DISSERTATION

RESOURCE ALLOCATION FOR SPACE DOMAIN AWARENESS AND SYNTHETIC APERTURE RADAR

Submitted by

Naomi Owens-Fahrner

Department of Mathematics

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2022

Doctoral Committee:

Advisor: Margaret Cheney

Jennifer Mueller
Patrick Shipman
Venkatachalam Chandrasekar

ABSTRACT

RESOURCE ALLOCATION FOR SPACE DOMAIN AWARENESS AND SYNTHETIC APERTURE RADAR

In this thesis, we will address two resource allocation problems. For each of these problems, the objective will be to make use of the resources in an optimal way. We will consider the Space Domain Awareness (SDA) sensor tasking problem as well as the Synthetic Aperture Radar (SAR) flight path planning problem.

We will first present a new objective function for the problem of Space Domain Awareness resource allocation (SDARA) as well as a novel algorithm to maximize this new objective function. This SDARA problem aims to maximize the total number of targets seen while minimizing resource costs. These resources, namely the optical sensors, are assumed to be heterogeneous and have different associated tasking costs.

The novel algorithm, called the "block greedy" algorithm, provides an approximate regional maximum of this objective function in a tractable amount of time. The block greedy algorithm is a hybrid of the weapon-target-assignment and greedy algorithms. This algorithm will be shown to outperform common algorithms used in solving the SDARA problem.

Second, we will present an approach to create an optimal SAR flight path by varying the vehicle's heading, pitch, and antenna steering angles. An optimal flight path is one in which the scene coverage and resolution are maximized. We will utilize the data-collection manifold as a tool to measure scene resolution. We will then add a scene coverage consideration to build an objective function in which we can plan an optimal flight path for an aircraft.

After this, we will consider many extensions and applications of using this objective function. These include adding a signal-to-noise ratio (SNR) consideration to SAR flight path planning. Additionally, we will extend this objective function to include multiple unmanned aerial vehicle

(UAVs) for optimal flight paths for a SAR system. We will use our objective function to optimally plan flight paths for multiple UAVs.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Dr. Margaret Cheney. Not only have you been a great resource for all dissertation related content, you have also been a mentor for me professionally and personally. You have provided a welcoming environment for me to approach you with questions and new ideas. I am thankful for your gracious approach to advising. The work of a PhD student is not easy, and I am grateful for your kindness, wisdom, and expertise.

I would also like to thank all of my fellow graduate students, especially Lara Kassab, Emily Heaver, Brittany Story, Shannon Golden, Richard Sampera, Danny Long, and Lander Ver Hoef. I am a better mathematician because of you all. Graduate school would have been a completely different and lesser experience without the community and support you have provided.

I would also like to thank my parents, Tom and Lois Owens. I would not be who I am without your love and support. Thank you for seeing my potential and pushing me to be the best I can be.

Last, and very far from least, I want to thank my husband. Caleb, you know better than anyone the undertaking this process has been. Thank you for walking through this with me and always believing in me. I am forever thankful for your love, friendship, and support.

My work on the Space Domain Awareness sensor tasking problem was conducted at Ball Aerospace. I would like to thank Ball Aerospace for funding this project. Additionally, I would like to thank my Ball Aerospace teammates for their help in my understanding of Space Domain Awareness and guiding me through quality software practices.

The remainder of this dissertation work was supported by the U. S. Office of Naval Research under award number N00014-21-1-2145 and by the Air Force Office of Scientific Research under award numbers FA9550-18-1-0087 and FA9550-21-1-0169. I would like to thank both of these sources for their funding.

DEDICATION

To my husband, Caleb, and my parents, Tom and Lois

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

Resource allocation problems have to do with assigning resources to tasks in a manner that optimizes the goals of that application. Resource allocation problems appear in many applications, including Space Domain Awareness (SDA) sensor tasking, Synthetic Aperture Radar (SAR) flight path planning for aircrafts and unmanned aerial vehicles (UAV) [11], and broadly the traveling salesperson problem (TSP). In this thesis, we will primarily examine the problems of SDA sensor tasking and SAR flight path planning. Both of these problems have important applications for government and industry work.

1.1 Outline

First, we will start with the history of SDA. We will cover an extensive literature review of our SDA resource allocation problem. This literature review will include algorithms used to solve the SDA sensor tasking problem. We will show that the new algorithm in Section 2.3 has not been implemented for this problem.

After this, we will discuss the history of SAR. This will be followed by an extensive literature review of the SAR flight path problem with a single aircraft, as well as with multiple UAVs. We will show that the presented SAR flight path planning work has not been done before.

In Chapter 2, we will begin by covering background material on space domain awareness. We will discuss the specific SDA sensor tasking problem that is the first project in this dissertation. The various coordinate systems used in SDA applications will be discussed. Additionally, we will discuss factors involved in calculating an object's SNR.

After this, we will dive into the SDA sensor tasking problem. We will formulate our objective function, along with our constraints. We will list common algorithms used to solve the SDA sensor tasking problem and outline the algorithms we implemented.

From there, we will describe the block greedy algorithm. We will cover the pros and cons of using this algorithm and compare it with the algorithms previously implemented. We will then perform an extensive analysis of all of the algorithms implemented. We will compare various features of the algorithms and show that the block greedy algorithm outperforms all of these algorithms. Finally, we will end Chapter 2 by summarizing our results for the SDA sensor tasking problem.

In Chapter 3, we will begin by covering an overview of synthetic aperture radar (SAR). After this, we will describe in detail our flight path planning problem. We will first cover our extension of the work done in Dr. Scott Altrichter's thesis [10]. Dr. Altrichter developed an objective function with which to maximize SAR image resolution as well as scene coverage. He used the 3D components of the data-collection manifold to build this objective function.

In his thesis, Dr. Altrichter optimized the aircraft's flight path with respect to the aircraft's heading and pitch. We first modify his objective function by only using the 2D components of the data-collection manifold. After this, we extend this work by maximizing the objective function with respect to the aircraft's heading, pitch, and steering angles. Lastly, we extend Dr. Altrichter's optimization process by implementing a greedy algorithm.

We will compare the result from optimizing the objective function with other SAR modes. Stripmap, spotlight, and circular are all common SAR modes. We compare these with the flight path generated from the objective function. We show that the optimized flight path outperforms the other common flight paths.

In Chapter 4 we discuss extensions and application of the formulated objective function. We begin by adding an SNR component to the objective function in order to model a more real-life scenario. We also include a target or target scene priority of visitation.

Lastly, in Chapter 5 we implement additional optimization algorithms for our objective function. In Chapter 3, we use a greedy algorithm to maximize the objective function. In Chapter 5, we also consider a random search algorithm and genetic algorithm to maximize the objective function. We compare the results of all of these algorithms to determine the genetic algorithm works best for this problem. Chapter 5 ends by detailing conclusions and proposed future work for this problem.

1.1.1 Thesis Contributions

The contributions of this dissertation include the following items:

- In the Space Domain Awareness sensor tasking problem:
  - Developed a new objective function by incorporating multiple, heterogeneous sensors with different tasking costs
  - Added real-life constraints to the objective function
  - Computed the complexity of brute-force computing the globally optimal solution
  - Developed the 'Block Greedy' algorithm
  - Implemented five optimization algorithms and compared their performance
  - Presented and published a paper at the Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS), titled, 'A Regional Greedy Algorithm for Space Domain Awareness Resource Allocation'
- In the Synthetic Aperture Radar flight path planning problem:
  - Modified Dr. Scott Altrichter's objective function to include only the 2D data-collection manifold information
  - Added the steering angles as arguments to maximize the objective function, thus maximizing the objective function with respect to the aircraft's heading, pitch, and steering
  - Added real-life constraints to the objective function
  - Quantified the way in which an aircraft's flight path impacts SNR
  - Added the effect of SNR to the objective function
  - Added a priority of visitation to specific targets or target scenes
  - Modified the objective function to generate optimal flight paths for multiple UAVs
  - Implemented a total of three optimization algorithms and compared their performance

– Submitted a portion of this work for publication in the IEEE Transactions on Aerospace and Electronic Systems journal, titled, 'Synthetic Aperture Radar Flight Path Optimization for Resolution and Coverage'. Additional authors on this paper include Dr. Scott Altrichter and Dr. Margaret Cheney.

1.2 Space Domain Awareness

1.2.1 History of SDA

The first satellite, Sputnik, was launched into space by the Soviet Union on October 4, 1957 [12]. Since then, thousands of more satellites have been launched into space. By 2019, there were over 2,500 active satellites in orbit [1]. Below is a graph of the number of satellites launched into space per year since 1957 through 2019.

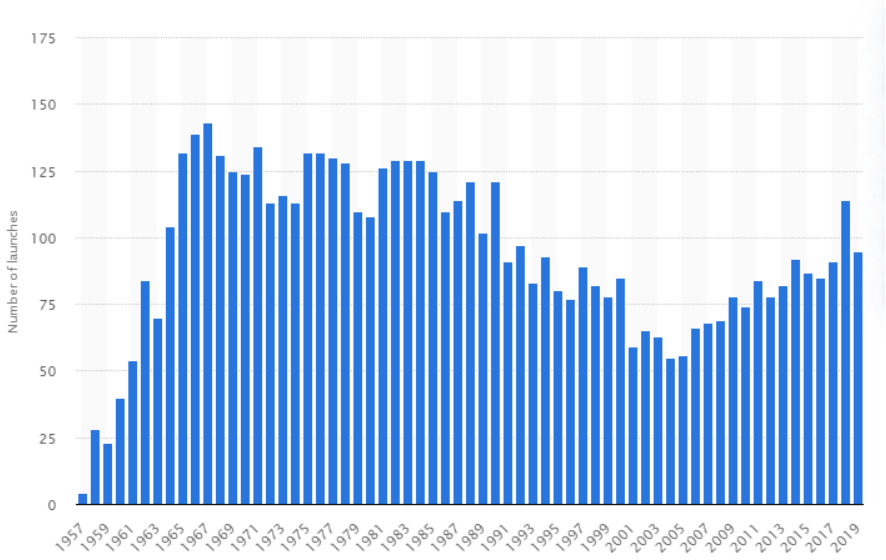

Figure 1.1: Number of satellites launched from 1957 to 2019. [1]

The first half of the 20th century saw the race to space. In the second half of the 20th century, that race was won. A man was put on the moon, and many other countries successfully launched their own satellites into space. There are currently around 6,000 total satellites in space (including those that are inactive). [1].

Space Domain Awareness (SDA) has to do with tracking and observing resident space objects (RSOs). These RSOs consist of any human-made object in orbit. In addition to satellites, other RSOs include space debris. Space debris is any human-made object in orbit that no longer functions. Space debris may range from no longer functional spacecraft to fragmentation from rocket bodies or spacecraft.

It is estimated that current space debris consists of 34,000 objects greater than 10 cm, 900,000 objects from 1 cm to 10 cm, and 128 million objects from 1 mm to 1 cm [1]. Space debris of any size poses a threat to the safety of operational spacecrafts. Thus, it is imperative to track space debris as well as spacecrafts.

With the increase of satellites in space, space debris also increases. Given the total population of RSOs, it is crucial that these are observed and tracked. As more objects are launched into space, this need is ever-increasing.

1.2.2 Literature Review

Before presenting our work in the area of SDA sensor tasking, we will summarize our literature review of the subject. The objective function used in the following work is an extension of Frueh's objective function [13]. In this objective function, the sensor costs are not considered, whereas we consider these costs in our objective function. Additionally, we consider additional constraints to further create a realistic scenario.

There are other applications of resource allocation problems that have considered the cost of tasking resources [14]. However, these have not been considered in the SDA sensor tasking problem. Cost has been implicitly considered by maximizing general sensor use, however the assumption has been that all sensors have the same tasking cost. In the following work, we consider the cost of heterogeneous sensors. We assume that each sensor has a unique cost associated with its tasking.

Some have formulated the sensor tasking problem as deterministic but subject to uncertainty due to weather and ad hoc collections [15]. Others have formulated this problem based on posi-

tion/velocity estimates and covariances [16]. We will not consider uncertainties such as these in our work as this leads in a direction different from our problem.

There have been many algorithmic approaches to solving the SDA sensor tasking problem. Although there has been some work done in machine learning to approach this problem [17], we will focus on more traditional optimization methods. Common optimization methods include simple greedy algorithms, weapon-target-assignment algorithms [18], and genetic algorithms [19] [20].

We will describe a regional greedy algorithm, called the “block greedy" algorithm [21]. This algorithm is innovative in the area of SDA sensor tasking. We will compare the block greedy algorithm with a simple greedy algorithm, weapon-target-assignment algorithm, random search algorithm, and genetic algorithm.

1.3 Synthetic Aperture Radar

1.3.1 History of SAR

Radar was a crucial tool used for target detection during World War II. It was not until 1951 that Synthetic Aperture Radar (SAR) was invented by Carl Wiley [22]. Up until that point in time, in order to maintain decent image resolution, exceedingly large antennas were needed. Wiley discovered a way to overcome this problem by using the motion of the antenna to synthesize a larger antenna. Thus, an antenna small enough to fit on an aircraft could be used. In 1954 Wiley patented his idea as “Simultaneous Buildup Doppler". This idea he patented would later be called Synthetic Aperture Radar (SAR).

Since then, SAR has been used in many applications. SAR has found its way into areas such as satellite and remote sensing [23]. SAR can be used to record the Earth’s surface deformation. SAR has even been used in remote sensing of wildfires. Since its creation, SAR has become an instrumental tool used in a plethora of fields.

