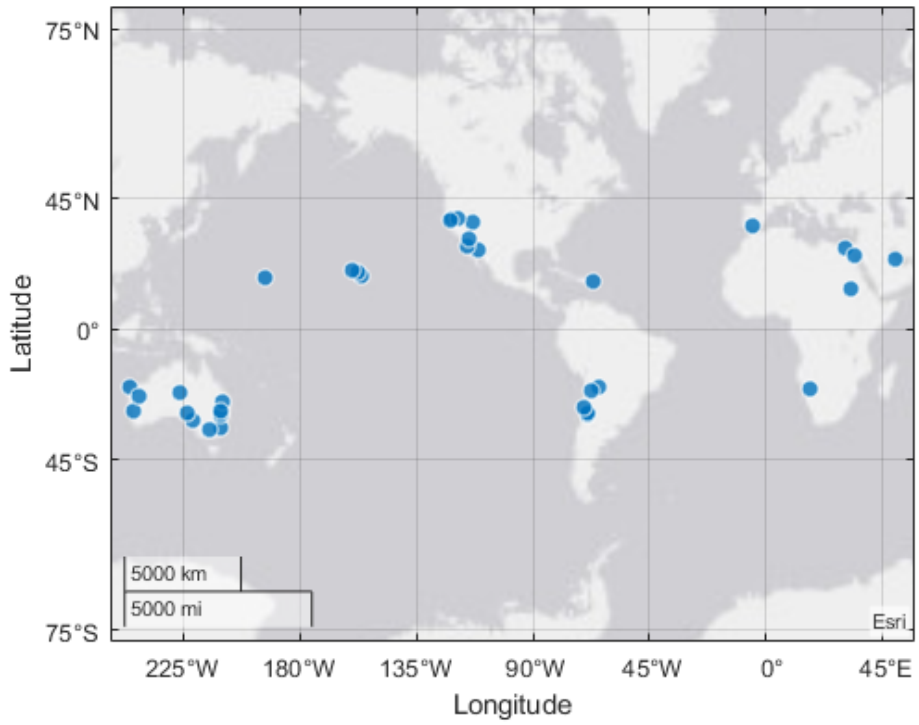


Figure 7.1. Multi-Sensor Location for 34 Sensor System, derived from Ackermann et al. (2018) [28]. The sensor locations, along with their altitudes are given in Table 7.1 and will be used throughout the results of this thesis.

Fig. 7.1 visually shows how the sensors are spread across the Earth. As stated above, these locations were developed to observe all GEO objects at least once per night [28]. The sensor locations, along with the RSO and sensor constants will be used in the remaining results, section 7.2 and section 7.3.

7.2 P_d Versus Elevation Constraint in the Sensor Tasking Problem

To fully test the effects of elevation and the probability of detection of RSOs, the algorithm described in Eqn. (3.15) is used on a single sensor example. A simple greedy algorithm is used, which will be both computationally efficient and will get close to maximizing the total number of objects seen [6]. This analysis is split into two parts: First, the single and multi-sensor tasking problem will be analyzed assuming

that the mean state of the object is the true state, or using Eqn. (3.16). Secondly, the same analysis is conducted and extended to include the uncertainty in the object's states using Eqn. (3.15).

7.2.1 Analysis Assuming The Mean State is the True State

Single Sensor Analysis

The sensor/sensors will try to maximize the number of GEO objects seen at least once in the following three scenarios:

1. Scenario 1: The RSOs will assume to have a $P_d = 1$ at all observation times
2. Scenario 2: The RSOs will have a variable P_d at all times, but the sensor tasking algorithm will not take this into account
3. Scenario 3: The RSOs will have a variable P_d at all times, and the sensor tasking algorithm will take this into account

First, the single sensor problem is studied. Each scenario in Fig. 7.2 - 7.4 was run at eight different latitude values for the single sensor's location and nine different minimum elevation constraints.

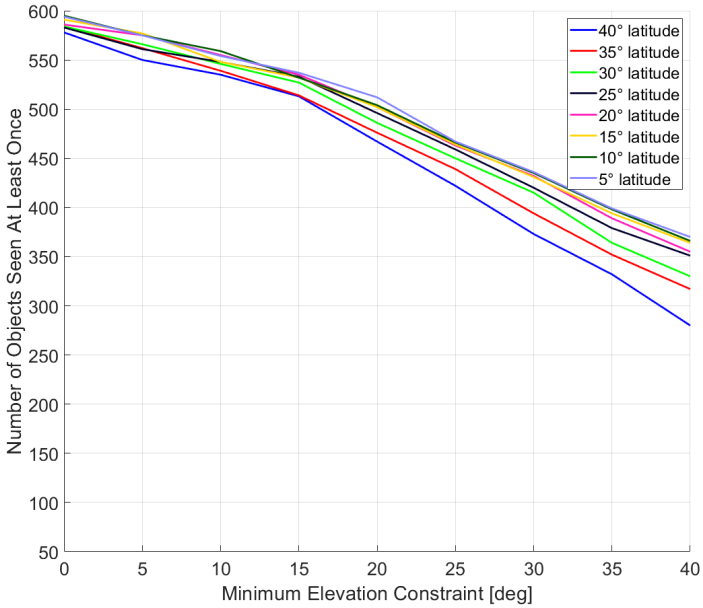


Figure 7.2. Scenario 1: The RSOs will assume to have a $P_d = 1$ at all observation times. This is the ideal case and does not represent the real world, but more of a "limit" on the total performance of the sensor.

Fig. 7.2 shows the first scenario: The RSOs are assumed to have a $P_d = 1$ at all observation times. As the elevation constraint for all latitudes tested decreases, the total number of GEO objects seen at least once increases, which is expected. Because $P_d = 1$, there is no limiting factor on the detectability of the objects as elevation decreases. The only focus is the total possible viewing directions. As the minimum elevation constraint decreases, the total number of possible viewing directions increases.

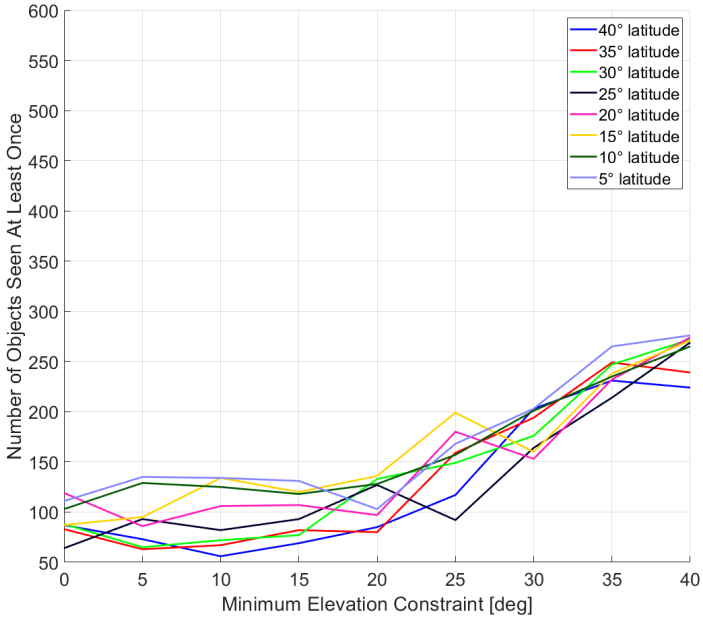


Figure 7.3. Scenario 2: The RSOs will have a variable P_d at all times, but the sensor tasking algorithm will not take this into account. This is the worst-case scenario, and performance decreases significantly as the minimum elevation constraint decreases.

The second scenario is shown in Fig. 7.3 and investigates the importance of taking into account the probability of detection in the sensor tasking optimization problem. Scenario 2 shows the results if the algorithm does not take into account the probability of detection of the objects when choosing an optimal viewing direction, but the total number of unique objects observed does take into account the probability of detection. For all latitudes tested, as the minimum elevation constraint decreases, the total number of unique RSOs also decreases. Because the algorithm is not taking into account the P_d of each object, once it has chosen all the viewing directions at higher elevations, it will choose the viewing directions at lower elevations, thinking that it is the optimal choice (because there are more objects now at those lower elevations with a $\mu(\tilde{X}_i) = 1$, which increases the weighting preference on that viewing direction). This implies, that a hard constraint on the minimum elevation (or a higher minimum elevation constraint) would improve the performance of the sensor's ability to see the

most amount of objects if the sensor does not take into account the probability of detection.

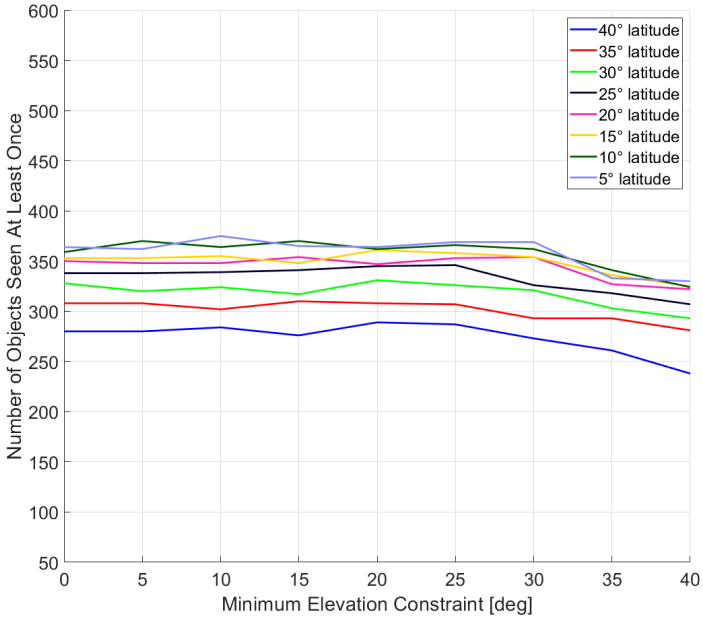


Figure 7.4. Scenario 3: The RSOs will have a variable P_d at all times, and the sensor tasking algorithm will take this into account. When the sensor is taking P_d into account, the performance increases as the minimum elevation constraint decreases, up to around 25° - 30° in this case.

When taking into account a variable P_d for each object at each observation time, the results shown in Fig. 7.4 are more resemblant of the conclusions made in chapter 4. For all latitude values tested, it appears that once the minimum elevation constraint decreases to around 25° - 30°, the total number of unique objects seen stays relatively constant. This is due to the decreasing P_d values shown in Fig. 4.5: for the entire solution space, once the elevation of the RSO decreased to a certain value (around 25° - 30° in this case), the probability of detection of that object becomes low. Because of the trend shown above, an elevation constraint is not useful when the sensor takes into account the probability of detection of the objects. The difference in performance at all latitudes and elevation constraints between scenarios two and three shows that

if the probability of detection of the objects is taken into account, the performance of the sensor increases without having to have a rigid minimum elevation constraint.