Synthetic Aperture Radar imaging consists of forming images of the ground from an aircraft or satellite. It uses electromagnetic wavelengths longer than optical or laser wavelengths. Radio waves range from around 1 mm to 100 km [24].

Most optical imaging systems require an external source of light, whereas SAR provides its own illumination. Additionally, optical imaging is greatly affected by deterrents such as fog, rain, or snow, whereas SAR is much less affected. Synthetic Aperture Radar imaging works regardless of weather or time of day. This makes SAR an appealing system for imaging.

The motion of the SAR vehicle allows us to synthesize a larger antenna than is actually used. In doing this, better resolution is achieved. We are able to further maximize image resolution, as well as scene coverage, by varying the SAR aircraft flight path.

1.3.2 Literature Review

Optimizing a SAR flight path using a single aircraft is a fairly new area of research. Some applications include 3D reconstruction using bistatic SAR [25], environmental monitoring, and disaster warning [26]. In our application of SAR flight path planning we will maximize scene resolution and coverage. We will then extend this to planning flight paths for multiple UAVs.

The use of SAR for a single UAV has been studied in [27] [28] [29]. In [30], a flight path is planned for a UAV in order to see a specific target or set of targets. Resolution is important, but it is not a parameter that is maximized. Additionally, in [27], a 2D flight path is planned for a UAV. Here, the goal is to cover a multi-region area.

In Altrichter's thesis [10], he proves that the surface area of the data-collection manifold (DCM) is directly related to the image resolution at a point. In order to define an objective function to maximize scene coverage and resolution, the manifold contributions for each point target are summed. This function is maximized with regard to the aircraft's heading and pitch angles, thus creating a 3D optimal flight path.

None of the above papers describe a flight path planning method like the one performed by Dr. Altrichter. We have further extended the work [10] by optimizing the objective function with

respect to the aircraft's steering angles. Additionally, we have extended this work by adding a weighting option for targets that have a higher priority to visit. We also have added an SNR consideration to the objective function.

In Section 4.3 we extend this work to multiple, mono-static UAVs. Unmanned aerial vehicles are the way of the future. There are many benefits of using UAVs for SAR, including the safety of military personnel, UAV maneuverability, the low cost of UAVs, and the capability to recover after the loss of a UAV. If we have access to multiple UAVs and some become inactive, we are still able to collect data from the remaining UAVs. Additionally, multiple UAVs are able to collect more data at one time than a single aircraft.

Flight path planning with multiple UAVs for SAR application has been briefly studied [31]. However, the goals of these current studies is to either maximize scene coverage [30] or resolution [32]. In this thesis, we maximize both scene coverage and resolution. We will use an extension of the objective function formulated in [10] in order to accomplish this.

Chapter 2

Space Domain Awareness Sensor Tasking

The entirety of the work presented in this chapter was funded by Ball Aerospace.

2.1 Background

Recall, Space Domain Awareness (SDA) has to do with tracking and observing resident space objects (RSOs). In SDA, often multiple sensors are used in order to observe and track RSOs. We will assume all of the sensors in this problem are optical. These sensors may be homogeneous or heterogeneous. Either way, depending on factors such as the constraints of each sensor, the cost to use each sensor, the number of available targets for each sensor, etc., it may be more cost-effective to task sensors unevenly. For instance, if one sensor is able to collect more images than a different sensor, that first sensor may be used more often. The resources in this resource allocation problem are the fields of view of the sensors.

2.1.1 Goals

With the rise in resident space objects (RSOs), SDA has become imperative. We are currently limited in the number of resources to observe RSOs. Thus, we want to use as few resources as possible. The problem of SDA sensor tasking aims to maximize the total number of RSOs detected, while also minimizing observer costs. We optimally allocate our resources in order to accomplish these objectives.

In order to optimally allocate resources for the SDA sensor tasking problem, we create a schedule of observers that consists of tasks to be completed over some time interval. Generally, one task will be assigned to each observer per time window.

Most formulations of the SDA sensor tasking problem do not consider heterogeneous sensors with varying sensor costs. We will address this real-life scenario. We will consider heterogeneous sensors that are GEO, LEO, and Earth-based.

There are many algorithms currently used to solve the SDA sensor tasking problem. In this thesis, we examine these algorithms along with a novel algorithm we created for this application. We will call this novel algorithm the “block greedy" algorithm. After implementing the traditional algorithms for the SDA sensor tasking problem, we are able to compare them to the block greedy algorithm.

This block greedy algorithm is a hybrid algorithm that combines the simple, local greedy algorithm with the regional weapon-target-assignment algorithm. This hybrid algorithm has been named the block greedy algorithm since it looks across some “block" of time windows. We look at the traditional approaches to the SDA sensor tasking problem and show how the hybrid block greedy approach outperforms these approaches.

In constructing our problem, we make the following assumptions:

1. We know the observers’ exact positions.
2. We know the position of the objects we want to detect.
3. These objects propagate at the same velocity the Earth rotates. Thus, the position of these objects relative to the Earth remain constant.
4. The probability of detecting an object is normally distributed. This will be computed using the cumulative distribution function (cdf) of a normal distribution.

2.1.2 Definitions

Before we go further, we must first define some common terminology in SDA applications. For our SDA application, we will primarily be concerned with objects in a geosynchronous equatorial orbit (**GEO**) and a low Earth orbit (**LEO**). The GEO-belt is located at an altitude of about 36,000 km and the LEO belt is located at an altitude between 160 km and 2,000 km [2]. The belt in between the GEO- and LEO-belts is called the Medium Earth Orbit (**MEO**).

We consider observers that lie in the GEO-belt, LEO-belt, or ground. The targets we observe lie in the GEO-belt. Below is a figure of these orbit regimes:

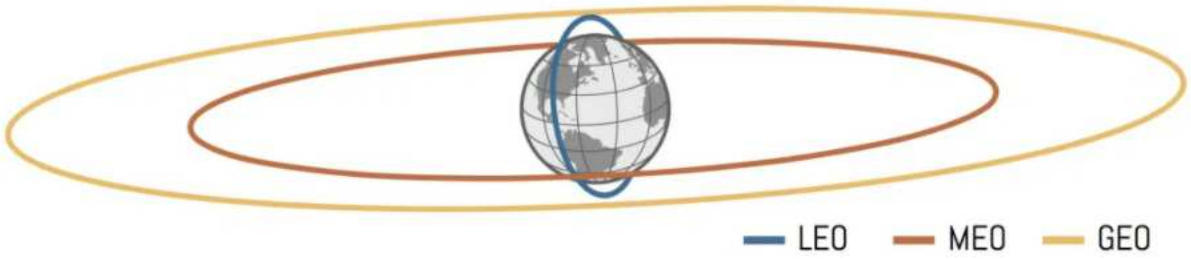

Figure 2.1: Popular Orbit Regimes. Low Earth Orbit (LEO) is shown in blue, Medium Earth Orbit (MEO) is in red, and Geosynchronous Orbit (GEO) is in yellow [2].

Around 55% of all active satellites lie in LEO, including the International Space Station [2]. It takes a satellite in LEO around 90 minutes to 2 hours to complete an orbit. Some satellites in this orbit pass over the Earth's poles, as shown in figure 2.1. This is called a polar orbit.

Only about 10% of all active satellites lie in MEO. This is largely due to the Van Allen belts. The Van Allen belts trap charge from solar winds and cosmic rays. This charge can interfere or damage satellite electronics.

The remaining 35% of active satellites lie in GEO. Most of these satellites orbit above the Earth's equator. These are called geostationary since they appear in the same part of the sky from any place on Earth. Since satellites in GEO do not move with respect to the Earth, their orbit is exactly one day.

Two-line element sets (**TLE**) are data sets that contain information on the position of an object. Given a time, a TLE will be able to describe an object's position and velocity with relative accuracy. Due to perturbations in orbit propagation, there are various models to compute position and velocity from a TLE.

As a note, determining the exact position and velocity of a space-propagating object is difficult. There are many factors involved in the propagation of an object in space. This problem is one of dynamical systems and is an entire field of study by itself. Hence, we will not address the effects of perturbations in orbit propagation. Instead, we will assume that we know the exact position and velocity of our orbiting objects by using the SGP4 propagation model [33].

2.1.3 Coordinate Systems

On Earth, we typically describe a position or location in terms of latitude, longitude, and altitude. When objects are space-based, we typically use different coordinate systems. The two main coordinate systems we will use are the Earth-Centered, Earth-Fixed coordinate system (**ECEF**) and the Earth-Centered Inertial coordinate system (**ECI**). Below is a table summarizing the main coordinate systems:

System	Symbol	Origin	Fundamental Plane	Principal Direction	Example Use
			Interplanetary systems		
Heliocentric	XYZ	Sun	Ecliptic	Vernal equinox	Patched conic
Solar system	$X_B Y_B Z_B$	Barycenter	Invariable plane	Vernal equinox	Planetary motion
			Earth-based systems		
Geocentric*	IJK	Earth	Earth equator	Vernal equinox	General
Earth-Moon (synodic)	$I_S J_S K_S$	Barycenter	Invariable plane	Earth	Restricted three-body
Earth-centered Earth-fixed	$ECEF$ or $(IJK)_{ECEF}$	Earth	Earth equator	Local meridian†	Observations
Topocentric horizon	SEZ	Site	Local horizon	South	Radar observations
Topocentric equatorial	$I_t J_t K_t$	Site	Parallel to Earth equator	Vernal equinox	Optical observations
			Satellite-based systems		
Perifocal**	PQW	Earth	Satellite orbit	Periapsis	Processing
Satellite radial	RSW††	Satellite	Satellite orbit	Radial vector	Relative motion, Perturbations
Satellite normal	NTW	Satellite	Satellite orbit	Normal to velocity vector	Perturbations
Equinoctial	EQW	Satellite	Satellite orbit	Calculated vector	Perturbations

*We specify a particular epoch for the J2000, or *ECI* system. This actually applies to all frames using ♈.

†If we use the Greenwich meridian, we use an *EFG* designation

**To form the nodal coordinate system, we use the ascending node instead of the perigee direction.

††A roll-pitch-yaw system results from rotating *RSW* 180° about the *S* axis.

Figure 2.2: Coordinate Systems Summary [3]

The origin of ECEF coordinates is located at the center of mass of the Earth. As its name suggests, this coordinate system is Earth fixed and rotates with respect to the stars. The coordinates

are generally written as right ascension, α, and declination, δ, coordinate pairs, (α, δ). Below is a figure of the Earth in ECEF coordinates:

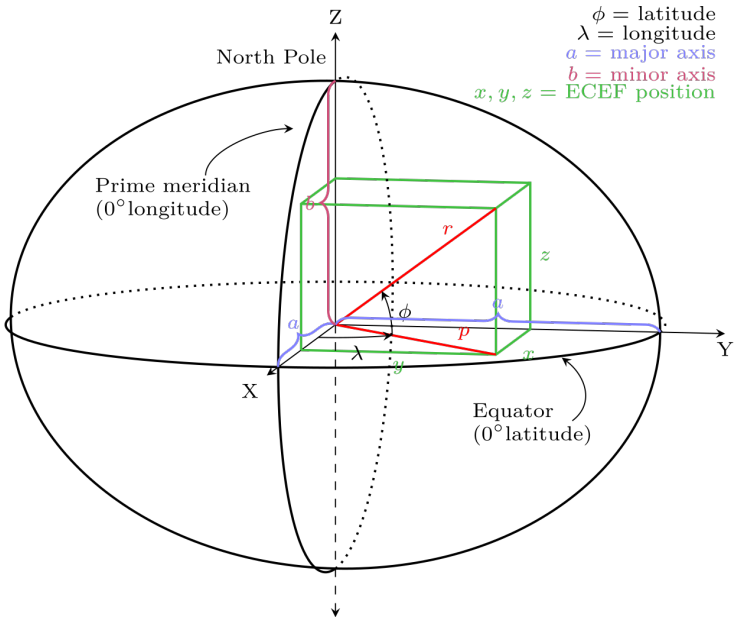

Figure 2.3: ECEF Coordinates [4]

Similar to the ECEF coordinate frame, the ECI coordinate frame has its origin at the center of mass of the Earth. However, this coordinate frame remains fixed with respect to the stars and rotates with respect to the Earth. Figure 2.4 describes a side-by-side comparison of ECI and ECEF coordinate systems:

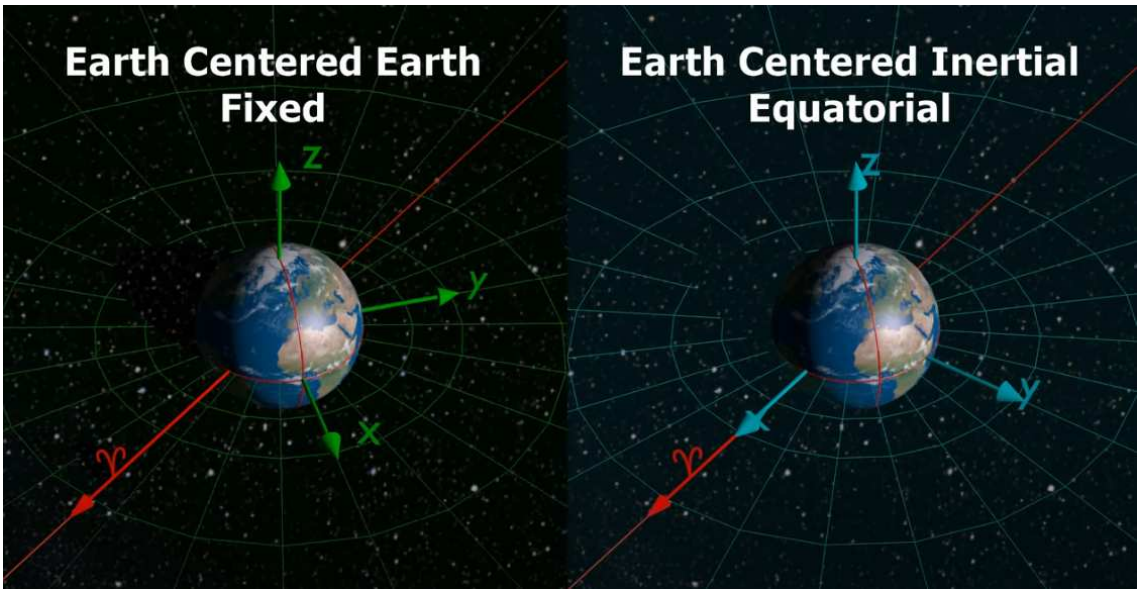

Figure 2.4: ECI and ECEF Coordinate Systems [5]

In figure 2.4, γ represents the vernal equinox. The vernal equinox is the place where the Sun crosses the celestial equator. The celestial equator lies on the same plane as the Earth's equator.

In this figure, the Earth rotates while γ remains fixed. On the left, in the ECEF coordinate frame, the $x, y,$ and z axes rotate along with the Earth so they remain fixed with respect to the Earth. On the right, in the ECI coordinate frame, the Earth rotates, but $x, y,$ and z remain fixed with respect to γ.

2.2 Resource Allocation for Space Domain Awareness

Space domain awareness resource allocation (SDARA) has gained much interest over the last few years. Given the rapid growth of resident space objects (RSOs), this is no surprise. With over 34,000 RSOs greater than 10 cm [1] and only a small fraction of that number of observers, allocating observer resources is imperative.

The resources in this SDARA problem are the fields of view of space domain awareness (SDA) sensors observing targets. The goal of SDA sensor tasking is to maximize the total number of

targets observed while also minimizing resource costs and target revisit time. This is done by constructing an objective function and optimizing this against various constraints.

Current objective function formulations of this SDA problem do not address the different costs associated with tasking heterogeneous sensors [16] [17]. With the growing number of RSOs and limited availability of observers, it is imperative that resource costs be minimized. Given the variation in possible observers, it is crucial to also consider the tasking cost of each observer.

This thesis presents a new objective function for the SDA sensor tasking problem that considers GEO, LEO, and ground-based optical sensors all at once. For this purpose, targets primarily consist of objects in the GEO-belt. This objective function considers the unique cost of moving the look angle of each observer as well as the cost of storing images in each observer.

Additionally, a novel algorithm to maximize this new objective function is discussed. This will be compared with other common algorithms used to optimize common SDA sensor tasking objective functions. It will be shown that this new algorithm outperforms the other algorithms in maximizing this objective function.

In order to allocate resources, we will create a schedule of tasks for each observer. This schedule will consist of a look angle for each observer at each time window. Tasks include slewing to that look angle and collecting images of targets that satisfy the specified constraints. We will choose the task for each observer at each time that maximizes the number of targets seen and minimizes target revisit time.

2.2.1 Schedule Generation

Resources are allocated by creating an optimal schedule for observers. A schedule is comprised of tasks for each observer at each time window over some time interval. This time interval is discretized into time windows. A task consists of slew and/or collect commands. The collect command schedules specific targets to be imaged during the given time window.

The general term for the system that schedules tasks will be called a “scheduler” for the remainder of this thesis. To create a schedule, the scheduler assigns tasks to each observer for each

time window. Algorithms will be implemented in the scheduler in order to create approximate optimal schedules for observers.

To generate a schedule, a search area is first discretized into voxels. For simplicity, a target is assumed to be the center of each voxel. Observers are then tasked to slew and/or collect images of targets. This process continues over all time windows until a schedule is generated [21].

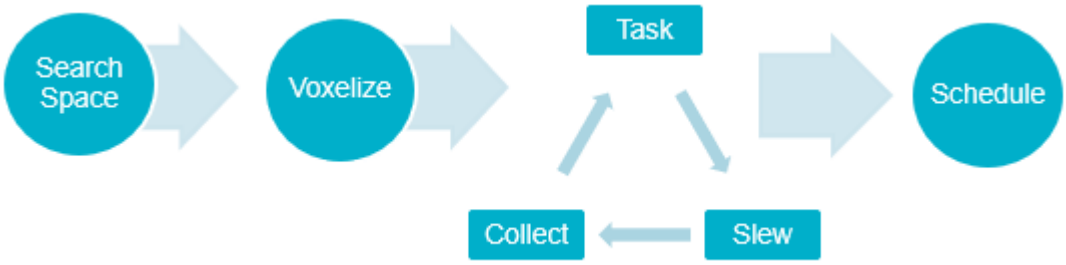

Figure 2.5: Schedule Generation

In figure 2.6, the search space lies in the GEO-belt. This space is discretized to voxels. The center of each voxel is a target. These targets are represented as the red spheres in this figure.

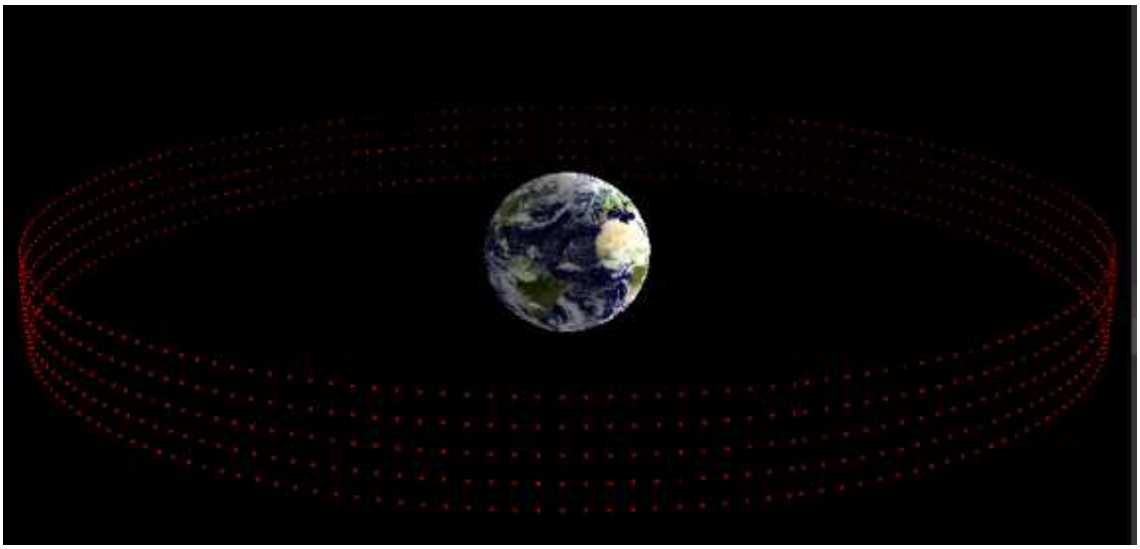

Figure 2.6: Discretized Geo-Belt Space [6]

In order to create an optimal schedule, each task is assigned a score. Factors in this score include a target's priority of visitation, a target's probability of detection, and observer costs. Additionally, there are constraints that must be considered for a target to be tasked. These include the following: a target must be within an observer's field of view, a target must lie outside of the Earth's umbra, the Earth must not block a target from an observer, and a target's SNR must be greater than or equal to an observer's SNR threshold.

Additional constraints to task an observer include the following: an observer's memory must not be full, it must be able to slew to the given position within the given time window, and the task score must be above the observer's score threshold.

2.2.2 Components of Objective Function

We will first define each piece of the objective function. After this, we will put the pieces together to define the full objective function and constraints.

The priority of visiting a target $n \in \mathbb{N}$, denoted $\mu_n : \mathbb{R} \to \mathbb{R}$, is a function of time and can be determined in a variety of ways. For the results to follow, the priority of visiting a target is a linear function of time. In the linear example, the more time that passes since a target was visited, the higher priority of visitation that target has. In addition to a linear priority of visitation, the scheduler can set a constant priority of visitation or a user defined priority of visitation. Once a target is visited, its priority is reset. This is a function of the current time, $t \in \mathbb{R}$, and the last time the target was visited, $t_n \in \mathbb{R}$.

Each target has a probability of being detected by each observer. This is modeled by the cumulative distribution function (cdf) of a normal distribution. Each observer has an SNR threshold in order to detect a target. If a target's SNR value is lower than an observer's SNR threshold, it will be undetectable by the observer. Thus, the probability of detecting a target is a function of the target's SNR and the observer's SNR threshold. This is defined below:

$$p_{ns}\left(r_{ns}, h_s\right) = 0.5\left(1.0 + erf\left(\frac{r_{ns} - h_s}{\sqrt{2}}\right)\right), \tag{2.1}$$

where p_{ns} is the probability of observer $s \in \mathbb{N}$ detecting target n, $r_{ns} : \mathbb{R}^7 \to \mathbb{R}$ is the SNR of target n received by observer s, and $h_s \in \mathbb{R}$ is the SNR threshold for observer s.

The SNR of a target is a function of time, observer position, and target position. Observer position is computed using a SGP4 propagator [33]. The SGP4 propagator takes inputs of time and observer two-line element set (TLE). Thus, $p_{ns} : \mathbb{R}^6 \to \mathbb{R}$ is dependent on observer SNR threshold, time, observer TLE, and target position (since target position is $\in \mathbb{R}^3$).

Each observer has a cost associated with tasking it to various objectives. One of these costs is the cost of storing images, $i_s : \mathbb{R}^2 \to \mathbb{R}$. This function takes inputs of the number of images currently stored, a_s, and the buffer size, b_s. The cost of storing a specific image is found by determining how much storage space remains. It should be noted that observer costs are negative and added to the objective function to eliminate ambiguity.

The other observer cost we will consider is the cost of an observer slewing to a specified position. The cost of slewing is called the pointing score, $p_s : \mathbb{R}^7 \to \mathbb{R}$. This takes inputs of the cost of minimal gimbal movement, $g_s \in \mathbb{R}$, the new position, $\vec{x}_{stk} \in \mathbb{R}^3$, and current position, $\vec{y}_s \in \mathbb{R}^3$. Again, this cost is negative and added to the objective function.

As mentioned, these costs vary per observer. This is done to create more realistic tasking scenarios, as real-life observers have different tasking costs. Depending on observer costs, it may be more beneficial to task some observers more often than others.

The scheduler has real-life constraints in order to schedule a target. The target must be in an observer's field of view, not blocked by the Earth, outside the Earth's umbra, have a high enough SNR, an observer must have available storage space, an observer must be able to slew in the time window, and the task score must be above the score threshold.

Each observer has an associated field of view (FOV) and a field of regard. An observer's FOV is the volume an observer can see at a fixed orientation. Its field of regard is the total volume an observer is capable of viewing if accounting for its movement. Below is a 2-dimensional example of this, where each grid represents an observer's possible FOV, the blue circles represent targets of interest:

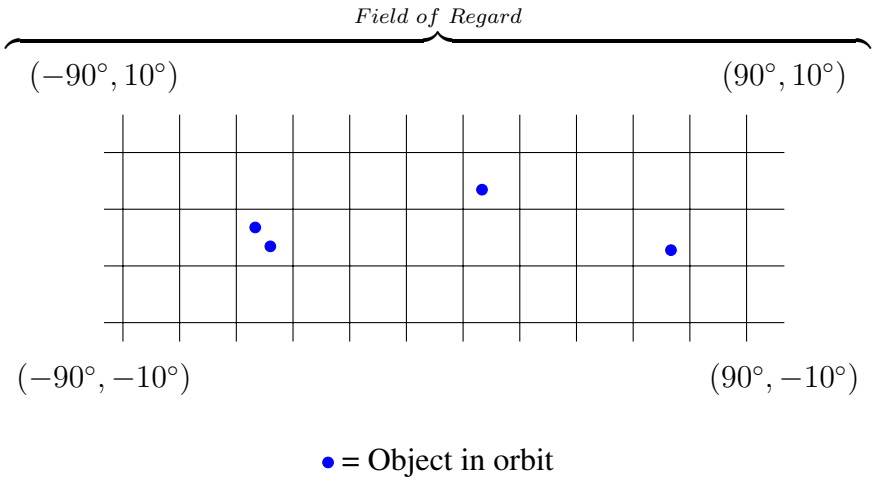

Figure 2.7: Field of View and Field of Regard

As we increase an observer's field of view, more targets are seen. This is shown in figure 2.8. These observers lie in the same location in the GEO-belt. Their field of views were 1.5-deg, 3.0-deg, and 5.0-deg. Note: The rise in score at the end is due to all targets being seen once and beginning to revisit previously seen targets.