Multi-Sensor Analysis

To fully determine the effects of the probability of detection on the sensor tasking problem, an analysis of more than one sensor needs to be done. The following will investigate the probability of detection on the multi-sensor tasking problem. The analysis will study the multi-sensor system set up by Ackermann et al. (2018) and use the sensor locations and properties analyzed in their study, which is shown in Table 7.1 [28]. This section will be split up into the three scenarios described above, and the same new two-line element catalog of the night of March 19^{th}, 2019 is used for each scenario. More specifically, each of the three multi-sensor scenarios will be run twice: one set will have all the sensors have a constant elevation constraint of $25°$, and the other set will have no elevation constraint. The first plot in Fig. 7.5 will show the entire system, and how each sensor performs. For this first plot, there will be several characteristics that are used to analyze the multi-sensor system: at the top of each plot is a blue horizontal, dashed line. This line is the maximum number of RSOs to be observed (limiting the RSOs to GEO objects for this study). The largest yellow line is the sum of all the objects seen by each sensor at each time, and the red vertical line is the time from the start of the observation period at which all the objects have been seen at least once. Finally, the legend to the right of the plot describes the sensor location and the performance of each specific sensor.

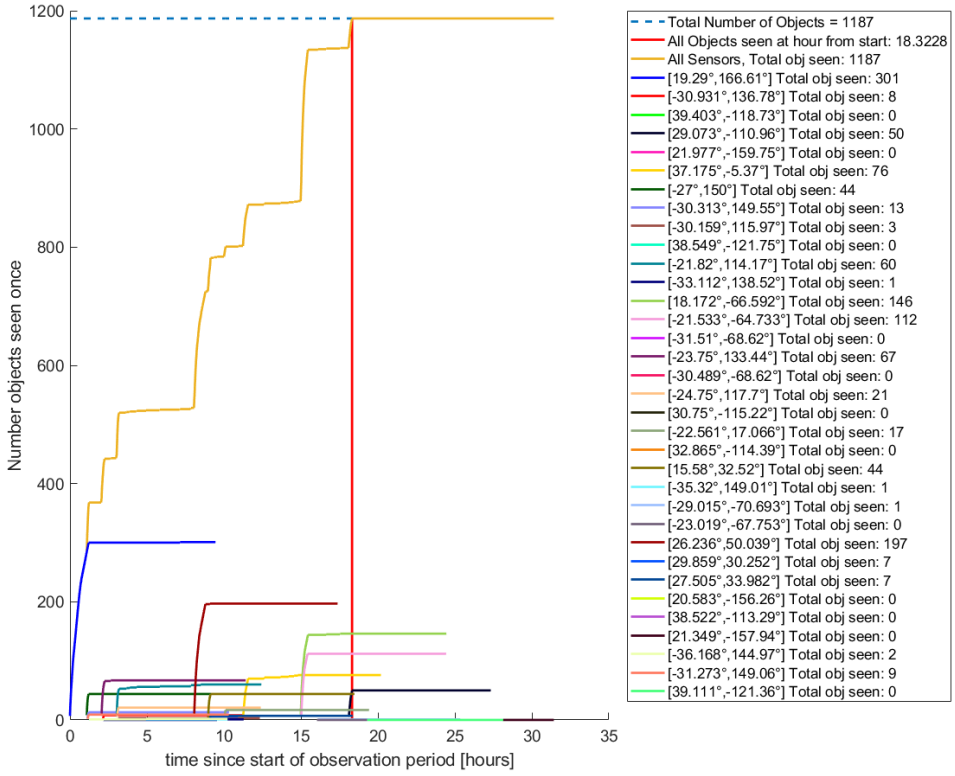


Figure 7.5. Scenario 1: Multi-Sensor tasking problem for the night of March 19th, 2019. P_d is assumed to be equal to one at all times for all RSOs in this scenario, and the time it takes for all the RSOs to be seen from the start of the night (red vertical line) is the shortest of all three scenarios.

Fig. 7.5 shows the ideal case: if every time an observation is taken, all the objects are detectable, no matter the correlation of the probability of detection with elevation. The plot shows that the sensors can view all GEO objects within 24 hours in this ideal case (about 18.3 hours from the start of the observation), as well as have 12 sensors not be utilized. Both the time at which all RSOs are observed and the total number of sensors not being utilized will be used as comparison tools for the next two scenarios. Though the results in Fig. 7.5 do not fully represent the real-world scenario, it is good to compare the results in scenarios two and three to the theoretical maximum. The remaining plots comparing the rigid and soft elevation constraints in

the different scenarios will only compare the total number of objects seen for each of the cases.

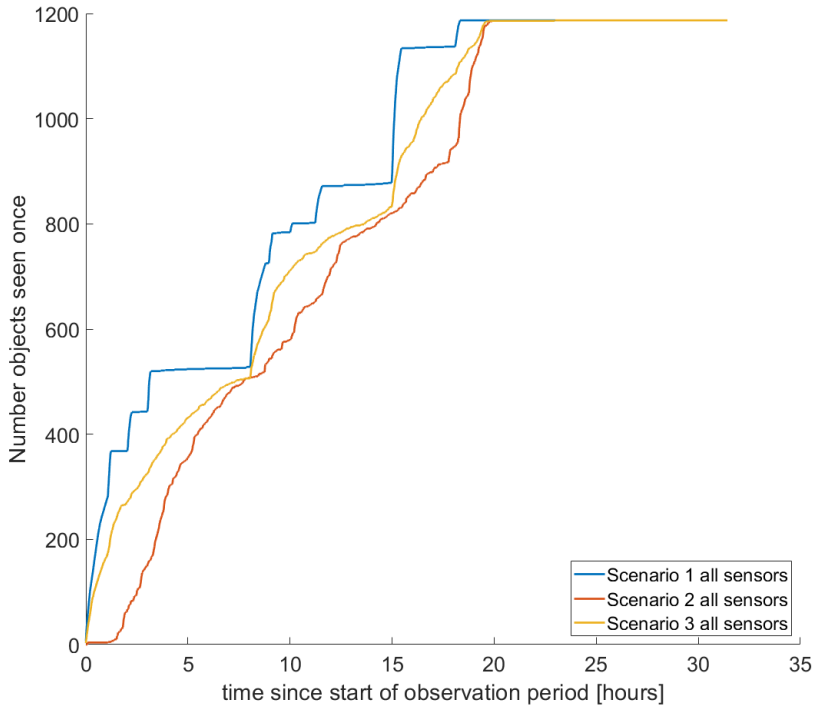


Figure 7.6. Comparison of Scenario 2 and 3 with a constant elevation constraint of 25° on each sensor. Because there is an elevation constraint in scenario 2, the total number of objects seen is not that different between each scenario. Scenario 1 is shown to show the maximum performance for this specific case.

Fig. 7.6 shows the comparison of all three scenarios, where each sensor has a constant elevation constraint. The main comparison is between scenario 2 and scenario 3. Since there is a strict elevation constraint of 25°, the performance difference between the multi-sensor tasking algorithm that does not consider the probability of detection (scenario 2) versus the multi-sensor tasking algorithm that does consider the probability of detection (scenario 3) is not that different, though scenario 3 does outperform scenario 2 before reaching the maximum number of RSOs to be viewed (1187 for this case). Even though P_d does not need to be modeled in the sensor tasking problem to have a reasonably optimal solution, when it is taken into account the

sensors saw the set of possible RSOs in their FOR faster, allowing the sensor operator to spend more time completing other tasks.

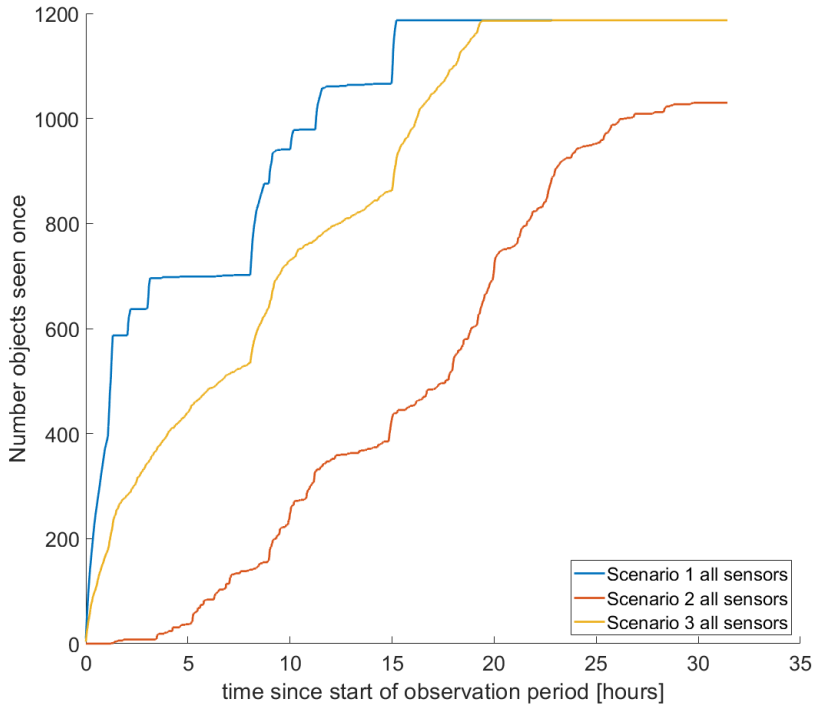


Figure 7.7. Comparison of Scenario 2 and 3 with no elevation constraint on each sensor. Because there is no elevation constraint in scenario 2, the total number of objects seen is significantly lower than in scenario 3. Scenario 1 is shown to show the maximum performance for this specific case.

Fig. 7.7 shows the same comparison as Fig. 7.6, but now each sensor has no rigid elevation constraint. Because of this, the difference between scenario 2 and scenario 3 is significant. Scenario 2 with no rigid elevation constraint is the only one out of all the tested scenarios to not reach the maximum number of RSOs to be observed (the total scenario 2 reaches is 1030 for this example). As described in the single sensor analysis for scenario 2, the lack of performance is due to the algorithm choosing viewing directions at lower elevations where the probability of detection of these objects is low. This causes times in the observation period for a sensor to see no new objects.

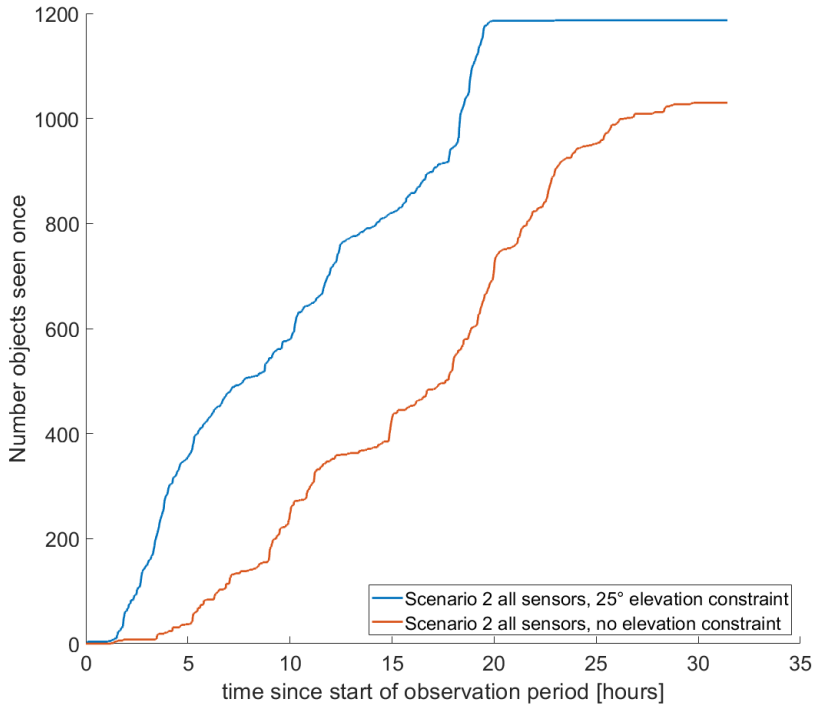


Figure 7.8. h_{min} Comparison for Scenario 2 in the Multi-Sensor System. When the P_d is not taken into account in the algorithm and there is no rigid elevation constraint, there is a significant loss in the multi-sensor tasking performance.