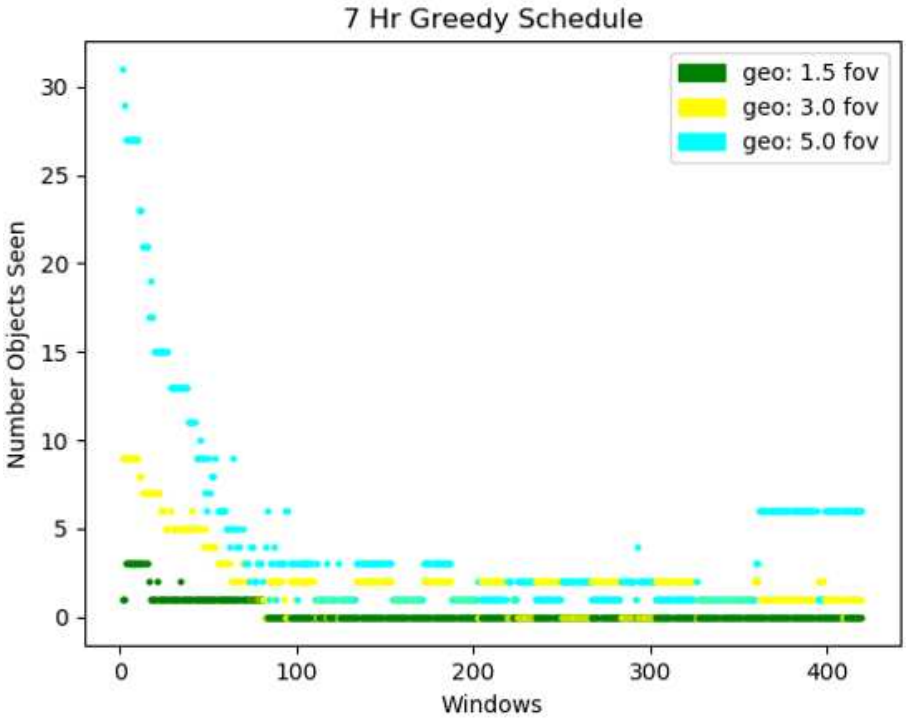

Figure 2.8: Varying Observer Field of View

Each observer has an SNR threshold, $h_s \in \mathbb{R}$, at which it can see a target. If the received SNR from a target to an observer, r_{ns}, is lower than the observer's threshold, the target will not be seen by that observer at that time. Figure 2.9 shows how changing an observer's SNR threshold varies the targets that get scheduled.

In figure 2.9, the upper right corners in each figure is a magnified view of the observer's field of view (FOV) at the given time. The white spheres in the FOV box are a close up view of the scheduled targets. These are the same as the white spheres represented in the GEO-belt. The red spheres in the belt around the Earth (in the GEO-belt) are the remaining targets in the search space. These targets are being observed by a GEO observer, which is represented by the green sphere.

As the distance between an observer and target decreases, the SNR value between the two increases. As the distance between an observer and target increases, the SNR value between the two decreases. As we increase the SNR minimum threshold, we are able to see fewer targets. On the left figure below, the observable scheduled targets lie closer to the observer, due to their higher SNR values. As we decrease the SNR minimum threshold, we are able to see more targets. On

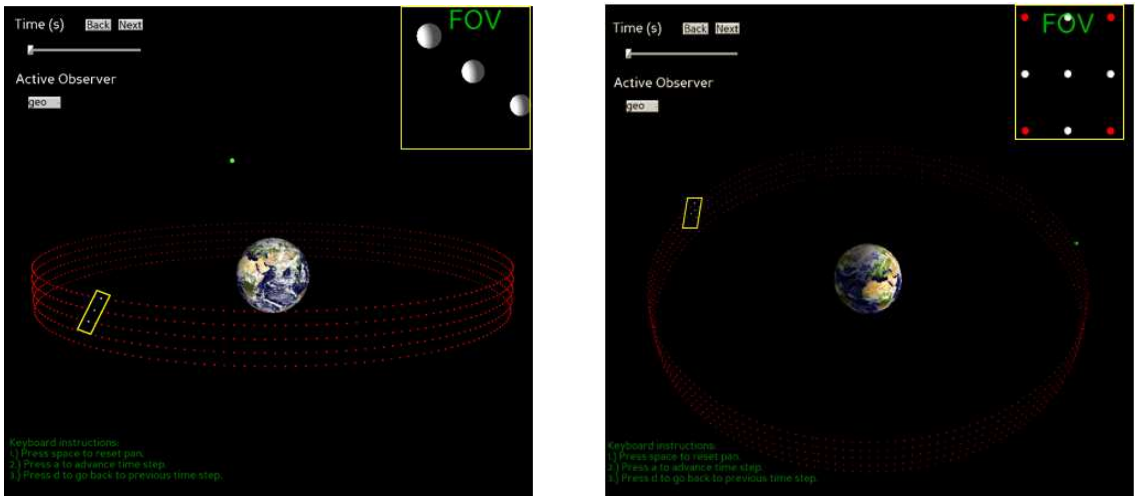

Figure 2.9: Left: SNR threshold of 20, Right: SNR threshold of 5 [6]

the right figure below, the sensor is able to see more targets. The observable targets can be further away from the observer since the SNR minimum threshold decreased.

If a target lies within the Earth's umbra, $[\vec{u}_{start}, \vec{u}_{end}] \in \mathbb{R}^{3\times 2}$, it will not be seen. Similarly, if the Earth lies between a target and an observer, the observer will not see that target at that time.

The gimbal of each observer has a unique moving speed, and thus a cost to move it. The slew time constraint only allows an observer to slew to a given position if it can move there within the time window. Additionally, each observer has a unique image storing capacity. If an observer's storage is full, it cannot schedule tasks to be collected.

Each observer has a score threshold such that a task will not be scheduled if its score is below the score threshold. This is formulated by the following:

$$\left[\sum_{n=0}^{N_{ts}-1} p_{ns}\mu_n\right] + p_s + i_s > h_s, \tag{2.2}$$

where, $N_{ts} \in \mathbb{N}$ is the number of available targets for observer s at time t. Additionally, p_{ns} is the probability that observer s will detect target n, μ_n is the priority of visiting target n, p_s is the cost to point observer s, i_s is the cost of storing an image in observer s, and h_s is the score threshold for observer s. Equation (2.2) is calculated for each observer s and each time window $w(t)$. A time window $w(t)$ is a time with an index t that determines which time window is being observed.

Ideally, revisit times are minimized as well as observer costs. As the score threshold increases, so does the max revisit time. Additionally, as the score threshold increases, the observer utilization and max score decrease. The max score is calculated as the sum of the scores of all scheduled tasks. With a higher score threshold, fewer tasks are scheduled and hence the total max score decreases. These results are seen figure 2.10.

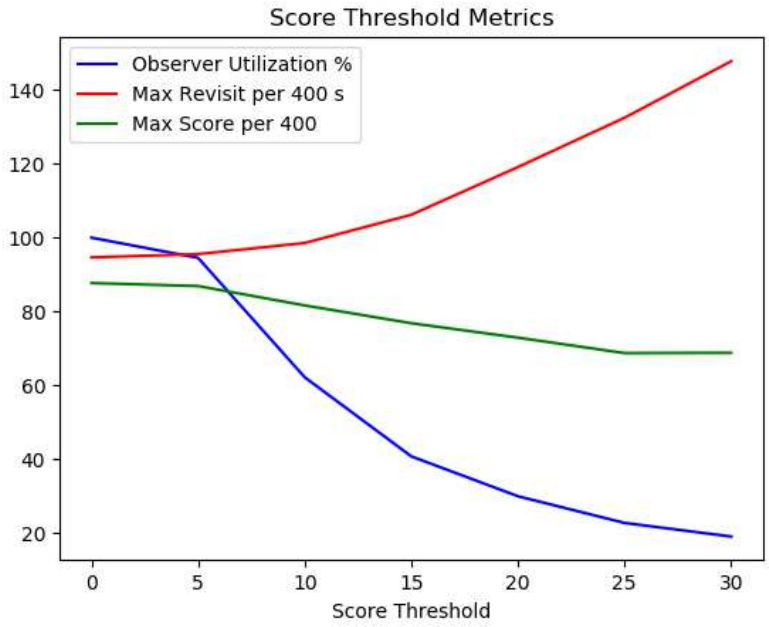

Figure 2.10: Score Threshold

2.2.3 Mathematical Formulation

Recall, the main goal of this SDA resource allocation problem is to generate an optimal schedule that maximizes the number of targets seen while minimizing resource costs and target revisit time. A schedule consists of tasks for observers over some time interval. This time interval is discretized into time windows. Tasks are then assigned to each observer for each time window. Assigned tasks include slew and/or collect commands.

In order to create an optimal schedule, each task is assigned a score. Factors in this score include a target's priority of visitation, a target's probability of detection, and observer costs. Additionally, there are constraints that must be considered for a target to be tasked.

We will define the following notations:

	Definitions
a_s	number of images currently stored for observer s
b_s	buffer size for observer s
d_s	score threshold for observer s
e_{pos}	earth position
F	set of all feasible schedules
fov_s	field of view of observer s
g_s	cost of minimum gimbal movement for observer s
h_s	threshold snr for a target to be seen by observer s
i_s	cost of storing images for observer s
l_s	slew time
M	number of all possible schedule combinations
M_{st}	number of possible looking directions for observer s at time $w(t)$
N_{ts}	number of available targets for observer s at time $w(t)$
p_s	pointing cost for observer s
p_{ns}	probability of observer s detecting target n
r_{ns}	SNR of target n received by observer s
S	total number of observers
T	total number of time windows
t_n	time target n was last visited
t_{size}	time window size
tle_s	two-line element set for observer s
$\vec{u}_{end}$	end of earth's umbra
$\vec{u}_{start}$	start of earth's umbra
$w(t)$	time window indexed at t
X^*	optimal schedule
X_k	kth schedule
$\vec{x}_n$	position of object n
$\vec{x}_{stk}$	kth schedule looking position of sensor s at time $w(t)$
$\vec{y}_s$	current looking position of observer s
μ_n	priority of visiting target n

The set of feasible solutions is a set consisting of all possible schedule combinations. Each possible schedule combination is one in which all constraints are satisfied, and each observer is tasked for each time window. Note: an observer task does not have to include collecting targets. There may be a case where the negative cost to the observer is more than the positive cost of collecting those targets, resulting in a negative task score.

A schedule is a set of looking directions for each observer for each time. A looking direction for observer s and time t is defined by the vector $\vec{x}_{st} \in \mathbb{R}^3$. This is generalized so that any coordinate system may be used. The looking directions are enumerated for each schedule. The kth schedule, denoted as X_k, is defined below:

$$X_k = \{\vec{x}_{stk} : \forall\ s \in [0, S-1]\,, \forall t \in [0, T-1]\}. \tag{2.3}$$

The set of all feasible solutions is the set of all possible schedules. This is denoted as F. These are defined mathematically below:

$$F = \{X_0, X_1, .., X_{M-1}\}, \tag{2.4}$$

where M is the total number of possible schedules.

An optimal schedule is found by solving the following:

$$X^* = max_{X_k \in F} \sum_{t=0}^{T-1} \sum_{s=0}^{S-1} \left[\sum_{n=0}^{N_{ts}-1} p_{ns}\left(w(t), h_s, tle_s, \vec{x}_n\right) \mu_n\left(w(t)\right) \right] + p_s\left(g_s, \vec{x}_{stk}, \vec{y_s}\right) + i_s\left(a_s, b_s\right) \tag{2.5}$$

$$s.t.\ \arccos\left(\frac{\vec{x}_n \cdot \vec{x}_{stk}}{||\vec{x}_n||\,||\vec{x}_{stk}||}\right) \leq FOV_s/2 \quad (field\ of\ view) \tag{2.6}$$

$$h_s \leq r_n \quad (SNR) \tag{2.7}$$

$$\vec{x}_n \notin [\vec{u}_{start}, \vec{u}_{end}] \quad (umbra) \tag{2.8}$$

$$\vec{e}_{pos} \neq (1-\lambda)\,\vec{s}_{pos} + \lambda \vec{x}_n,\ \forall \lambda \in [0,1] \quad (Earth\ blocking) \tag{2.9}$$

$$l_s(\vec{x}_{stk}, \vec{y}_s) < t_{size} \quad (slew\ time) \tag{2.10}$$

$$a_s < b_s \quad (buffer\ space) \tag{2.11}$$

$$\left[\sum_{n=0}^{N-1} p_{ns}\mu_n\right] + p_s + n_s \geq d_s. \quad (score\ threshold) \tag{2.12}$$

2.2.4 Dimensionality

This scheduling problem is highly dimensional. For instance, suppose we only considered a single ground-based sensor observing GEO-based targets. This problem is fairly simple since neither the sensor or targets move with respect to the Earth. Thus, the sensor could easily just scan its field of regard in an ordered manor.

Now suppose we consider a single sensor in orbit observing the same targets. This adds some complexity to the problem since the targets and sensor may be moving at different speeds. Thus, there needs to be more thought given to the order in which the sensor scans its field of regard. Perhaps there is a target within its field of regard at time t_1, but not at time t_2. In that case, the sensor may want to ensure it sees that target at t_1 and thus steers its gimbal intentionally.

To further complicate this problem, suppose we have multiple sensors in the same orbit. Now target n may be seen by sensor s_1 at time t_1 but not at time t_2. However, sensor s_2 is able to see this target at both times t_1 and t_2. In this scenario, it may be better for sensor s_1 to look at targets other than target n and let sensor s_2 look at target n.

Our problem is even more complex than this. We consider multiple sensors in different orbits. Each sensor has a different cost associated with performing the same task, thus adding additional complexity.

To see this complexity mathematically, let M_{st} = the number of possible looking positions for observer s at time t. The total number of possible schedules, call this M, is calculated by the following:

$$M = \prod_{s=0}^{S-1} \prod_{t=0}^{T-1} M_{st}, \tag{2.13}$$

since for each observer s there are $\prod_{t=0}^{T-1} M_{st}$ possible schedules. This is multiplied by all observers' possible schedules to get all possible total schedules of all observers.