The results showed Fig. 7.8 are consistent with the conclusions made in the previous sections: when the probability of detection is not going to be considered in the sensor tasking algorithm, it is best to employ rigid elevation constraints on the sensors to achieve adequate results. The difference in performance between scenario 2 with and without an elevation constraint shown in Fig. 7.8 is too much to ignore. Without a rigid elevation constraint in a sensor tasking algorithm that does not account for the probability of detection, the performance diminishes greatly.

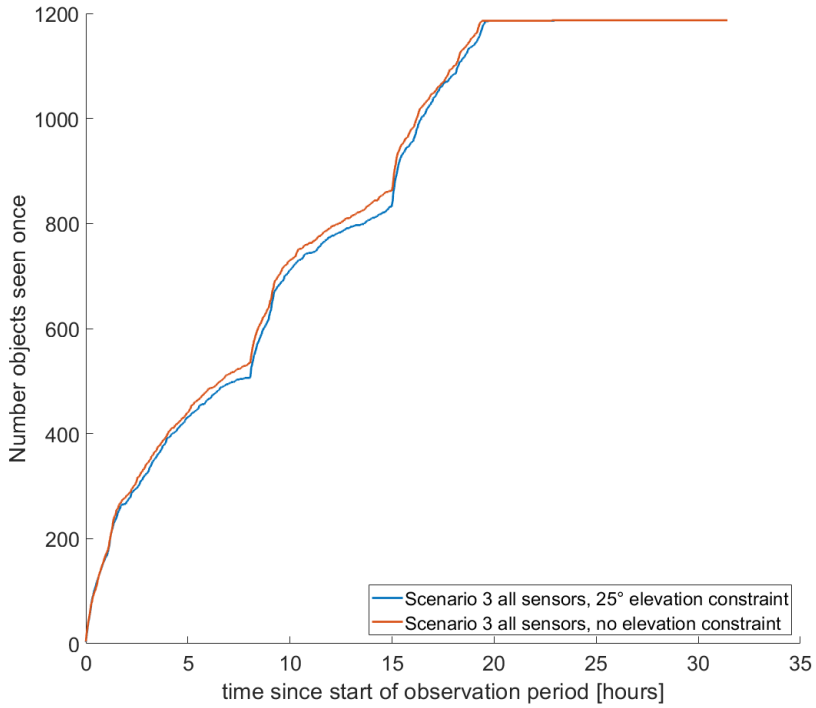


Figure 7.9. h_{min} Comparison for Scenario 3 in the Multi-Sensor System. When P_d is taken into account, there is no need to have a rigid elevation constraint on the sensors.

When analyzing Figure 7.9, the conclusions are consistent with the previous sections: when the probability of detection is taken into account into the sensor tasking algorithm, a rigid elevation constraint will not significantly change the performance, and in this case, actually decrease the performance, though very slightly. The only simulation difference between the red and blue lines of scenario 3 is that the blue line has a rigid elevation constraint, and the red line does not. These results make sense because the rigid elevation constraint of 25° is fairly strict. There is no significant computational difference between the multi-sensor system with the 25° elevation constraint on each sensor's FOR and the multi-sensor system with no elevation constraint. Overall, from above, the following conclusions can be drawn: if the probability of detection is considered and modeled in the single and multi-sensor tasking problem, then a heuristic elevation constraint is not necessary. However, if the probability of

detection is not considered, then a rigid elevation constraint of around 25° - 30° is recommended.

7.2.2 Analysis Considering Uncertainty in RSOs' States

The conclusions drawn from section 7.2.1 assume that the mean state of the object retrieved from the Two-Line Element catalog is the actual state of the object. Now, the same analysis as before is done, but now uncertainties in the RSOs' states are considered. This increases the complexity of the problem, but also represents the real-world system with greater detail.

Single Sensor Analysis

In this section, only scenario 2 and scenario 3 described in section 7.2.1 are shown, as scenario 1 was used as a proof-of-concept for the simulation. Fig. 7.10 and 7.11 shows the results of scenario 2 and scenario 3 when uncertainty in the RSOs' states are considered.

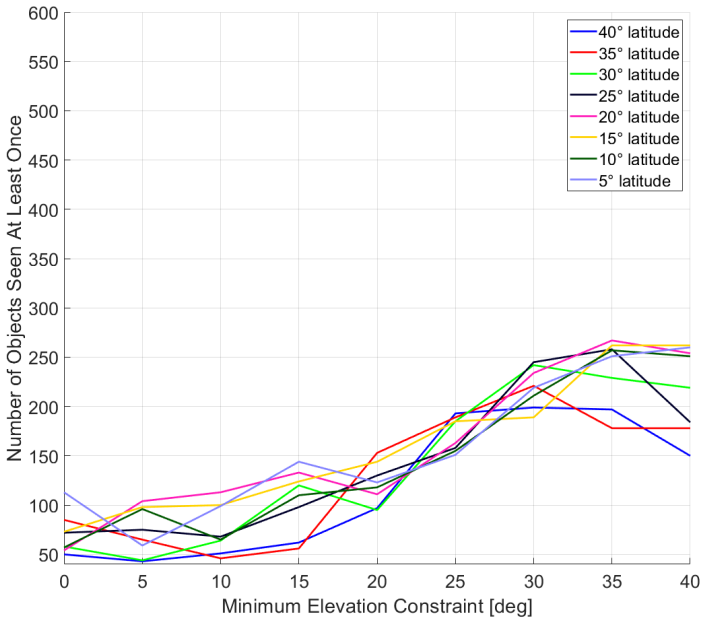


Figure 7.10. Scenario 2: The RSOs will have a variable P_d at all times, but the sensor tasking algorithm will not take this into account. Uncertainties in the RSOs' states are included in the analysis.

Fig. 7.10 shows a similar trend as in Fig. 7.3. When P_d is not considered, the performance of the sensor decreases as the elevation constraint decreases. However, in Fig. 7.10, there appears to be a maximum in performance for this specific sensor scenario when the rigid elevation constraint is set at around 25°-35° for all latitudes tested, with a majority of the latitudes tested having a maximum between 30°-35°. This may just be due to where the single sensor system is located and the specific TLE catalog used, but further testing has to be done, as well as how the uncertain states affect the multi-sensor tasking system.

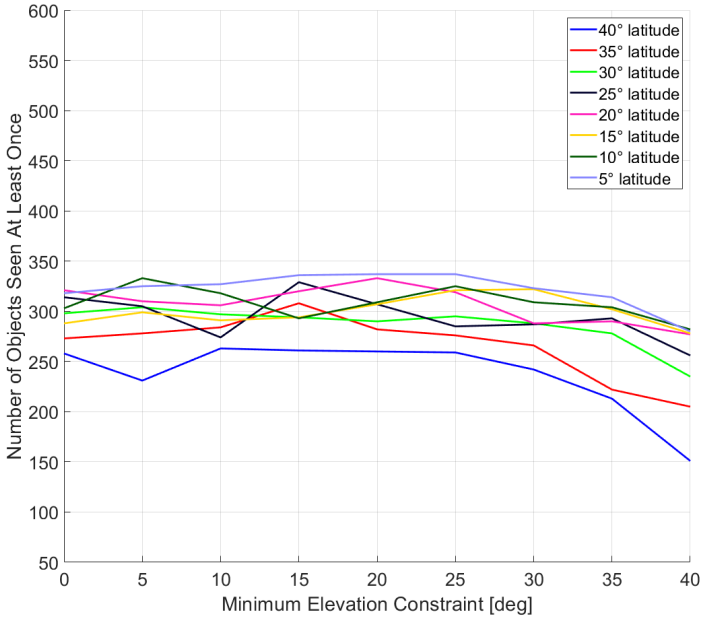


Figure 7.11. Scenario 3: The RSOs will have a variable P_d at all times, and the sensor tasking algorithm will take this into account. Uncertainties in the RSOs' states are included in the analysis.

Fig. 7.11 shows a similar trend as in Fig. 7.4. When P_d is considered, the performance of the single sensor increases as the rigid elevation constraint decreases until around an elevation constraint of 25°-30°, then performance increase is minimal. The results when adding uncertain states to the RSOs is consistent with the results when the mean state is assumed to be the true state for the single sensor system.

Multi-Sensor Analysis

Now, the multi-sensor system is analyzed similar to section 7.2.1, but now the calculation of the CDF at each observation is done. The same multi-sensor system developed by Ackermann et al. (2018) is implemented, and the same 1187 object set used in section 7.2.1 for the night of March 19$^{\text{th}}$, 2019 is studied. Scenario 2 and scenario 3 are analyzed and compared side-by-side to each other, shown in Fig. 7.12 and Fig. 7.13.

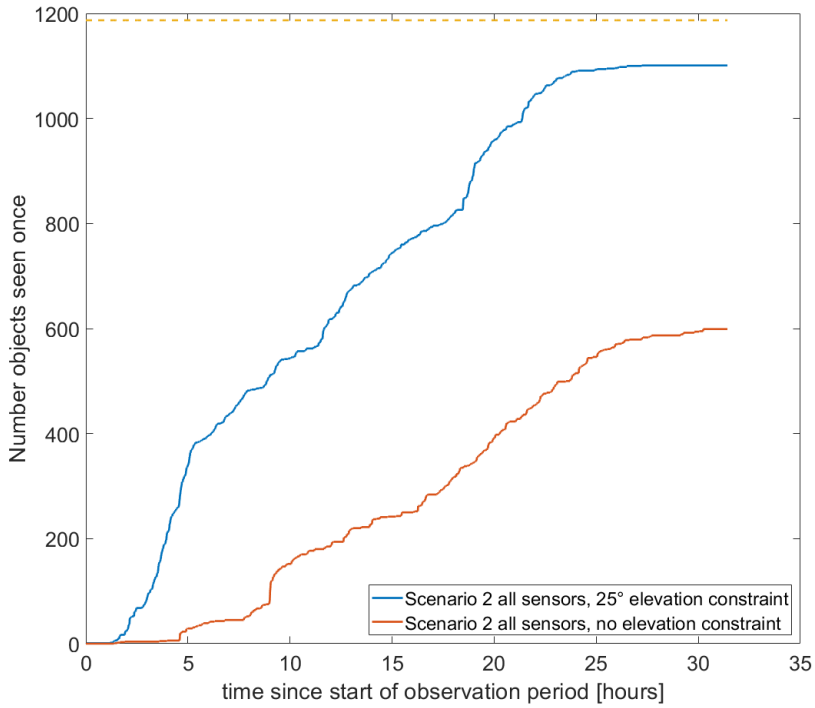


Figure 7.12. h_{min} Comparison for Scenario 2 in the Multi-Sensor System. Uncertainties in the RSOs' states are considered. There is a significant difference between the results due to P_d not being considered. The 34 sensor system does not observe all GEO objects when adding uncertainty in the RSOs' states. The dotted line is the total number of GEO objects to be observed (1187 in this example).

A few conclusions can be drawn from Fig. 7.12. The most relevant one is that when the uncertainty in the RSOs' states is considered, the multi-sensor system developed by Ackermann et al. (2018) does not observe all 1187 GEO objects for this night. This, however, is assuming P_d is not considered by the algorithm. For both cases ran for scenario 2, with and without a rigid elevation constraint, the multi-sensor system was not able to detect all GEO objects, with 1101 objects detected when all sensors have a 25°, and only 599 objects detected when no rigid elevation constraint is employed, which is almost half as less as when the uncertainty is not considered, shown in Fig. 7.8.

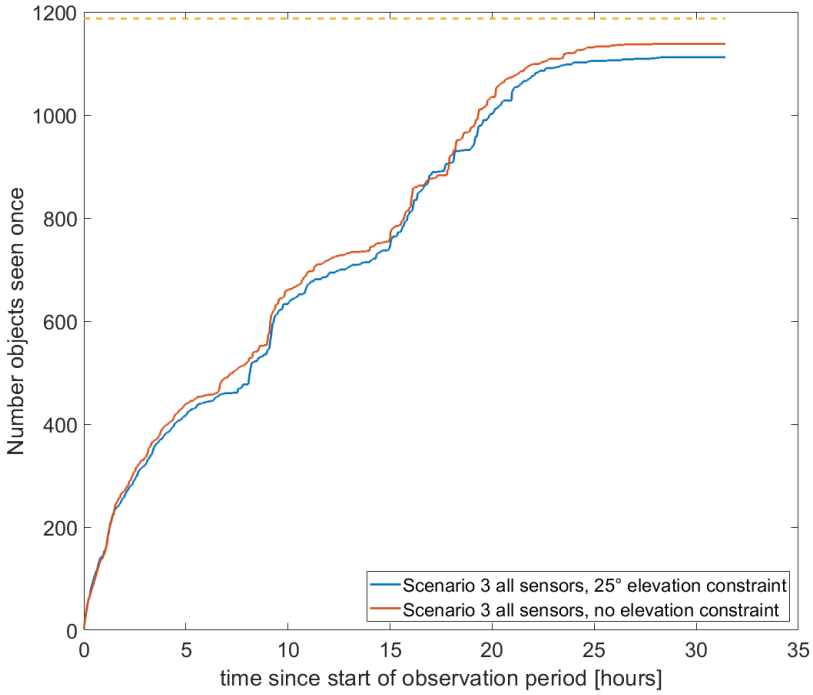


Figure 7.13. h_{min} Comparison for Scenario 3 in the Multi-Sensor System. When P_d is taken into account, there is no need to have a rigid elevation constraint on the sensors. The 34 sensor system does not observe all GEO objects when adding uncertainty in the RSOs' states. The dotted line is the total number of GEO objects to be observed (1187 in this example).

Fig. 7.13 also shows that the multi-sensor system does not observe all GEO objects. 1112 objects are observed for scenario 3 when there is a 25° rigid elevation constraint on the sensors, while 1138 objects are observed when there is no rigid elevation constraint. The performance of scenario 2 and scenario 3 when there is a 25° elevation constraint is very similar, with only eleven total objects observed differences. This is consistent with section 7.2.1 where the CDF was not calculated and simulated for the observation period. The difference between both of the scenario 3s is also consistent with that from section 7.2.1, and as expected, the performance of the sensors with no rigid elevation constraint is slightly better than the performance with a rigid elevation constraint, though small.

From the analysis shown above, when considering the uncertainty in the objects' states, the Ackermann et al. (2018) does not observe all cataloged GEO objects on the night of March 19$^{\text{th}}$, 2019. Therefore, complete coverage of the GEO regime is not reached. As well, if a sensor system does not consider the probability of detection when choosing viewing directions, then a rigid elevation constraint of 25°-35° is recommended and has been shown to produce results close to the system that does model the probability of detection. If the probability of detection is modeled, however, then no rigid elevation constraint is needed if there are no natural obstructions near the sensor.

7.3 Variable Repositioning Analysis

The following is a recap of the constant and variable repositioning cases described in chapter 5:

1. **Case 1**: When the repositioning time is constant and is not being compared to the variable repositioning time, then the sensor tasking performance is the true performance in the simulation, but it creates a model mismatch when applied to an actual sensor.

2. **Case 2**: When the constant repositioning solution is directly compared to the variable repositioning time solution, which is the most realistic simulation to the actual sensor, the optimizer solution is analyzed further:

    a. **A**: If the constant assumed for the repositioning time is smaller than the actual variable repositioning time for that m_g, the sensor does not view the objects at the time. The optimizer knows this and will account for these "missed" objects in future viewing direction considerations.

    b. **B**: If the constant assumed for the repositioning time is smaller than the actual variable repositioning time for that m_g, the sensor does not view the objects

at the time. The optimizer does not know this and assumes it detected the "missed" objects, even though they were never detected.

This section will now look at the single sensor tasking problem, but now comparing the constant repositioning solution with the more real-world variable repositioning model (case 2). The results in this section will be split up into two main focus areas: First, a comparison between several constant repositioning assumptions will be made with the true variable repositioning model. The performance will be compared for the two repositioning models of $\zeta = 1$ and $\zeta = 5$, where ζ is the constant multiplier from Eqn. (5.3) that determines how large of a factor the repositioning time is relative to the image processing time from Eqn. (5.2). Then, the sensor tasking problem will consider various weight models on the variable repositioning time. These weight models will be compared to the unweighted variable repositioning time for both $\zeta = 1$ and $\zeta = 5$ repositioning models.

7.3.1 Comparison of Constant and Variable Repositioning

In this section, the comparison of the assumed constant repositioning time to the real-world variable repositioning time will be made. For this section, three constant repositioning times will be analyzed: First, the "A to B" constant repositioning time will be modeled. This "A to B" repositioning time is calculated by finding the time it takes to reposition the sensor from any grid "A" to any other grid "B" using Eqn. (5.3). This is done for all viewing directions and is then averaged. This repositioning time will be less than the maximum possible repositioning time, which means that the system might not reach a specific viewing direction based on the assumed constant repositioning time. Therefore, Case 2A and Case 2B described in section 5.1 will be studied to analyze the problem. The other two constant repositioning times that will be studied assume that the constant repositioning time is greater than or equal to the maximum possible repositioning time, and therefore no viewing direction can be missed when repositioning. More specifically, two constants will be studied, the

constant repositioning time calculated be assuming that $\max(|\ \Delta\alpha\ |, |\ \Delta h\ |) = 180^\circ$ at all times, or the maximum time assuming no unwinding is modeled, and $\max(|\ \Delta\alpha\ |, |\ \Delta h\ |) = 360^\circ$ at all times, or assuming that this is the absolute maximum time with unwinding in the system. These two repositioning models will be referred to in this paper as "no unwind max" and "unwind max", described in Table 7.2.

Table 7.2. Maximum Constant Repositioning Time Models. The formulation is derived from Eqn. 5.3.

Constant Repositioning Time Model	Repositioning Time Calculations
no unwind max	$\max(\|\ \Delta\alpha\ \|, \|\ \Delta h\ \|) = 180^\circ$
unwind max	$\max(\|\ \Delta\alpha\ \|, \|\ \Delta h\ \|) = 360^\circ$

All these repositioning times will be studied for the test scenarios of $\zeta = 1$ and $\zeta = 5$, described in Eqn. 5.3 and chapter 5. The single sensor that is analyzed is SNSR1 from Table 7.1, which is located at a latitude of 19.29° and a longitude of 166.61°, for the night of March 19$^{\text{th}}$, 2019, and has the same characteristics described in section 7.1.2.

Sensor with Constant Repositioning Multiplier $\zeta = 1$

First, the system with $\zeta = 1$ is studied. For this system, the "A to B" constant repositioning time is calculated to be around 8.6 seconds. The constant repositioning times used for this section with Constant Repositioning Multiplier $\zeta = 1$ is shown in Table 7.3.

Table 7.3. Constant Repositioning Time Models for constant repositioning multiplier $\zeta = 1$. These times are calculated based on Eqn. (5.3).

Constant Repositioning Time Model	Repositioning Time [sec]
A to B	8.6
no unwind max	13.42
unwind max	18.97

The performance of the sensor tasking scenario assuming this constant is shown in Fig. 7.14.

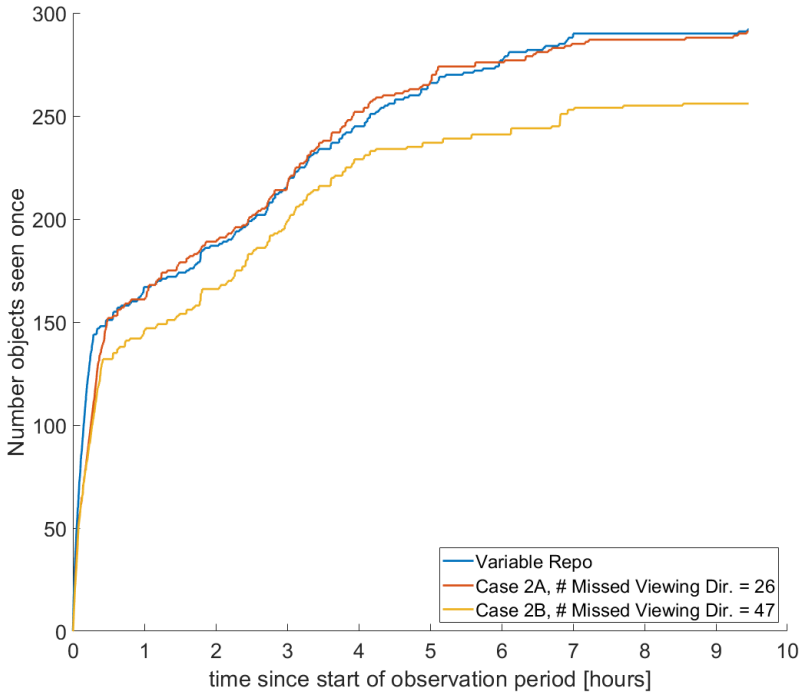


Figure 7.14. "A to B" Constant Repositioning Time, $\zeta = 1$. This Constant Repositioning time is equal to about 8.6 seconds for this sensor system.

In Fig. 7.14, the two cases are compared with the variable repositioning result. Here, Case 2A assumes that if the constant assumed for the repositioning time is smaller than the actual variable repositioning time for that observation period, the

sensor does not view the objects at the time. The optimizer knows this and will account for these "missed" objects in future viewing direction considerations. Case 2B assumes that if the constant assumed for the repositioning time is smaller than the actual variable repositioning time for that observation period, the sensor does not view the objects at the time. The optimizer does not know this and assumes it detected the "missed" objects, even though they were never detected. For this example, the results of Case 2A do not reach the viewing direction assuming the 8.6 second constant repositioning time a total of 26 times, while Case 2B does not reach the viewing direction a total of 47 times. This sounds significant, but for both of these cases, a total of 2704 observations were made for the given night.

When analyzing Fig. 7.14, two conclusions can be drawn. If the sensor knows when it misses the viewing direction based on its constant repositioning time and accounts for this (Case 2A), then the performance does not significantly change. However, if the variable repositioning time is not considered when the sensor misses a viewing direction (Case 2B), then performance decreases. This decrease is due to the sensor algorithm assuming that it has seen the objects in a viewing direction where the constant repositioning time assumed is less than the actual calculated repositioning time to reach that viewing direction. There is also the computational time to consider. The variable repositioning time model takes around 124 hours to run on the SID's server described in section 6.1, while the maximum amount of time it takes to run any of the constant repositioning models described in section 7.3 is the "A to B" model described above, with a time of around 22.5 hours. All other constant repositioning times are shorter, with the "A to B" model using $\zeta = 5$ taking around 8.8 hours. Therefore, from a computational standpoint, the variable repositioning model is not effective if the sensor uses a constant repositioning time and knows when it has missed a viewing direction.