For demonstration, suppose there are 3 observers, 300 time windows, and 2 possible looking positions for each observer at each time. This results in $(2^{300})^3 \simeq 8.452712 \times 10^{270}$ total possible

schedules. Even if it only took 0.000001 seconds to generate a single schedule, it would take approximately 2.678503×10^{257} years to generate all possible schedules in this simple scenario.

In one of the simplest scenarios represented in the results to follow, there are 3 observers, 1440 time windows, and approximately 3 looking positions for each observer at each time. For this example, M is too large for a standard computer to handle.

Consequently, there is no tractable way to brute-force compute the global optimum from (2.5). Thus, the following algorithms are implemented in order to solve this: greedy, genetic, random search, weapon-target-assignment, and block greedy. In all these algorithms, the constraints remain the same as in (2.6) - (2.12).

2.2.5 Common Algorithms for SDA Sensor Tasking

The **random search algorithm** is the simplest of the algorithms implemented. In the random search algorithm, a random schedule is generated. Its score is then calculated. If this score is greater than or equal to a preset desired score, that schedule is chosen. If the score is lower than the present desired score, a new schedule is generated. This process continues until a schedule is generated with a high enough score. This algorithm is fast when the schedule score is set low.

The local **greedy algorithm** schedules the maximum task at each time window for each observer [17] [18] [34]. It solves the following modified version of (2.5):

$$X^*(s,t) = max_{X_k(s,t) \in X} \left[\sum_{n=0}^{N-1} p_{ns} \mu_n \right] + p_s + a_s, \tag{2.14}$$

for all $s \in [0, S-1], t \in [0, T-1]$, where $X_k(s,\ t)\ =\ \vec{x}_{stk}$.

With this greedy algorithm, X^* is a local optimum. Due to its simplicity, this algorithm runs very quickly. The sacrifice for this quick run time is that this algorithm only guarantees a local optimal solution.

The goal of the **weapon-target-assignment algorithm** (WTA) is to maximize damage by weapons to targets [18]. The dynamic problem determines which targets survived after previous

rounds. It then assigns and fires a subset of the remaining weapons with an objective of minimizing the total expected value of surviving targets at end of final stage.

For the SDA tasking problem, the objects to be observed are analogous to the targets and the observers' looking positions available are analogous to the weapons. Only one "weapon" is deployed at a time. Multiple "targets" may be engaged by each "weapon". This algorithm breaks the time interval into groups of time windows. It finds all possible combinations of schedules for each time group and chooses the highest scoring schedule. This is formulated below:

$$X^*(t,c) = max_{X_k(t,c)\in X} \sum_{t=0}^{t+c-1} \sum_{s=0}^{S-1} \left[\sum_{n=0}^{N-1} p_{ns}\mu_n \right] + p_s + a_s, \tag{2.15}$$

for all $s \in [0, S-1]$, $t \in [0, T-1]$, and $c =$ number of time windows of observation.

Here, $X_k(t,c) = \{\vec{x}_{st_0k} : \forall t_0 \in [t, t+c-1], \forall s \in [0, S-1]\}$. As c approaches the total number of time windows, X^* approaches the global optimum. If c is less than the total number of time windows, X^* is a regional optimum. Due to its complexity, this is the slowest running algorithm of the five implemented algorithms.

Genetic algorithms are commonly used in high dimensional problems [35]. Because of this, it has been used in the SDA sensor tasking problem [19] [20]. Although genetic algorithms have been shown to be effective for these types of problems, there is not precise way to determine how close to the optimal solution this algorithm gets.

The genetic algorithm uses the following processes: initialize, crossover, and mutate population. A member of a population consists of a possible schedule. In an example with 3 observers and 2 time windows (in the actual simulation there are hundreds of time windows), a schedule consists of the following:

Figure 2.11: kth Schedule

An initial population of size N is randomly created. From the initial population, random member pairs are chosen to crossover. Often these member pairs will be chosen with a weighted factor in order to choose higher scoring members.

The crossover step is completed by randomly selecting pieces of random length of the scheduled pairs. This is done until a new population of size N is created. This process is shown by the following, with random member pairs population A and population B creating new member population C:

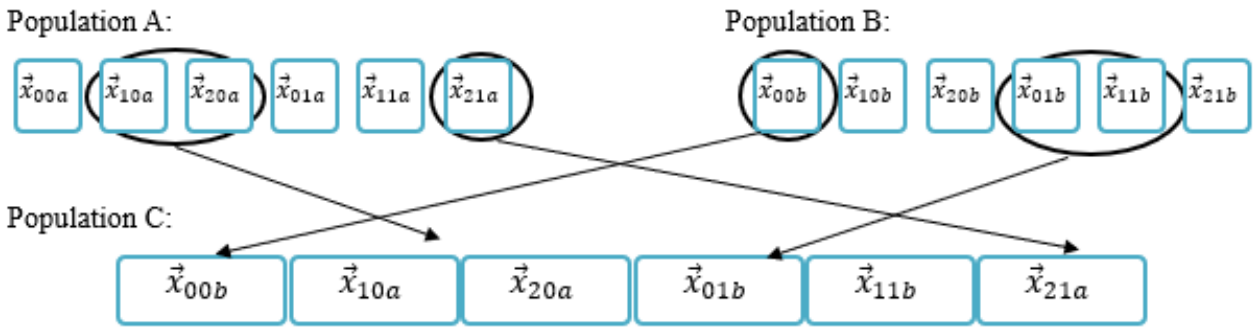

Figure 2.12: Crossover

After this, the new population schedules are mutated. This is done by choosing a random length segment of the schedule and exchanging it with a random length of another schedule. Mutation is shown by the following:

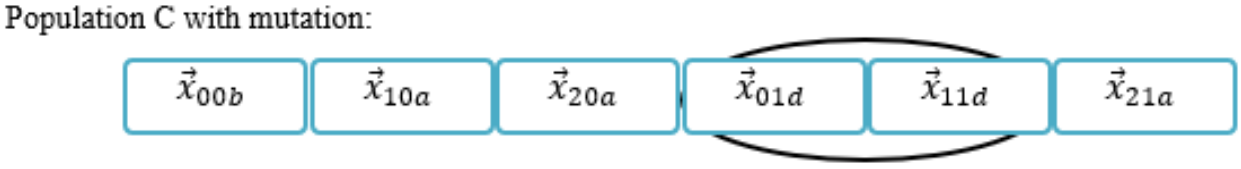

Figure 2.13: Mutation

This entire process is repeated for M generations. After these M generations, the highest scoring schedule is chosen as the approximate optimal schedule.

The genetic algorithm is not the slowest or fastest algorithm, nor does it generate the lowest or highest scoring schedules. This algorithm is a common approach in solving the SDA sensor tasking problem. Hence, it is a valuable metric for comparison.

2.3 Block Greedy Algorithm

Finally, the novel **block greedy algorithm** is a hybrid of the WTA and greedy algorithms, with some additional features. This new algorithm produces an approximately regional optimal schedule and runs in a tractable amount of time. The block greedy algorithm has not before been proposed or implemented in work on the SDA sensor tasking problem.

Like the WTA algorithm, the block greedy algorithm considers multiple time windows at once. This group of time windows is called a "time block", hence the name "block greedy". Instead of computing the score of every possible schedule combination over this time block as in the WTA, the score for each task at each time window is computed. This is like the local greedy algorithm. However, in the local greedy algorithm, tasks are considered in temporal order. In the block greedy algorithm, tasks are considered without regard to temporal order.

In addition to the mentioned similarities to the greedy and WTA algorithms, the block greedy algorithm has added features. After each task is scheduled, all task scores are reset. This is a key component of the block greedy algorithm. This score reset allows the highest priority tasks to be scheduled first and incrementally schedules lower priority tasks. These lower priority tasks become higher priority tasks as more time passes before they are scheduled. Ultimately, this results in a higher scoring schedule.

To create a schedule using the block greedy algorithm, the score for each task in each time window is calculated for all time windows in each time block. The highest scoring task is scheduled. After this, the priority of visitation is reset for all visited targets. The scores for the tasks across

the time block are recalculated and the new highest scoring task is scheduled. This continues until all observers have been tasked for each time window in the time block.

Figure 2.14: Block Greedy Algorithm

This is formulated mathematically below:

$$X^*(s,[t,t+c-1]) = max_{X_k(s,[t,t+c-1]))\in X}\left[\sum_{n=0}^{N-1} p_{ns}\mu_n\right] + p_s + a_s, \tag{2.16}$$

where, $X_k(s,[t,t+c-1]) = \{\vec{x}_{st_0k} : \forall t_0 \in [t,t+c-1], \forall s \in 0, S-1\}$ and $c =$ number of time windows in time block.

The block greedy algorithm is not much slower than the local greedy algorithm and much faster than the WTA. Additionally, the block greedy algorithm generates a higher schedule score than all other implemented algorithms.

2.3.1 Results

In the following simulations, our search space consists of volumes in the GEO-belt. We voxelize this volume and assume the center of each voxel to be a target. In order to ensure that entire volume is viewed, each voxel in the volume is generated to be at least as small as the minimum field of view of the observers.

We use the Python function that uses the Simplified General Perturbations (SGP4) propagator [33] to calculate observer propagation. This function accounts for real-life effects of perturbation on object propagation. Each of our observers has a TLE associated with it. This TLE is used as an input to the SGP4 function to calculate its propagation in time. We will assume this function outputs an observer's exact position.

In a real-life scenario, targets do not lie evenly on a voxelized grid as our simulated targets do. However, since we assume we know the exact positions of our targets, our algorithms run the same as they would if the target positions were randomly distributed across the search volume. Hence, our targets are generated in this way for simplicity.

Recall, equation (2.5) considers multiple, heterogeneous sensors. Each sensor can have different costs associated with tasking it. Thus, a higher schedule score may be generated by unevenly tasking sensors. Below is an example of the observer utilization of three heterogeneous sensors in a 24-hour schedule:

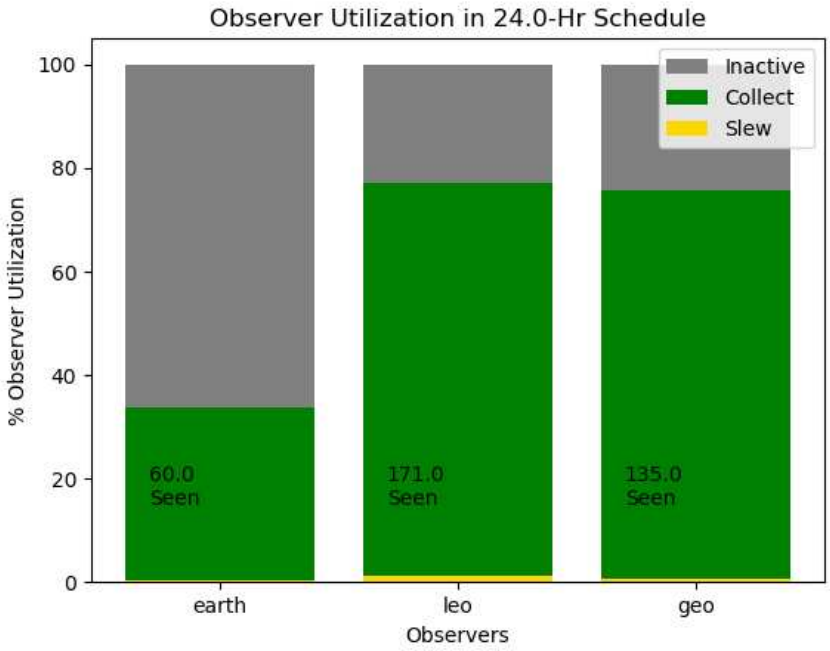

Figure 2.15: Observer Utilization Percent

In this example, the Earth-based sensor has the lowest reward to cost ratio to task, while the LEO-based sensor has the highest. This means the difference between the positive score and negative resource costs are the most positive in the LEO-based sensor case.

The block greedy algorithm was presented as a new algorithm to solve equation (2.5). This is compared with the traditional greedy algorithm, the WTA algorithm, the genetic algorithm, and a random search algorithm. The block greedy algorithm is shown to overall outperform all other compared methods. It runs faster than almost all the algorithms and generates the highest score.

Figure 2.16 is a chart of total schedule scores per algorithm. A schedule score is computed by adding all task scores in a schedule. The figure below was generated with 970 synthesized targets, 3 observers, over 24-hours with 60 second time windows. The block greedy algorithm had 30 time window blocks. Due to the incredibly slow run time of the WTA algorithm, even at 2 time window blocks, this was run with a single time window block. As expected, the block greedy algorithm generates the highest scoring schedule, while the random search algorithm generates the lowest.

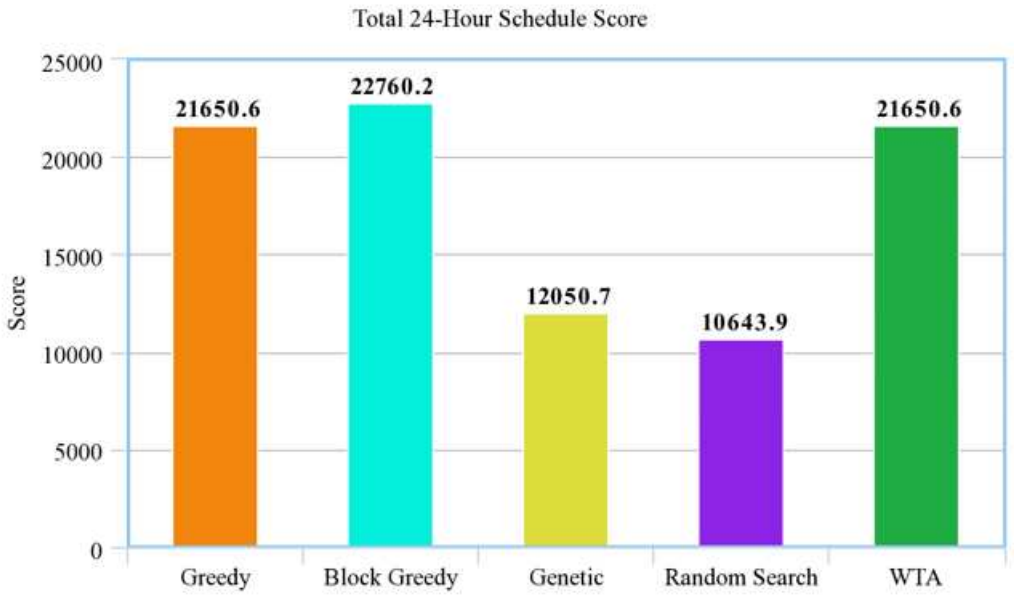

Figure 2.16: Total Schedule Score by Algorithm

Although this schedule was run for a 24-hour interval, most algorithms were able to see all targets after a much shorter time. Figure 2.17 shows the amount of time each algorithm took to see all objects. Additionally, a lower max revisit time is preferable. This is analyzed below as well. In the following results, the WTA algorithm was run with a 2 time window block and there were 366 synthesized targets.

Algorithm	Time to See All Targets	Max Revisit Time
Greedy	6 hours, 48 min	10 hours, 31 min
Block Greedy	6 hours, 59 min	10 hours, 53 min
Weapon-Target-Assignment	6 hours, 48 min	10 hours, 36 min
Random Search	None	None
Genetic	16 hours, 42 min	None

Figure 2.17: Time to See All Targets and Max Revisit Times Per Algorithm

The greedy and WTA algorithms were able to see all targets in the shortest amount of time. They also had the two shortest max revisit times. However, the block greedy algorithm was able to see all objects in just 2.7% more time. Additionally, the block greedy algorithm had only 3.5% slower max revisit time than the greedy algorithm. The max revisit time was only calculated once all targets were visited twice. Hence, the random search and genetic algorithms were not able to see all targets twice in this time interval.

In terms of run time, the random search algorithm was the fastest. Given the results above, the random search algorithm performed the worst outside of this factor. The next fastest algorithm was the local greedy algorithm. The slowest algorithm was the WTA, as expected. The block greedy runs only 3 times slower than the local greedy algorithm nearly 22 times faster than the WTA algorithm.

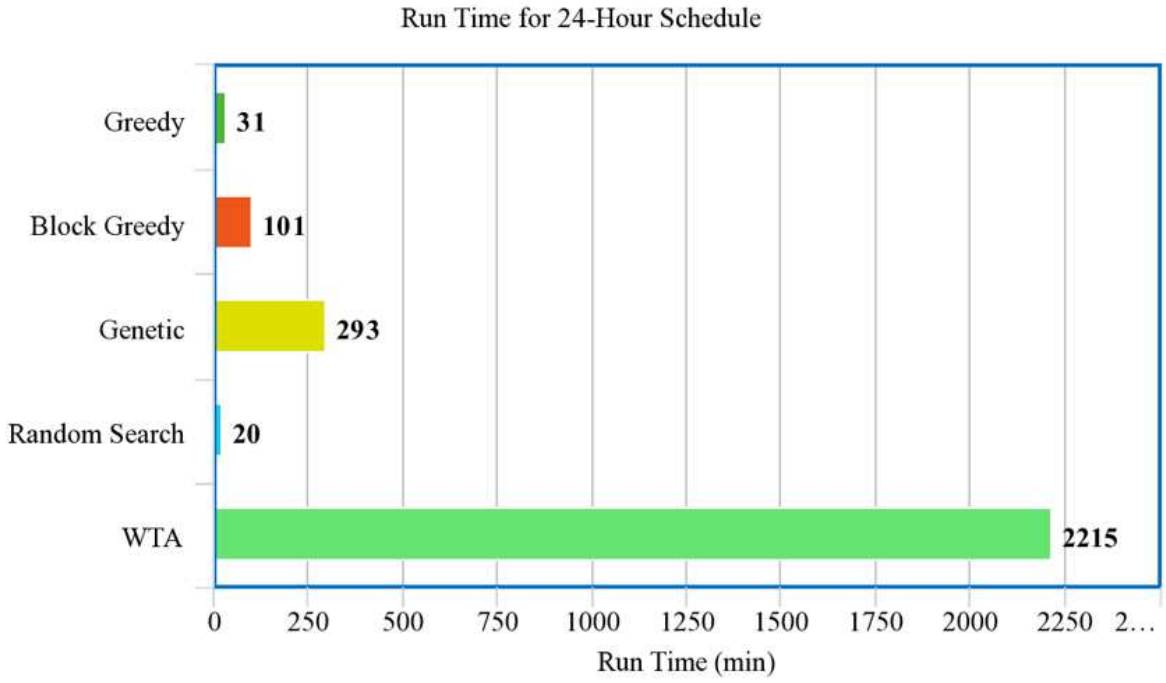

Figure 2.18: Run Time Per Algorithm

Altogether, the block greedy has a much higher max score than all other algorithms compared. It successfully reduces resource costs efficiently. It runs only 3 times slower than the local greedy algorithm for 30 time windows in a block. What is sacrificed in run time using the block greedy algorithm is gained in its higher schedule score.

2.3.2 Conclusion

A robust space domain awareness scheduler has been created with realistic sensor tasking scenarios. This scheduler functions with real-life constraints such as only considering targets within observers' field of view, outside of the Earth's umbra, not blocked by the Earth, and with high enough SNR. This scheduler is capable of handling multiple, heterogeneous observers at once and creating an approximate optimal schedule for all observers.

This scheduler considers the unique tasking cost of each observer. It computes the costs of both storing images and slewing for each observer. Not only does this scheduler maximize the total number of targets seen, it also minimizes the total cost of tasking the available observers.

Five algorithms have been implemented in the SDA scheduler. Four of these algorithms are common approaches used in solving the SDA observer tasking problem. These algorithms generate a worse schedule than the block greedy algorithm.

The block greedy algorithm borrows speed from the greedy algorithm and optimality from the WTA. When analyzed, it is found that the block greedy outperforms the other implemented algorithms. It consistently produces a higher scoring schedule and runs in a tractable amount of time.

In summary, this work thus not only introduces a new, more realistic objective function formulation for the SDA resource allocation problem, but also creates a better algorithm for tasking resources. This new SDA resource allocator both optimizes the total number of targets to be visited and successfully minimizes resource costs.

Chapter 3

Flight Path Planning for SAR

3.1 Background

3.1.1 Goals

The goal of the SAR Flight Path Planning problem is to maximize image resolution and scene coverage by creating an optimal flight path. This work will be an extension of Dr. Scott Altrichter's PhD thesis titled "Flight Path Optimization for Resolution and Coverage in Synthetic Aperture Radar" [10].

In his thesis, Dr. Altrichter formulates an objective function for the purpose of optimizing image resolution and scene coverage by varying an aircraft's flight path. His objective function is derived from the 3-dimensional data-collection manifold. He describes a single, mono-static aircraft SAR scenario.

In this thesis, we will first modify this objective function to use only the first two components of the data-collection manifold since these are the primary components with which we are concerned. We will compare the results of this objective function with common SAR modes including stripmap, spotlight, and circular SAR.

Additionally, in Dr. Altrichter's thesis, the flight path is only optimized against the aircraft's heading and pitch. We will extend this work by also optimizing against the aircraft's antenna steering angles.

In the next chapter, we will show the flexibility of this objective function by adding many realistic features. These include an SNR consideration and a priority of targets or targets scenes. Additionally, we will extend the formulated objective function to flight path planning of mono-static, multiple unmanned aerial vehicles (UAVs). Lastly, we will implement additional optimization algorithms with the objective function and compare their performance.

First, the following assumptions will be made in solving this problem:

1. We assume we know the locations of the targets to be observed.

2. We assume the target scene is flat. Thus, targets lie on the $xy-$plane.

3. For simplicity, we assume the antenna is rectangular planar.

4. Lastly, we assume the antenna footprint is an ellipse.

3.1.2 Synthetic Aperture Radar Derivation

In a standard SAR system, an antenna is mounted on an aircraft or satellite. This SAR system transmits radio waves to a scene. These radio waves are reflected by the scene and the scattered waves are received, usually on the same antenna. From this, mathematical procedures are used to recover the scene and form an image [7].

Synthetic Aperture Radar deals with microwaves which are a type of electromagnetic wave. Hence, the waves used in SAR are governed by Maxwell's equations (for a derivation of Maxwell's equations, see Appendix B).

Let $\boldsymbol{\mathcal{E}}(t, \boldsymbol{x})$ be the electric field, $\boldsymbol{\mathcal{B}}(t, \boldsymbol{x})$ be the magnetic induction field, $\boldsymbol{\mathcal{D}}(t, \boldsymbol{x})$ be the electric displacement field, $\boldsymbol{\mathcal{H}}(t, \boldsymbol{x})$ be the magnetic intensity field, $\rho(t, \boldsymbol{x})$ be the charge density, and $\boldsymbol{\mathcal{J}}(t, \boldsymbol{x})$ be the current density. Then Maxwell's equations in the time domain are defined below:

$$\nabla \times \boldsymbol{\mathcal{E}} = -\frac{\partial \boldsymbol{\mathcal{B}}}{\partial t} \tag{3.1}$$

$$\nabla \times \boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{J}} + \frac{\partial \boldsymbol{\mathcal{D}}}{\partial t} \tag{3.2}$$

$$\nabla \cdot \boldsymbol{\mathcal{D}} = \rho \tag{3.3}$$

$$\nabla \cdot \boldsymbol{\mathcal{B}} = 0 \tag{3.4}$$

From Maxwell's equations, we will derive the inhomogeneous wave equation for the electric field $\mathbf{E}$. First, we will note that, in free space, $\rho(t, \boldsymbol{x}) = 0$. We will also note the following constitutive relations [7]:

$$\boldsymbol{\mathcal{D}} = \epsilon_0 \boldsymbol{\mathcal{E}} \tag{3.5}$$

$$\boldsymbol{\mathcal{B}} = \mu_0 \boldsymbol{\mathcal{H}} \tag{3.6}$$

Additionally, in free space, $\nabla \cdot \boldsymbol{\mathcal{D}} = 0$. Thus, $\epsilon_0 \nabla \cdot \boldsymbol{\mathcal{E}} = 0$. This will be useful later.

Now, we will take the curl of (3.1). This leads to the following:

$$\begin{aligned}
\nabla \times \nabla \times \boldsymbol{\mathcal{E}} &= \nabla \times \left(-\frac{\partial \boldsymbol{\mathcal{B}}}{\partial t} \right) \\
\nabla(\nabla \cdot \boldsymbol{\mathcal{E}}) - \nabla^2 \boldsymbol{\mathcal{E}} &= \nabla \times \left(-\frac{\partial \mu_0 \boldsymbol{\mathcal{H}}}{\partial t} \right) \\
-\nabla^2 \boldsymbol{\mathcal{E}} &= -\mu_0 \nabla \times \left(\frac{\partial \boldsymbol{\mathcal{H}}}{\partial t} \right) \\
\nabla^2 \boldsymbol{\mathcal{E}} &= \mu_0 \left(\frac{\partial (\nabla \times \boldsymbol{\mathcal{H}})}{\partial t} \right) \\
&= \mu_0 \left(\frac{\partial}{\partial t} \left(\boldsymbol{\mathcal{J}} + \frac{\partial \boldsymbol{\mathcal{D}}}{\partial t} \right) \right) \\
&= \mu_0 \epsilon_0 \frac{\partial^2 \boldsymbol{\mathcal{E}}}{\partial t^2} + \mu_0 \frac{\partial \boldsymbol{\mathcal{J}}}{\partial t}
\end{aligned} \tag{3.7}$$

$$\implies (\nabla^2 - \mu_0 \epsilon_0 \frac{\partial^2}{\partial t}) \boldsymbol{\mathcal{E}}(t, \boldsymbol{x}) = \mu_0 \frac{\partial \boldsymbol{\mathcal{J}}}{\partial t}. \tag{3.8}$$

The term $\mu_0 \frac{\partial \boldsymbol{\mathcal{J}}}{\partial t}$ gets quite complicated when predicting what $\boldsymbol{\mathcal{J}}$ arises on an object from an impinging wave. From here on out, we will denote this term as $s(t, \boldsymbol{x})$. This is a source term that involves the electric field and the magnetic field [36]. Now we have the following inhomogeneous wave equation in the time domain:

$$(\nabla^2 - \mu_0 \epsilon_0 \frac{\partial^2}{\partial t}) \boldsymbol{\mathcal{E}}(t, \boldsymbol{x}) = \mathbf{s}(t, \boldsymbol{x}). \tag{3.9}$$

It will be useful to work in the frequency domain for the remainder of this derivation. In order to transform equation (3.9) from the time domain to the frequency domain, we will use the inverse Fourier Transform (IFT). The forward and inverse Fourier Transforms are defined below, respectively:

$$f(t) = \mathcal{F}\{F(\omega)\} = \frac{1}{2\pi} \int e^{-i\omega t} F(\omega) d\omega \tag{3.10}$$

$$F(\omega) = \mathcal{F}^{-1}\{f(t)\} = \int e^{i\omega t} f(t) dt \tag{3.11}$$

Equation (3.9) in the time domain is the inhomogeneous Helmholtz equation for the electric field $\mathbf{E}$. This is defined as the following:

$$(\nabla^2 + k^2)\mathbf{E}(k, \boldsymbol{x}) = \boldsymbol{S}, \tag{3.12}$$

where $\boldsymbol{S}$ is the inverse Fourier Transform of s, c is the wave propagation speed in a vacuum, and $k = \omega/c$ is the wavenumber. In the following, we only consider one component E of the electric fields.