The performance of the true, variable repositioning model is not significantly different than that of Case 2A. This is most likely due to the fact that only 26 out of the 2704 observation times were "missed" so the system had several more

opportunities to try to observe those "missed" objects again. Fig. 7.15 shows how the true variable repositioning model repositions throughout the night in the azimuth, elevation (α, h) frame.

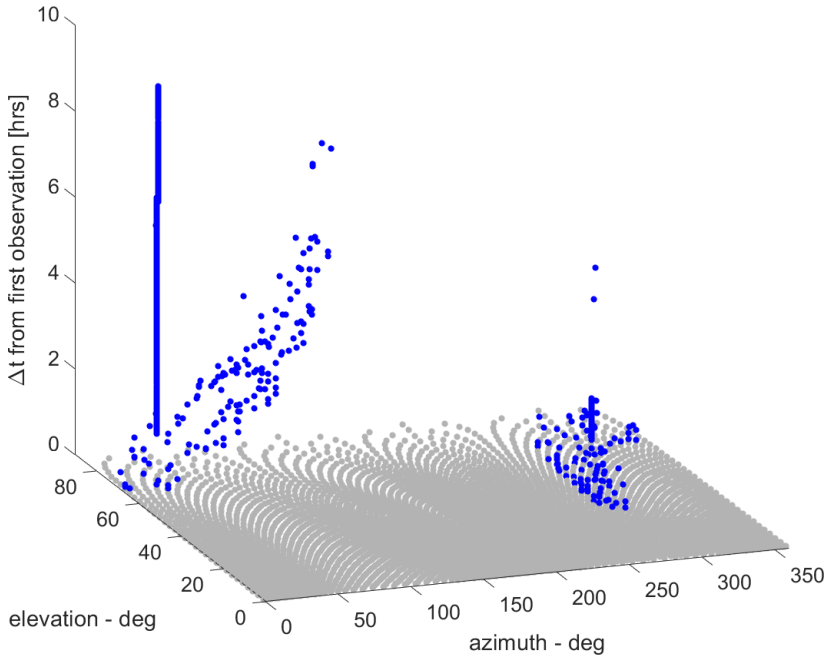


Figure 7.15. Variable repositioning viewing direction choice throughout the night in the azimuth, elevation (α, h) frame.

Fig. 7.15 shows the azimuth and elevation frame in the 2D projection XY plane, with the time that the observation was made in the Z-axis. This plot shows that as the night progresses, the observations move from the right azimuth of around 250°-350° to an azimuth of around 0°-100°. This shows that in the true variable repositioning time model, the system does not generally make large repositioning motions, so the general repositioning time is small. This small repositioning motion resembles the probability of detection throughout the night described in Fig. 4.5 in chapter 4, where the initial large probability of detection of these GEO objects started around 250°-350° azimuth and then moved to the right to the 0°-100° azimuth range at the

end of the night. The other conclusion as to why the repositioning time effect is not significant in Fig. 7.14 is the "pillars" shown in Fig. 7.15. These pillars are due to the initial CDF value of a GEO object being largely in this viewing direction, but the actual object is in a viewing direction where its initial CDF is small. The algorithm does not know this, but only the CDF probability of where the object might be. This shows an area of improvement for the sensor tasking algorithm, but also that the system has so many opportunities to make observations, that it can spend the time looking at the same viewing direction multiple times. This is one of the reasons why the "A to B" system for Case 2A performs the same relatively the same as the variable repositioning time model, as the repositioning model of $\zeta = 1$ is not a significant amount of time spent repositioning, so there are plenty of opportunities for the assumed constant repositioning time model to try to observed objects that were "missed."

Now, the assumed constant repositioning time for the "no unwind max" and the "unwind max" models are analyzed. For $\zeta = 1$, the times for these models are around 13.42 seconds, and 18.97 seconds respectively. The variable repositioning model assumes that unwinding is not a factor in the sensor's repositioning, so the maximum amount of time for the variable repositioning possible is equal to the "no unwind max" constant repositioning time. The comparison of these three models is shown in Fig. 7.16.

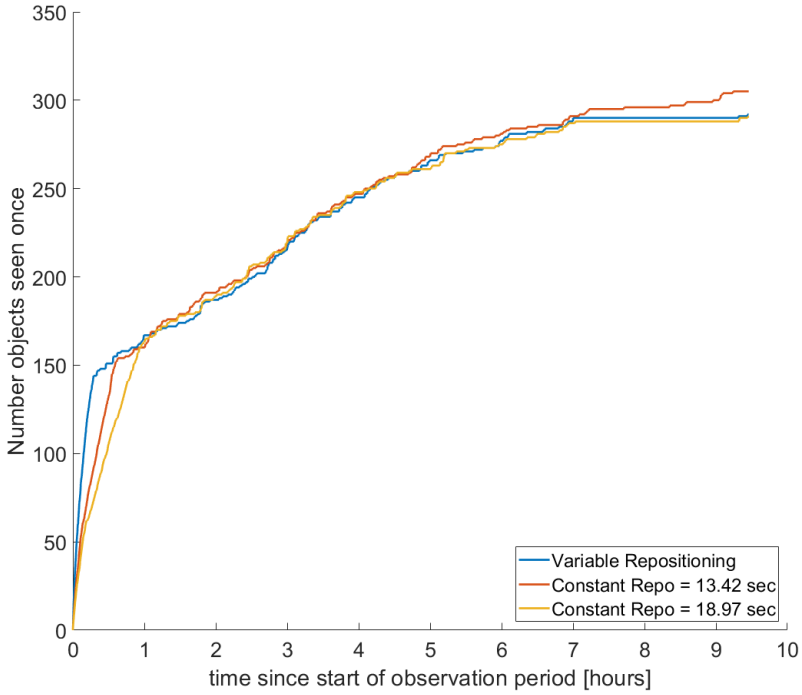


Figure 7.16. Comparison of the maximum and the unwinding maximum repositioning time (calculated from Eqn. (5.3)) for $\zeta = 1$. It is assumed that the maximum slew distance is 180°, and the unwinding maximum is a slew distance of 360°.

Fig. 7.16 shows that both of the constant repositioning results either perform the same or better than the variable repositioning model in this specific case. Again, this is most likely due to the fact that the "no unwind max" model (red) has a total number of observations of 1956 for the night, and the "unwind max" (yellow) has a total of 1483 observations.

For $\zeta = 1$, there are a few conclusions that can be drawn. First, for a constant repositioning time that is less than the maximum repositioning time, if the algorithm does not consider when its constant repositioning is less than the actual repositioning for an observation, then there is a decrease in performance. If the algorithm does account for this, then the performance is similar to that of the true, variable repositioning time. If, however, the constant repositioning time is greater than the maximum calculated repositioning time, then the sensor performs around the same as

the true, variable repositioning time scenario for this case and for $\zeta = 1$. Finally, the computational time to complete the true variable repositioning time is significantly higher than all other constant repositioning model simulations.

Sensor with Constant Repositioning Multiplier $\zeta = 5$

Now, the same analysis will be conducted, but with $\zeta = 5$ from Eqn. (5.3). The constant repositioning times used for this section with Constant Repositioning Multiplier $\zeta = 5$ is shown in Table 7.4.

Table 7.4. Constant Repositioning Time Models for constant repositioning multiplier $\zeta = 5$. These times are calculated based on Eqn. (5.3).

Constant Repositioning Time Model	Repositioning Time [sec]
A to B	42.98
no unwind max	67.08
unwind max	94.87

In this scenario, the "A to B" constant repositioning time is calculated to be around 42.98 seconds, with the maximum calculated repositioning time being around 67.08 seconds. This allows for the comparison of a scenario where t_{repos} is significantly larger than t_{img} from Eqn. (5.2), where $t_{img} = 4$ seconds. The $\zeta = 5$ model is also the model that closely models the POGS sensor.

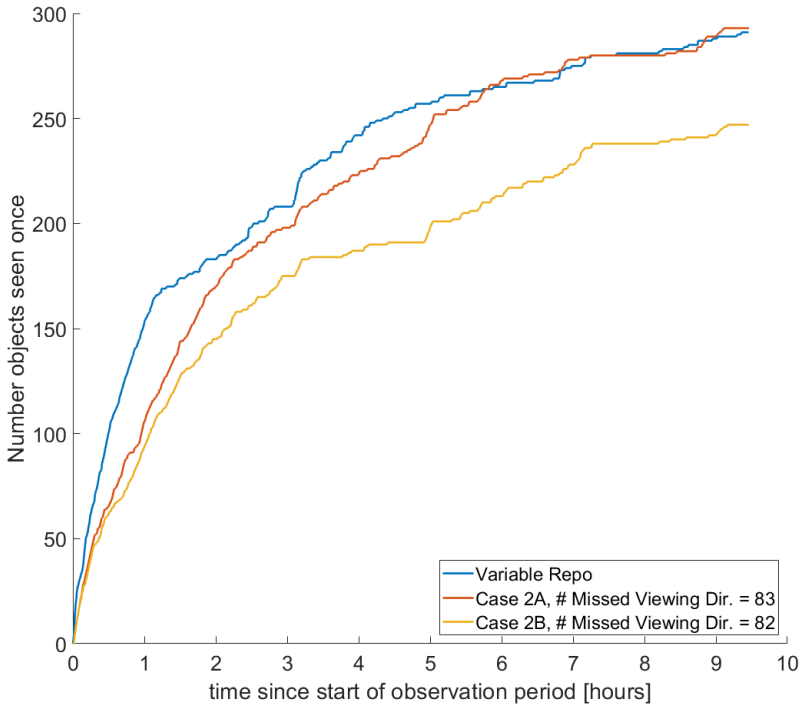


Figure 7.17. "A to B" Constant Repositioning Time, $\zeta = 5$. This Constant Repositioning time is equal to about 42.98 seconds, for this sensor system.

Fig. 7.17 shows the results when Case 2A and Case 2B apply to an assumed constant repositioning time for $\zeta = 5$. For Case 2A and Case 2B, the total number of observations for the night was 726. Again, like in Fig. 7.14 we see the same results: if the sensor knows that it "missed" a viewing direction and takes it into account, then the performance is close to the performance of the actual, variable repositioning model. However, if the algorithm does not take the missed viewing directions into account, then the performance of the sensor decreases. In Case 2A, the total number of viewing directions missed is a larger percentage of the total of observations made than for the repositioning model with $\zeta = 1$, but the system still has plenty of other opportunities to observe the objects it did miss.

Now, the "no unwind max" and the "unwind max" models will be compared with the true, variable repositioning model. The assumed repositioning time for these

models is around 67.08 seconds and 94.87 seconds respectively. The results are shown in Fig. 7.18.