The outgoing Green's function is important in solving the Helmholtz equation with an arbitrary source term. First, define the Green's function as the following:

$$G(\omega, \mathbf{x}) = \frac{e^{ik|\mathbf{x}|}}{4\pi|\mathbf{x}|}; \qquad \text{it satisfies} \qquad (\nabla^2 + k^2)G = -\delta. \tag{3.13}$$

The Helmholtz equation with an arbitrary source term is represented as

$$(\nabla^2 + k^2)E(\mathbf{x}) = S(\mathbf{x}). \tag{3.14}$$

Thus, the Green's function allows us to obtain the following outgoing solution:

$$E(\mathbf{x}) = -\int G(\mathbf{x} - \mathbf{y})S(\mathbf{y})d\mathbf{y}. \tag{3.15}$$

Now S can be broken up into two pieces. Define S^{in} as proportional to the current density on the antenna, while S^{sc} denotes the piece that models the scattering object. Thus, $S = S^{in} + S^{sc}$. Consequently, $E^{tot} = E^{in} + E^{sc}$. Here, E^{sc} denotes the scattered field, which is due to scattering from the object. Additionally, E^{in} is the incident field. This satisfies the wave equation for a prescribed S^{in}.

The target's response to an incidence field is represented as S^{sc}. Using the scalar model for S^{sc}, we have

$$S^{\mathrm{sc}}(\mathbf{x}) = -V(\mathbf{x})E^{\mathrm{tot}}(\mathbf{x}). \tag{3.16}$$

The reflectivity function is denoted as V. The reflectivity function represents the scene we wish to recover. Substituting (3.16) into (3.15), we obtain the Lippmann-Schwinger integral:

$$E^{\mathrm{sc}}(\mathbf{x}) = \int G(\mathbf{x}-\mathbf{y})V(\mathbf{y})E^{\mathrm{tot}}(\mathbf{y})d\mathbf{y}. \tag{3.17}$$

Finally, we will use the Born approximation. The Born approximation is a useful tool in imaging problems because it makes the problem linear. The Born approximation is found by replacing E^{tot} in the Lippmann-Schwinger integral with E^{in}. As a result, the Born approximated scattered field is

$$E_B^{sc}(\mathbf{x}) = \int G(\mathbf{x}-\mathbf{y})V(\mathbf{y})E^{in}(\mathbf{y})d\mathbf{y} = \int \frac{e^{ik|\mathbf{x}-\mathbf{y}|}}{4\pi|\mathbf{x}-\mathbf{y}|}V(\mathbf{y})\frac{e^{ik|\mathbf{y}-\mathbf{x^0}|}}{4\pi|\mathbf{y}-\mathbf{x^0}|}\widehat{\mathscr{F}(\omega, \mathbf{x}-\mathbf{x^0})}d\mathbf{y} \tag{3.18}$$

where $\mathscr{F}$ denotes the Fourier transform of the current density on the antenna.

Assuming the transmitting antenna and the receiving antenna are in the same location, the received signal is then modeled by

$$S_B(\omega) = \int e^{2ik|\mathbf{x^0}-\mathbf{y}|}A(\omega, \mathbf{x}, \mathbf{y})V(\mathbf{y})d\mathbf{y}. \tag{3.19}$$

Here, A includes the geometrical spreading factors, transmitted waveform, and antenna beam patterns.

In SAR, the antenna is moving. Thus, instead of representing the location of the antenna as x^0, we will denote it as $\gamma(s)$, where s represents time. Aircraft motion is on a different time scale from wave propagation. These time scales are generally referred to as slow time and fast time, respectively. Thus, s is the slow time. Then the received signal becomes:

$$D(\omega, s) = f[V](\omega, s) := \int e^{2ik|\gamma(\mathbf{s})-\mathbf{y}|} A(\omega, s, \mathbf{y}) V(\mathbf{y}) d\mathbf{y} \tag{3.20}$$

where $f[V](\omega, s)$ denotes the forward map from V to the data.

In order to recover the image, V, we will use the following imaging operator [7], where Q is a filter to be determined:

$$I(\boldsymbol{y}) = B[D](\boldsymbol{y}) := \int e^{-2ik|\gamma(s)-\boldsymbol{y}|} Q(\omega, s, \boldsymbol{y}) D(\omega, s) d\omega ds. \tag{3.21}$$

Now we will analyze the image to determine its quality. In this process, we will choose Q. Equation (3.21) describes how we process data. By substituting (3.20) into (3.21), we get the following:

$$I(\boldsymbol{y}) = \int \int e^{-2ik(|\boldsymbol{R}_{s,\boldsymbol{y}}|-|\boldsymbol{R}_{s,\boldsymbol{x}}|)} Q(\omega, s, \boldsymbol{y}) A(\omega, s, \boldsymbol{x}) d\omega ds V(\boldsymbol{x}) d\boldsymbol{x}, \tag{3.22}$$

where $\boldsymbol{R}_{s,\boldsymbol{z}} = \gamma(s) - \boldsymbol{y}$. In order to create our image analysis, we will define the following:

$$K(\boldsymbol{z}, \boldsymbol{x}) := \int e^{-2ik(|\boldsymbol{R}_{s,\boldsymbol{y}}|-|\boldsymbol{R}_{s,\boldsymbol{x}}|)} Q(\omega, s, \boldsymbol{y}) A(\omega, s, \boldsymbol{x}) d\omega ds. \tag{3.23}$$

Equation (3.23) is called the Point Spread Function (PSF). The PSF describes how a point in a scene appears in an image. In order to perfectly recover V, the PSF must be equal to the $\delta-$function, below:

$$\delta(\boldsymbol{z} - \boldsymbol{x}) \propto \int e^{i(\boldsymbol{z}-\boldsymbol{x})\cdot\boldsymbol{\xi}} d\boldsymbol{\xi}. \tag{3.24}$$

However, in reality the PSF does not exactly equal the $\delta-$function. The closer the PSF is to the $\delta-$function, the better the resolution. Thus, we will choose a Q that results in a PSF that is as close as possible to the $\delta-$function [7]. The PSF determines image resolution. We show in Appendix C that the PSF can be expressed as an integral of the form eq. (3.24) where the integration is over a set called the data-collection manifold. This set thus determines the resolution.

3.1.3 Types of SAR Systems

Some of the most common SAR systems include *Spotlight*, *Circular*, and *Stripmap*. These each have unique benefits. Depending on the goals and constraints of the SAR mission, it may be better to use one type over the other.

In spotlight SAR, the origin of coordinates is placed at the center of the illuminated scene. The radar is focused on that location throughout the entire flight path. The aircraft flies along a straight path while the antenna is steered to illuminate that specific region in the scene. Because of this feature, spotlight SAR provides a greater scene resolution than stripmap SAR.

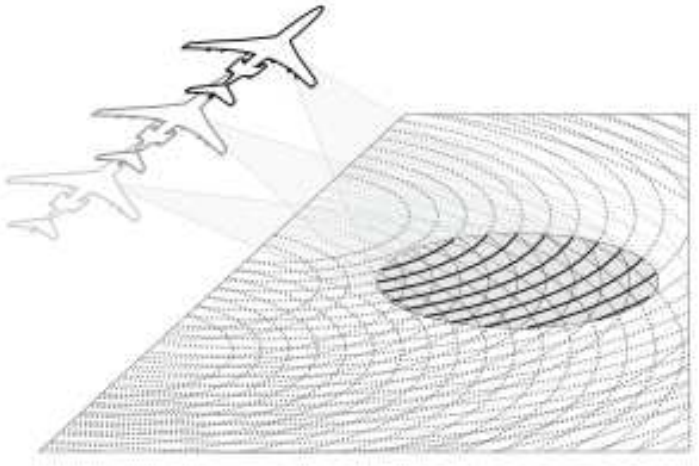

Figure 3.1: Spotlight SAR [7]

In stripmap SAR, the aircraft again flies along a straight path. Here, the antenna viewing direction is fixed relative to the platform. Because of this, stripmap SAR provides better scene coverage than spotlight SAR.

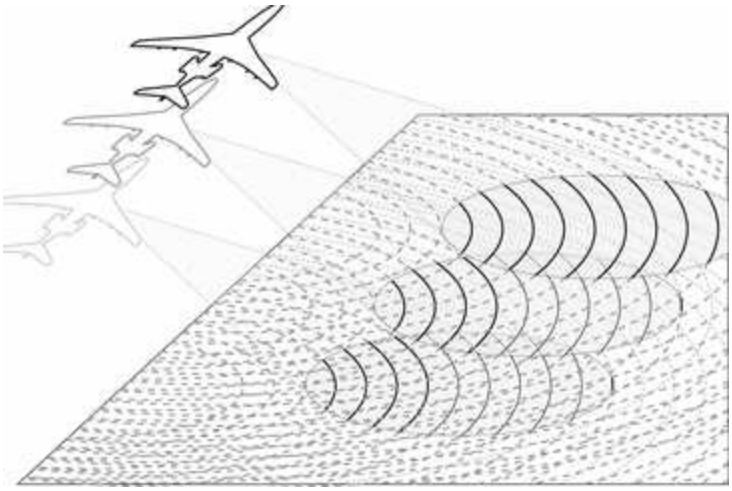

Figure 3.2: Stripmap SAR [8]

An additional SAR mode to consider is circular SAR. In circular SAR, the platform flies in a circular path around the target scene with the antenna steered towards a specific target or part of a target scene. The benefit of this SAR mode is that it can obtain data over a 360 degree azimuthal range [37].

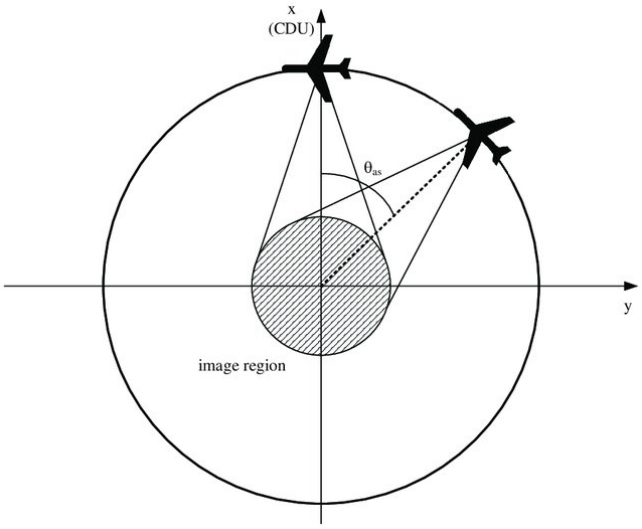

Figure 3.3: Circular SAR path [9]

The main benefit of spotlight SAR and circular SAR is image resolution. The main benefit of stripmap SAR is scene coverage.

3.2 SAR Flight Path Planning

Two key components in Synthetic Aperture Radar (SAR) are image resolution and scene coverage. To improve SAR image resolution, one may increase the bandwidth of the system (frequency diversity) or increase the range of aspect angles to view the target (geometric diversity). In most applications, the radar must operate within a fixed and specified frequency band, so increasing the frequency diversity is not feasible [38].

Thus, to improve target resolution one must use geometric diversity. This exploits the motion of the antenna platform and/or steering of the antenna as a way to increase the range of aspect angles to view the target [39, 40]. The formulas that determine resolution in these dimensions are well-known and widely used [7, 40–46].

To use geometric diversity to maximize scene coverage and image resolution, one may fly around the target scene of interest in a non-straight path and steer the radar antenna towards the desired targets. By using geometric diversity to improve target resolution, one encounters a trade-off with scene coverage and image resolution [43]. This is best shown in the comparison of stripmap SAR, spotlight SAR, and circular SAR.

In stripmap SAR, the platform moves along a straight path with the antenna fixed to broadside. This illuminates a continuous swath on the ground parallel to the flight path. Ultimately, the angular aperture for targets in the swath is limited by the beamwidth of the physical antenna [46]. In circular SAR, the platform flies in a circular path with the antenna steered toward a particular area of the target scene. In spotlight SAR, the platform flies along a linear path and the antenna is steered toward a particular area of the target scene. In both circular and spotlight SAR, angular aperture is increased for the illuminated area [44, 46, 47].

There are many other SAR modes such as scan SAR, spotdwell SAR, and arbitrary stripmap SAR, interferometric wide swath (IW), extra wide swath (EW), wave (WV), along with others [48, 49]. However, we will primarily compare our results with the stripmap, spotlight, and circular SAR modes as these are the most similar to our objective.

The majority of this section is devoted to developing a mathematical tool that considers both

target resolution and scene coverage. This tool involves the data-collection manifold (DCM) whose volume corresponds inversely to the image resolution at a point [7]. Adding together these manifold contributions for each observed target gives us an objective function that is determined by resolution and scene coverage. We then use this function as a way to determine optimal flight paths that exploit geometric diversity for desired resolution and scene coverage objectives.