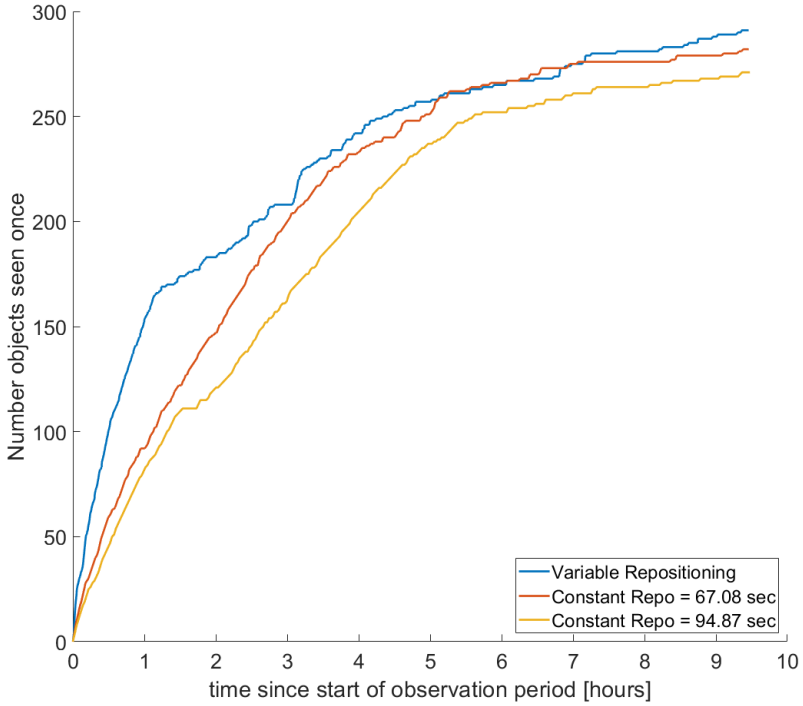


Figure 7.18. Comparison of the maximum and the unwinding maximum repositioning time (calculated from Eqn. (5.3)) for $\zeta = 5$. It is assumed that the maximum slew distance is 180°, and twice the unwinding maximum is a slew distance of 360°.

Similar to Fig. 7.16, Fig. 7.18 shows that the performance of the assumed constant repositioning models are similar to the true, variable repositioning model. The total number of observations made for the constant repositioning models were 480 and 346 for the "no unwind max" and the "unwind max" models respectively. However, in Fig. 7.18, both the "no unwind max" and the "unwind max" models perform worse than the variable repositioning model, which was not observed when $\zeta = 1$. When $\zeta = 5$, the repositioning time is generally greater and therefore is a larger factor in the sensor tasking problem. Therefore, it is expected that if the repositioning time becomes a greater factor, then the variable repositioning model will generally produce

a more optimal result than the assumed repositioning times that are greater than the maximum calculated repositioning times.

The conclusions for the constant repositioning multiplier $\zeta = 5$ are similar to the conclusions made for the constant repositioning multiplier $\zeta = 1$. If the constant repositioning time is less than the maximum repositioning time, then like for $\zeta = 1$, if the algorithm does not consider when its constant repositioning is less than the actual repositioning for an observation, then there is a decrease in performance. If the algorithm does account for this, then the performance is similar to that of the true, variable repositioning time. For an assumed constant repositioning time that is greater than the maximum, there is a small difference in solutions between $\zeta = 1$ and $\zeta = 5$. If $\zeta = 5$, then the assumed constant repositioning times that are larger than the maximum calculated repositioning times perform worse than if $\zeta = 1$.

7.3.2 Optimizing with Repositioning Time Considerations

Now, the optimizers described in section 5.2.2 and Eqn. (5.4) using the single sensor system will be analyzed to study the benefits and drawbacks of considering the variable repositioning time in the sensor tasking framework. As in section 7.3.1, the repositioning calculations using $\zeta = 1$ and $\zeta = 5$ will be tested. From here, four weighted repositioning time optimizers will be analyzed and compared to the unweighted variable repositioning time. These optimizers use the constant value described in Eqn. (5.5) that change the amount of weight given to the repositioning time, and are described as "C". The constant Cs that are used are C = 1, C = 2.5, C = 5. The linear weighted repositioning model described in Eqn. (5.7) is also analyzed alongside the C values.

Sensor with Constant Repositioning Multiplier $\zeta = 1$

First, the variable repositioning calculation with $\zeta = 1$ will be used to analyze the weighted and unweighted solutions.

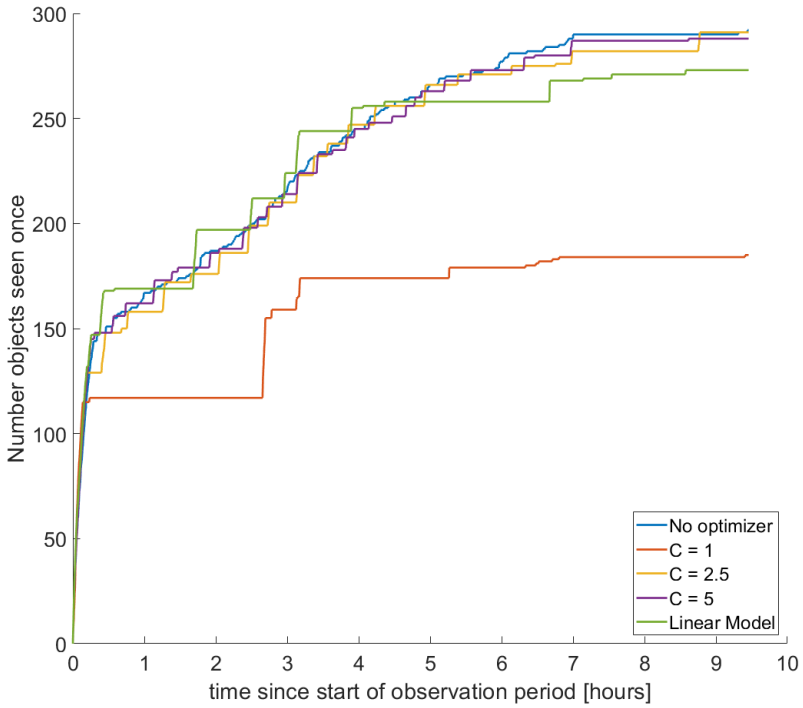


Figure 7.19. Comparison of Repositioning Weight Functions for $\zeta = 1$. For this example, the weight models described in chapter 5 did not perform any better than the unweighted solution.

Fig. 7.19 shows the comparison of the unweighted variable repositioning solution (blue), with the weighted solutions described by Eqn. (5.5) and Eqn. (5.7). For the unweighted solution, a total of 7931 observations, while C = 1, C = 2.5, C = 5, and the linear model had a total of 8448, 8289, 8237, and 8354 observations for the night respectively. These values are also displayed in Table 7.5.

Table 7.5. Total number of observations for weighted repositioning models for constant repositioning multiplier $\zeta = 1$.

Weighted Repositioning Model	Total Number of Observations
Unweighted Model	7931
C = 1	8448
C = 2.5	8289
C = 5	8237
Linear Model	8354

Because all the weighted solutions put a penalty on the time for the sensor to reposition, the sensor tasking algorithm will favor viewing directions that are closer together, and therefore will have a higher amount of observations throughout the entire night than the unweighted variable repositioning solution.

There are a few conclusions that can be drawn from Fig. 7.19. First, if the weight of the repositioning time is too strict, i.e. C = 1, then the solution is sub-optimal compared to the unweighted solution. When C = 1, if the sensor tasking algorithm finds a highly weighted grid, but it would take too long for the sensor to reposition, this time would bring down the weight of that grid significantly, and therefore will ignore potentially more optimal grids then the grid the algorithm actually chooses. This is an initial indication that the repositioning time may not be a significant variable to model, at least for scenarios where the repositioning time is small ($\zeta = 1$). The other conclusion is that the remaining weighted optimizer generally performs around the same as the unweighted solution. As described above in section 7.3.1 and in Fig. 7.15, because there are so many opportunities for the sensor to make observations in all of these cases, there is little benefit to considering the amount of time saved by choosing a viewing direction that is close to the previous viewing direction.

Sensor with Constant Repositioning Multiplier $\zeta = 5$

Now, the sensor with the repositioning calculation of $\zeta = 5$ is analyzed. This model increases the overall repositioning time between viewing directions, so it is expected that there will be some differences between these weighted models and the weighted models described above for $\zeta = 1$.

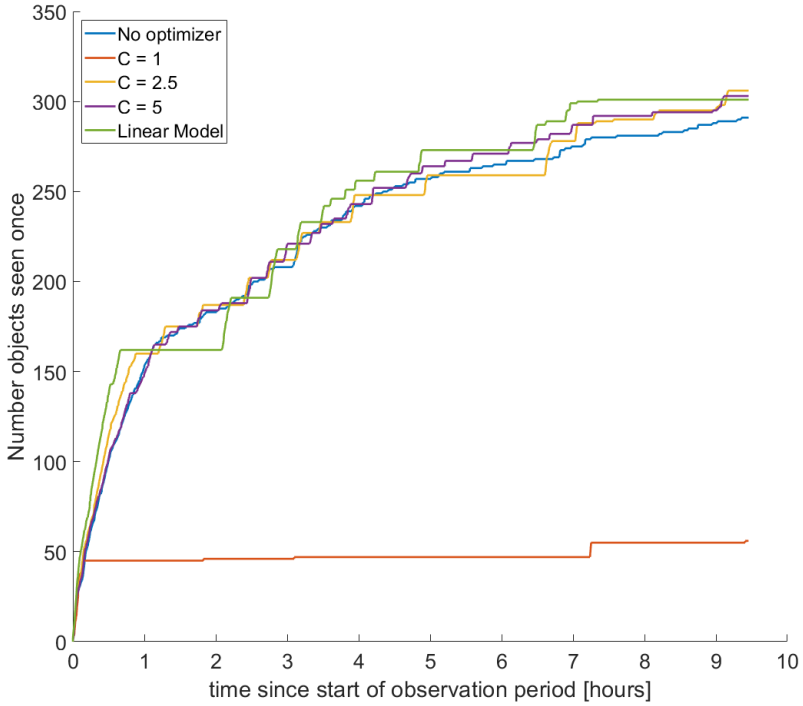


Figure 7.20. Comparison of Repositioning Weight Functions for $\zeta = 5$. For this example, the weight models described in chapter 5 do have performance increase above the unweighted solution, besides the constant wieght value C = 1 model described in Eqn. (5.5).

Because the repositioning time is greater in Fig. 7.20, it is expected that the total number of viewing directions should decrease from the $\zeta = 1$ case. For the unweighted solution, a total of 6220 observations, while C = 1, C = 2.5, C = 5, and the linear model had a total of 8444, 7239, 6866, and 7628 observations for the night respectively. The total number of observations is also displayed in Table 7.6

Table 7.6. Total number of observations for weighted repositioning models for constant repositioning multiplier $\zeta = 5$.

Weighted Repositioning Model	Total Number of Observations
Unweighted Model	6220
C = 1	8444
C = 2.5	7239
C = 5	6866
Linear Model	7628

The total number of observations is indeed less, but not significantly, and hence why in Fig. 7.20 the optimizers do not perform significantly better than the unweighted solution, similar to Fig. 7.19. C = 1 is also significantly less than all other optimizers. Similar to Fig. 7.19, this is due to the weight on the repositioning time being too heavy. In fact, in the model where the repositioning time from one viewing direction to another is larger ($\zeta = 5$), C = 1 performs worse than the same weight model in the scenario where the repositioning time is smaller from one viewing direction to another ($\zeta = 1$). In Fig. 7.20 however, all the weighted optimizers besides C = 1 do perform better than the unweighted solution. This suggests that the model can be optimized when repositioning time is considered and if the repositioning time is large enough to play an important factor. However, the weighted models that do outperform the unweighted model do not outperform significantly. This suggests that, as stated in the previous sections, the variable repositioning factor is not too significant of a factor to consider in the sensor tasking formulation. This will have to be studied further for sensors where the t_{img} is a larger factor in the observation process, as described in Eqn. (5.2). In this analysis, t_{img} was small and therefore allowed for a large number of observations to be made. If t_{img} was larger, fewer observations would have been made, and therefore the time saved from choosing viewing directions that result in a low repositioning time would have been a greater factor in the sensor tasking problem.

8. SUMMARY

This thesis analyzed the sensor tasking optimization for a single electro-optical sensor and a multi electro-optical sensor network for complete coverage of the geosynchronous region. The goal of this thesis is to understand the optical and physical assumptions on a ground-based sensor, and how these assumptions affect the single and multi-sensor tasking problem. This goal is achieved by answering the three main questions described in section 1.1. First, what is the effect of the probability of detection on the sensor tasking, especially when compared to rigid elevation constraints? Secondly, can the framework from Frueh et al. (2018) be expanded towards complete coverage of the GEO regime [6, 24]? Are the assumptions of Ackermann et al. (2018) correct [28]? Finally, how much of a benefit is there to modeling the computationally expensive variable repositioning time in the sensor tasking framework to achieve more accurate simulation models? If incorporated, can an optimizing strategy be developed to improve the performance of the sensor tasking problem?

8.1 Conclusions

The first question is answered in several parts. First, the comparison of a varying minimum elevation constraint on a sensor, considerations of the probability of detection of resident space objects, and the effects of the probability of detection on the multi-sensor tasking problem are shown. For constant irradiation, as elevation decreases, so does the probability of detection. It has been shown in chapter 4, that when considering the effects of a changing geometry on the overall irradiation as well, the probability of detection becomes more complex, and the probability of detection decreases significantly past an elevation of 20°. The single sensor tasking problem is then analyzed with the probability of detection and rigid elevation constraints in

mind, shown in chapter 7. As the minimum elevation constraint increases, the total number of possible viewing directions decreases. When comparing the probability of detection to a sensor with varying minimum elevation constraints, it was shown that as the minimum elevation constraint decreased to around 25° - 35°, the sensor's performance remained constant, and decreasing the minimum elevation constraint further did not improve the performance of the sensor. For the scenario where the probability of detection was not calculated, the maximum performance of the single sensor was achieved with an elevation constraint between 25° - 35°, and beyond that, the performance decreased significantly. The multi-sensor system developed by Ackermann et al. (2018) was studied to analyze elevation constraints [28]. The multi-sensor setup was analyzed with and without a rigid elevation constraint, and with and without the probability of detection considered in the sensor tasking algorithm. These scenarios were then run using the two-line element catalog for the night of March 19^{th}, 2019, to confirm the results presented in the single sensor study. When there was no elevation constraint but the probability of detection was incorporated in the sensor tasking formula, the system performed slightly better than the same scenario when there was a rigid elevation constraint. When the probability of detection was not considered in the sensor tasking problem, a rigid elevation constraint of 25° produced results similar to the system that had the probability of detection considered. However, if the system does not consider the probability of detection and does not have an elevation constraint, then the multi-sensor system performs significantly worse. From this analysis, a rigid elevation constraint between 25° - 35° is useful when not taking the probability of detection into account, as the true probability of detection is elevation-dependent. However, this is also neglecting further illumination effects that lead to a varying probability of detection. When accounting for the probability of detection correctly, using the formulation of the optimizer shown in this paper, an elevation constraint is obsolete and overall a larger number of objects are detected in the investigated scenario [24].

The second question builds on the first, but now incorporates the multi-sensor framework developed by suggested in Ackermann et al. (2018) [28]. When the multi-sensor system only includes the probability of detection in the analysis, the entire system can achieve full GEO coverage. Then, the uncertainty of the object's states was then considered in the multi-sensor tasking framework. When the uncertainties were included, the multi-sensor tasking system could not achieve full coverage for scenario two or scenario three. Therefore, with the assumptions presented by Ackermann et al. (2018), the multi-sensor tasking system cannot achieve full coverage of GEO objects.

The final question analyzes the single sensor framework with a constant and variable repositioning time in chapter 7. First, the variable repositioning time model is compared to two constant repositioning systems with ζ from Eqn. (5.3) being equal to one and five to test the dependence on the significance of the repositioning time. It was shown that for $\zeta = 1$ and $\zeta = 5$ if the constant repositioning time is less than the maximum repositioning time, there could be a significant decrease in performance if the repositioning time is not considered in the optimizer. For both $\zeta = 1$ and $\zeta = 5$, it was also shown that there is not a significant change in the solution if the constant repositioning time is greater than the maximum possible. Then, the variable repositioning time is analyzed with and without weighted considerations on the repositioning time, as described in chapter 5 and chapter 7. These results are analyzed again for $\zeta = 1$ and $\zeta = 5$. For $\zeta = 1$, it was shown in this specific scenario, the weighted repositioning solutions to the single sensor tasking solution did not produce results that performed better than the unweighted repositioning time. However, for $\zeta = 5$, there was an increase in performance when a weighted model for repositioning time was used versus the unweighted solution for this specific test case, though the performance increase was not significant. For both ζ values, it was also shown that the weighted models that performed the best did not weigh the repositioning time heavily, suggesting that for this example, the repositioning time is not a significant

factor in the sensor tasking problem if the image processing step of the observation is small.

8.2 Recommendations and Future Work

From the work and conclusions presented above, several recommendations to sensor operators and modelers can be made. First, a rigid elevation constraint of between 25° - 35° is recommended to an electro-optical sensor operator using a CCD if the probability of detection of GEO objects is not modeled. This will allow for a fairly optimal sensor tasking solution to be produced without the computational time of modeling the probability of detection. If the sensor operator is modeling the probability of detection, then there is no need to have a rigid elevation constraint.

Because the multi-sensor framework developed by Ackermann et al. (2018) did not achieve full coverage, this is a point for further research before full recommendations can be made. Some research includes testing various multi-sensor setups, modifying the greedy algorithm to better account for object state uncertainty, as well as including weather effects at the sensors' location to achieve full coverage of GEO objects. One of the main reasons why this sensor setup did not achieve full coverage was due to the initial covariance assumed for the objects [64]. Studying various initial covariance to find what assumed initial covariance achieves full coverage with the multi-sensor setup assumed for this work is some future work that could also be analyzed [28].

As for the results of the variable repositioning time, a few recommendations can be made for electro-optical sensor operator using a CCD. If the sensor has a short image processing time and a fast repositioning time, then it is recommended that the sensor operator ignores the variable repositioning time and assumes a constant repositioning that is greater than the maximum repositioning time, to avoid any errors. If the repositioning time is a large component, however, then there will be a decrease in performance if the variable repositioning is not considered, though it is not

significant. The computational time for calculating the variable repositioning time is much greater than the performance increase in the sensor tasking problem. These recommendations are limited due to the complexity of the problem, so more research needs to be completed. The repositioning model used in this work developed based on the Purdue Optical Ground station. Other repositioning models could be further studied to extend the results in this research. The sensor model in this work assumed a small amount of time for the sensor to take the images and then reposition to the next viewing direction. A study to analyze the effects of increasing this image processing time versus the variable repositioning time will be beneficial. This reposition model also assumed there was no unwinding in the sensor, and that the sensor slewed at the same rate in all directions, which does not have to be the case. In this thesis, only the single sensor tasking problem was studied with a variable repositioning time due to high computational times. If the work could be parallelized more, or if a more efficient computer was used, then perhaps the variable repositioning model could be extended to the multi-sensor tasking problem.

REFERENCES

[1] David A. Vallado. *Fundamentals of Astrodynamics and Applications*, 3rd ed., Microcosm Press/Springer, 2007.

[2] Richard H. Battin. *An Introduction to the Mathematics and Methods of Astrodynamics, Revised Edition* American Institute of Aeronautics and Astronauctics, Inc., 1999

[3] Victor G. Szebehely. *Adventures in Celestial Mechanics*. University of Texas Press, Austin, Tx, 1989

[4] O. Montenbruck, P. Steigenberger, and U. Hugentobler. Enhanced Solar Radiation Pressure Modeling for Galileo Satellites. *Journal of Geodesy*, 89(3):283-297, 2015

[5] Roshan Thomas Eapen. *Averaged Solar Radiation Pressure Modeling for High Area-to-Mass Ratio Objects in Geostationary Space*. Masters, Purdue University, 2017

[6] Bryan Little. *Optical Sensor Tasking Optimization For Space Situational Awareness*, Ph.D, Purdue University, 2019.

[7] Carolin Frueh. *Chapter 3. Available Space Object Catalogs and Two Line Elements (TLE)*, AAE 590: Space Traffic Management, Lecture Script, Purdue, School of Aeronautics and Astronautics, 2019

[8] Carolin Frueh. *Chapter 4. Observations*, AAE 590: Space Traffic Management, Lecture Script, Purdue, School of Aeronautics and Astronautics, 2019

[9] Carolin Frueh. *Chapter 5. Coordinate Systems and Time*, AAE 590: Space Traffic Management, Lecture Script, Purdue, School of Aeronautics and Astronautics, 2019

[10] Carolin Frueh. *Chapter 8. Orbit Propagation and Perturbations in the Near Earth Space*, AAE 590: Space Traffic Management, Lecture Script, Purdue, School of Aeronautics and Astronautics, 2019

[11] Carolin Frueh. *Chapter 9. Orbit Improvement/Filtering: Minimum Mean Square Error Estimation*, AAE 590: Space Traffic Management, Lecture Script, Purdue, School of Aeronautics and Astronautics, 2019

[12] Oliver Montenbruck and Thomas Pfleger. *Astronomy on the Personal Computer*. Springer Berlin Heidelberg, Berlin, Heidelberg, second edition, 1994.

[13] Oliver Montenbruck and E. Gill. *Satellite Orbits: Models, Methods, and Applications*. Springer-Verlag, Berlin, 3rd edition, 2005.

[14] Lenonard Meriovitch. *Methods of Analytical Dynamics*. Dover Publications, Inc., 2003

[15] Francois Sanson and Carolin Frueh. *Noise Estimation and Probability of Detection in non-resolved Images: Application to Space Object Observation,* Advances in Space Research, submitted 2018

[16] Roshan Thomas Eapen and Carolin Frueh. *Averaged Solar Radiation Pressure Modeling for High Area-to-Mass Ratio Objects in Geosynchronous Orbits*, Advances in Space Research, Vol. 63, Issue 1, pp. 127-141, doi:10.1016/j.asr.2018.03.042, 2018

[17] Space Operations, Joint Operations, 2020.

[18] Flury, W. Contant, J.. (2001). The updated IAA position paper on orbital debris. 473. 841-849.

[19] United States Strategic Command. Combined Space Operations Center, July 2018.

[20] David Vallado, Paul Crawford, Richard Hujsak, and T.S. Kelso. *Revisiting Spacetrack Report 3*. In AIAA/AAS Astrodynamics Conference, pages 1-94, Keystone, Colorado, 2006.

[21] Wei Dong and Zhao Chang-yin. *An Accuracy Analysis of the SGP4/SDP4 Model.* Chinese Astronomy and Astrophysics, 34(1):69-76,2010

[22] T. Flohrer, H. Krag, H. Klinkrad, B. Bastida Virgili, C. Früh, *Improving ESA's Collision Risk Estimates by an Assessment of the TLE orbit errors of the US SSN Catalogue.* In Proceedings of the Fifth European Conference on Space Debris, ESOC, Darmstadt, Germany, 30 March-2 April 2009.

[23] "Astrodynamics/N-Body Problem - Wikibooks, open books for an open world", *En.wikibooks.org* Available: https://en.wikibooks.org/wiki/Astrodynamics/N-Body_Problem.

[24] Frueh, C., Fiedler, H., and Herzog, J., "Heuristic and Optimized Sensor Tasking Observation Strategies with Exemplification for Geosynchronous Objects," *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 5, 2018, pp. 1036–1048. doi: 10.2514/1.G003123.

[25] F. Sanson, C. Frueh, "Noise Estimation and Probability of Detection in non-resolved Images: Application to Space Object Observation", *Advances in Space Research*, submitted 2018

[26] P.C. Mahalanobis. "On The Generalized Distance in Statistics". In *National Institute of Science*, volume 2, pages 49-55, India, 1936

[27] Little, B. D., and Frueh, C., "SSA Sensor Tasking : Comparison of Machine Learning with Classical Optimization Methods," *Advanced Maui Optical and Space Surveillance Technologies Conference*, 2018, pp. 1–17.

[28] Ackermann, M., Kiziah, R., Zimmer, P., and J.T. McGraw and Associates, "Weather Considerations for Ground-Based Optical Space Situational Awareness Site Selection", *Advanced Maui Optical and Space Surveillance Technologies Conference*, 2018.

[29] Friedman, A. M., and Frueh, C., "Determining characteristics of artificial near-Earth objects using observability analysis," *Acta Astronautica*, Vol. 144, 2018, pp. 405–421. doi:10.1016/j.actaastro.2017.12.028, URL https://linkinghub.elsevier. com/retrieve/pii/S0094576517307737.

[30] C. Früh, M. Jah, "Efficient High Area-to-Mass ratio (HAMR) Object Propagation including Self-Shadowing," Acta Astronautica, Vol. 95, pp. 227 - 241, 2014

[31] J. R. Janesick, T. Elliott, S. Collins, M. M. Blouke, J. Freeman, "Scientific Charge-Coupled Devices," Opt. Eng. 26(8) 268692 (1 August 1987) https://doi.org/10.1117/12.7974139

[32] Virtanen, J., Poikonen, J., Säntti, T., Komulainen, T., Torppa, J., Granvik, M., Muinonen, K., Pentikäinen, H., Martikainen, J., Näränen, J., Lehti, J., and Flohrer, T., "Streak detection and analysis pipeline for space-debris optical images", *Advances in Space Research*, Vol. 57, No. 8, 2016, pp. 1607-1623. https://doi.org/10.1016/j.asr.2015.09.024

[33] Schildknecht, T., Schild, K., Vannanti, A., 2015. "Streak detection algorithm for space debris detection on optical images". In: *Advanced Maui Optical and Space Surveillance Technologies Conference*.

[34] Sanson, F., and Frueh, C., "Quantifying uncertainties in signal position in non-resolved object images: Application to space object observation", *Advances in Space Research*, Vol. 63, No. 8, 2019, pp. 2436-2454. https://doi.org/10.1016/j.asr.2018.12.040

[35] Hagen, N., and Dereniak, E., "Gaussian profile estimation in two dimensions", Applied Optics, Vol. 47, No. 36, 2008, p. 6842. https://doi.org/10.1364/AO.47.006842

[36] Houtz, N., Frueh, C., 2018. "Streak detection and characterization in ground-based optical observations of space objects". AAS/AIAA Astrodynamics Specialist Conference, Snowbird, Utah, August.

[37] Merline, W., and Howell, S., "A realistic model for point-sources imaged on array detectors: The model and initial results", Experimental Astronomy, Vol. 6, No. 1-2, 1995, pp. 163-210. https://doi.org/10.1007/BF00421131

[38] Schildknecht, T., Hugentobler, U., and Ploner, M., "Optical surveys of space debris in Geosynchronous Earth Orbit", *Advances in Space Research*, Vol. 23, No. 1, 1999, pp. 45-54. https://doi.org/10.1016/S0273-1177(98)00229-4

[39] R. Linares and R. Furfaro. "An Autonomous Sensor Tasking Approach for Large Scale Space Object Cataloging." In *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*, pages 1-17, 2017

[40] R. S. Erwin, P. Albuquerque, S. K. Jayaweera, and I. Hussein. "Dynamic sensor tasking for Space Situational Awareness." In *Proceedings of the 2010 American Control Conference*, pages 115-124, 2010

[41] C. Frueh. "Sensor Tasking for Mulit-Sensor Object Surveillance." In *7th European Conference on Space Debris*, Darmstadt, Germany, Apr. 2017.

[42] A. D. Jaunzemis, M. J. Holzinger, and M. K. Jah. "Evidence-based Sensor Tasking for Space Domain Awareness." In *Advanced Maui Optical and Space Surveillance Technologies Conference*, Maui, HI, 2016. Maui Economic Development Board.

[43] Schildknecht, T., "Optical surveys for space debris", *The Astronomy and Astrophysics Review*, Vol. 14, No. 1, 2007, pp. 41-111. https://doi-org.ezproxy.lib.purdue.edu/10.1007/s00159-006-0003-9

[44] Linares R. and Furfaro R., "Dynamic Sensor Tasking For Space Situational Awareness via Reinforcement Learning." In *Advanced Maui Optical and Space Surveillance Technologies Conference 2017*, Maui, HI, 2017. Maui Economic Development Board.

[45] K. Hill, P. Sydney, K. Hamada, R. Cortez, K. Luu, M. Jah, P. Schumacher, M. Coulman, J. Houchard, and D. Naho'olewa, "Covariance-Based Network Tasking for Optical Sensors." In *Advances in the Astronautical Sciences*, 136(July):769-786, 2010

[46] Z. Sunberg, S. Chakravorty and R. S. Erwin, "Information Space Receding Horizon Control for Multisensor Tasking Problems," in IEEE Transactions on Cybernetics, vol. 46, no. 6, pp. 1325-1336, June 2016, doi: 10.1109/TCYB.2015.2445744.

[47] J. R. Shell, "Optimizing Orbital Debris Monitoring with Optical Telescopes", In *Advanced Maui Optical Space Surveillance Technologies Conference*, 2010.

[48] Africano, J., Schildknecht, T., Matney, M., Kervin, P., Stansbery, E., and Flury, W., "A Geosynchronous Orbit Search Strategy," Space Debris, Vol. 2, No. 4, 2000, pp. 357–369. doi:10.1023/B:SDEB.0000030025.04930.08

[49] Ronald P.S. Mahler. Multitarget Bayes Filtering via First-Oder Multitarget Moments. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1152-1178, 2003.

[50] William E. Wiesel. *Modern Orbit Determination*. Aphelion Press, 2003

[51] Kyle T. Alfriend and Inkwan Park. When Does the Uncertainty Become Non-Gaussian. In *Advanced Maui Optical and Space Surveillance Technologies Conference*, 2016

[52] R. De Maesschalck, D. Jouan-Rimbaud, and D.L. Massart. The Mahalanobis Distance. *Chemometrics and Intelligent Laboratory Systems*, 50(1):1-18, Jan 2000.

[53] Mihir Patel, Andrew J. Sinclair, and Koki Ho. Information-Theoretic Target Search for Space Situational Awareness. In *2018 Space Flight Mechanics Meeting*, Chapter AIAA SciTe. American Institute of Aeronautics and Astronautics, Kissimmee, Florida, 2018

[54] Ronald P.S. Mahler. A Theoretical Foundation for the Stein-Winter "Probability Hypothesis Density (PHD)" Multitarget Tracking Approach. In *The 2000 MSS National Symposium on Sensor and Data Fusion*, pages 99-117, San Antonio, TX, 2000 Army Research Office Alexandria VA.

[55] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2014

[56] Kazuo Murota. *Discrete Convex Analysis*. Society for Industrial and Applied Mathematics, Jan 2003.

[57] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement Learning: A survey. In *Journal of Artificial Intelligence Research*, 4:237-285, 1996.

[58] David L. Neel and Nancy Ann Neudauer. Matroids You Haven Known *Mathematics Magazine*, 82(1):26-41, Feb 2009.

[59] Robert A. Murphey. *Target-Based Weapon Target Assignment Problems*, chapter Target-Bas, pages 39-53. Kluwer Acadmic Publishers, 2000

[60] Marco Dorgio and Luca Maria Gambardella. Ant Colonies for the Travelling Salesman Problem. *Bio Systems*, 43(2):73-81, 1997

[61] Marco Dorgio, Mauro Birattari, Thomas Stutzle. Ant Colony Optimization. *IEEE Computational Intelligence Magazine*, 1(4)28-39, 2006

[62] Carlos E. Mariano and Eduardo F. Morales. DQL: A New Updating Strategy for Reinforcement Learning Based on Q-Learning. In L. De Raedt and P. Flach, editors, *Machine Learning: ECML*, volume 2167, pages 324-335. Springer, Berlin, Heidelberg, 2001.

[63] T. Schildknecht. Optical Surveys for Space Debris. *Astron. Astrophys. Rev.,* 14:14-111, DOI 10.2007/s00159-006-003-9, 2007.

[64] C. Früh, T. Schildknecht, Accuracy of Two Line Element Data for Geostationary and High-Eccentricity Orbits, AIAA, Guidance, Control and dynamics, Vol. 35, No. 5, pp. 1483-1491, doi:10.2514/1.55843, 2012

[65] "Solar Simulation - Spectral Irradiance - AM0-AM40 — AM1.5G", G2V Optics Available: https://g2voptics.com/solar-simulation/.

[66] ESA Space Operations. Space Situational Awareness Programme Overview, 2017.

[67] Erin Salinas. Space Situational Awareness is Space Battle Management. May 2018

[68] Space Debris Office. European Space Agency's Annual Space Environment Report. Technical Report, European Space Agency, 2020

[69] NASA Orbital Debris Program Office. 2020. "Orbital Debris", *Orbitaldebris.jsc.nasa.gov.* Available: `https://www.orbitaldebris.jsc.nasa.gov/quarterly-news/pdfs/odqnv24i1.pdf`

[70] Johannes Herzog. *Cataloguing of Objects on High and Intermediate Altitude Orbits*. Dissertation, University of Bern, 2013

[71] Edward P. Chatters and Brian J. Crothers. Space Surveillance Network. In Brian Tichenor, editor, *AU-18 Space Primer*, chapter 19, pages 248-258. Air University Press, 2009.

[72] Regan, D., "Modular Neural Network Tasking of Space Situational Awareness Systems", AMOSTech, 2018.

[73] Dararutana, K., "Comparison of Novel Heuristic and Integer Programming Schedulers for the USAF Space Surveillance Network", *Wright-Patterson Air Force Base: AFIT*, 2020.

[74] Dzamba, T., and Enright, J., "Ground Testing Strategies for Verifying the Slew Rate Tolerance of Star Trackers", MDPI Sensors, Vol. 14, No. 14, 2014. doi:10.3390/s140303939

[75] Runge, C. Ueber die numerische Auflösung von Differentialgleichungen. Math. Ann. 46, 167–178 (1895). https://doi.org/10.1007/BF01446807

[76] "What's a good value for R-squared?", Linear regression models Available: https://people.duke.edu/ rnau/rsquared.htm.