3.2.1 Data-Collection Manifold

In order to construct an objective function to maximize scene coverage and image resolution, we will start with the data-collection manifold. Data is collected from a target when it lies in a sensor's antenna footprint. This collection of data is called the data-collection manifold (DCM). The shape of the data-collection manifold is determined by the frequencies transmitted and the line-of-sight from the antenna to the target point $\boldsymbol{z}$. This is defined below [10]:

$$\Omega_{\mathbf{z}} = \left\{ \xi = \frac{2\omega}{c} \hat{\mathbf{R}}_{s,\mathbf{z}} : \omega \in [\omega_{min}, \omega_{max}], \mathbf{z} \in \text{antenna footprint} \right\}, \tag{3.25}$$

where $\hat{\mathbf{R}}_{s,\mathbf{z}} = \widehat{\gamma(s) - \mathbf{z}}$ is the unit line-of-sight vector from the antenna to the target. The parameters that vary are ω and s. Refer to Appendix C for a derivation of the DCM.

We see that for a target illuminated by the antenna, the DCM associated with that target is determined by the angular aperture and the transmitted bandwidth. The PSF is the Fourier transform of the DCM. Thus, we want the DCM to be large in order to make the support of the main peak of the PSF small. In radar imaging, the resolution of the image is determined by the PSF. For a more detailed proof of this, see [7].

When considering resolution as governed by the point-spread function, we see that in the cross-range direction, resolution is improved with longer observation times. Longer observation times mean that we can observe the target over a larger angular aperture. This is often called the integration angle [46].

Ideally, in stripmap SAR, we would like to have the widest possible antenna beamwidth to increase the length of time that we view a target. This, however, decreases antenna gain and decreases target detection in noise.

The data-collection manifold determines the point-spread function, and we know that the resolution of an image is determined by the point-spread function. In the following sections, we will develop the data-collection manifold as a measure for quantifying attainable resolution. This can then be used as a tool for mission planning and data-collection.

3.2.2 Objective Function

In the previous section, we demonstrated how the data-collection manifold (DCM) determines the point-spread function (PSF) and thus the resolution of a target. The dimensions of the manifold are determined by the bandwidth of the system and the aspect angles at which the sensor views the target [7, 46].

It is our goal to quantify the size of the manifold in some way. This allows us to use that quantity as a measure for the attainable resolution for a target. In this section, we will follow the steps outlined in [10] by developing a technique for calculating the surface area of the data-collection manifold associated with a target $\boldsymbol{z}$. The surface area will then give us a measure for analyzing target resolution information.

A. DCM as a parameterized surface

The process of building the objective function in Sections 3.2.2A and 3.2.2B and the parameter definitions in Section 3.2.2C were developed by Dr. Scott Altrichter [10]. In his work, he derived the objective using the 3D data-collection manifold. These steps below are a variation from his work in that we will only consider the 2D components of the data-collection manifold. For a derivation of the objective function using the 3D components of the data-collection manifold, see Appendix D.

We will begin by defining the DCM as a parametrized space. With the parameter space defined by angular frequency ω and slow-time s, the data-collection manifold associated with that parameter space is a surface given by the following:

$$\boldsymbol{\xi}(\omega, s) = \frac{2\omega}{c}\widehat{\boldsymbol{R}}_{s,\mathbf{z}} \tag{3.26}$$

$$= [\xi_1(\omega, s), \xi_2(\omega, s)], \tag{3.27}$$

where ξ_1 and ξ_2 are the x and y coordinates in frequency space. It should be noted that in most SAR scenarios, we only care about the horizontal image resolution. Thus, we will only consider the ξ_1 and ξ_2 components in $\boldsymbol{\xi}$. If considering the vertical resolution were also important, we would include the ξ_3 component of $\boldsymbol{\xi}$ as in [10]. Below is a figure of this parametrization in all three components of the DCM:

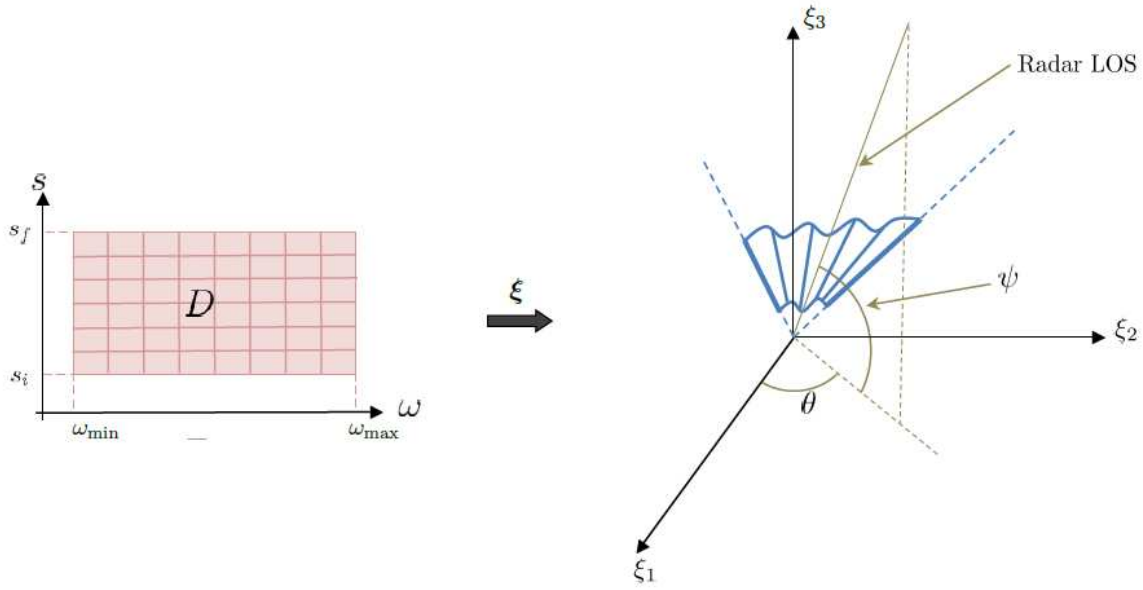

Figure 3.4: Parameter Space and Corresponding DCM [10]

With the target placed at the origin, it is convenient to write these coordinates in terms of azimuthal and elevation angles. These angles potentially change with each transmit time, so we write these as functions of the slow-time variable, $\theta(s)$ and $\psi(s)$, respectively. The parameterized

surface can now be defined as the following:

$$\boldsymbol{\xi}(\omega, s) = \frac{2\omega}{c}[\cos\theta(s)\cos\psi(s), \sin\theta(s)\cos\psi(s)] \tag{3.28}$$

for $\omega \in [\omega_{\text{min}}, \omega_{\text{max}}]$ and $s \in [s_i, s_f]$, where $[\omega_{\text{min}}, \omega_{\text{max}}]$ is the bandwidth and $[s_i, s_f]$ is the observation interval [25]. The angles θ and ψ are defined as the following for a target $\boldsymbol{z}$:

$$\begin{aligned}
\theta_{\boldsymbol{z}} &= \arctan\left(\frac{y - \gamma_2(s)}{x - \gamma_1(s)}\right) \\
\psi_{\boldsymbol{z}} &= \arctan\left(\frac{\gamma_3(s)}{L_{\boldsymbol{z}}}\right) \\
L_{\boldsymbol{z}} &= \sqrt{(x - \gamma_1)^2 + (y - \gamma_2)^2)},
\end{aligned} \tag{3.29}$$

where x and y are the coordinates for target $\boldsymbol{z}$ and the flight path at time s is $\boldsymbol{\gamma}(s)$, where $\boldsymbol{\gamma}(s) = [\gamma_1(s), \gamma_2(s), \gamma_3(s)]$.

B. DCM surface area and volume

Using multivariable calculus tools, we will now formulate an expression for the surface area of the data-collection manifold associated with a point $\boldsymbol{z}$. This surface area will be a tool for measuring the resolution of a single target. In order to also consider scene coverage, we will integrate over all targets, leaving us with a volume for the DCM.

We begin by using (3.28) as the parameterized representation of the DCM. From this, we can calculate the partial derivatives with respect to our two parameters, ω and s. We will denote these partial derivatives as $\boldsymbol{\xi}_\omega$ and $\boldsymbol{\xi}_s$, respectively.

These are the following:

$$\boldsymbol{\xi}_\omega(\omega, s) = \left[\frac{\partial \xi_1}{\partial \omega}, \frac{\partial \xi_2}{\partial \omega}\right] \tag{3.30}$$

$$\boldsymbol{\xi}_s(\omega, s) = \left[\frac{\partial \xi_1}{\partial s}, \frac{\partial \xi_2}{\partial s}\right], \tag{3.31}$$

where

$$\frac{\partial \xi_1}{\partial \omega} = \frac{2}{c}\cos\theta(s)\cos\psi(s) \tag{3.32}$$
$$\frac{\partial \xi_2}{\partial \omega} = \frac{2}{c}\sin\theta(s)\cos\psi(s) \tag{3.33}$$
$$\frac{\partial \xi_1}{\partial s} = \frac{2\omega}{c}(-\theta'(s)\sin\theta(s)\cos\psi(s) - \psi'(s)\cos\theta(s)\sin\psi(s)) \tag{3.34}$$
$$\frac{\partial \xi_2}{\partial s} = \frac{2\omega}{c}(\theta'(s)\cos\theta(s)\cos\psi(s) - \psi'(s)\sin\theta(s)\sin\psi(s)). \tag{3.35}$$

The expression for a surface element of the DCM can be found by computing the magnitude of the cross product of $\boldsymbol{\xi}_\omega$ and $\boldsymbol{\xi}_s$. This is found by computing the following:

$$\begin{aligned} ||\boldsymbol{\xi}_\omega \times \boldsymbol{\xi}_s|| &= \begin{vmatrix} \frac{\partial \xi_1}{\partial \omega} & \frac{\partial \xi_2}{\partial \omega} \\ \frac{\partial \xi_1}{\partial s} & \frac{\partial \xi_2}{\partial s} \end{vmatrix} \\ &= \left(\frac{\partial \xi_1}{\partial \omega}\right)\left(\frac{\partial \xi_2}{\partial s}\right) - \left(\frac{\partial \xi_2}{\partial \omega}\right)\left(\frac{\partial \xi_1}{\partial s}\right) \\ &= \frac{4\omega}{c^2}\left|\frac{\partial \theta}{\partial s}\right|\cos^2\psi(s). \end{aligned} \tag{3.36}$$

With the calculated surface element, we can formulate an expression for the surface area of the DCM at the target location $\boldsymbol{z}$. This is given by the following

$$A(\Omega_{\boldsymbol{z}}) = \int_{s_i}^{s_f}\int_{\omega_{\min}}^{\omega_{\max}} ||\boldsymbol{\xi}_\omega \times \boldsymbol{\xi}_s|| d\omega ds \tag{3.37}$$
$$= \int_{s_i}^{s_f}\int_{\omega_{\min}}^{\omega_{\max}} \frac{4\omega}{c^2}\left|\frac{\partial \theta}{\partial s}\right|\cos^2\psi(s) ds \tag{3.38}$$
$$= \frac{4}{c^2}\cdot\frac{1}{2}(\omega_{\max}^2 - \omega_{\min}^2)\int_{s_i}^{s_f}\left|\frac{\partial \theta}{\partial s}\right|\cos^2\psi(s) ds \tag{3.39}$$
$$= 2(k_{\max}^2 - k_{\min}^2)\int_{s_i}^{s_f}\left|\frac{\partial \theta}{\partial s}\right|\cos^2\psi(s) ds, \tag{3.40}$$

where in the last line, $k_{\max} = \omega_{\max}/c$ and $k_{\min} = \omega_{\min}/c$.

As mentioned previously, (3.40) measures the resolution for a single target, $\boldsymbol{z}$. Maximizing this will lead to better resolution. We also want to maximize scene coverage, so we will integrate this over all targets. Our final objective function that we want to maximize is the following:

$$V = 2(k_{\text{max}}^2 - k_{\text{min}}^2) \int_{\boldsymbol{Z}} \int_{s_i}^{s_f} \left| \frac{\partial \theta_z}{\partial s} \right| \cos^2 \psi_z(s) ds dz, \ \forall \boldsymbol{z} \in \boldsymbol{Z}, \tag{3.41}$$

where $\boldsymbol{Z}$ denotes the set of target positions. Here, all targets have an equal weight associated with them. If desired, we are able to change this weight so as to emphasize more important targets.

C. Optimization parameters

The parameters we use to maximize (3.41) are an aircraft's heading (η), pitch (λ), and steering angles (β_{LOS} and ϕ_B). The heading and pitch will directly influence θ and ψ that appear in (3.41). First, the aircraft's heading and pitch are defined recursively as the following for time s_{n+1}:

$$\eta(s_{n+1}) = \eta(s_n) + \Delta\eta(s_n) \tag{3.42}$$

$$\lambda(s_{n+1}) = \lambda(s_n) + \Delta\lambda(s_n). \tag{3.43}$$

The flight path, $\gamma(s)$, is then parametrized by the following:

$$\begin{aligned} \gamma_1(s_{n+1}) &= \Delta s \sin \eta(s_n) \cos \lambda(s_n) + \gamma_1(s_n) \\ \gamma_2(s_{n+1}) &= \Delta s \cos \eta(s_n) \cos \lambda(s_n) + \gamma_2(s_n) \\ \gamma_3(s_{n+1}) &= \Delta s \sin \lambda(s_n) + \gamma_3(s_n), \end{aligned} \tag{3.44}$$

where, Δs is the distance between transmit times and η and λ are the aircraft's heading and pitch at time s. Finally, $\psi_{\boldsymbol{z}}$ and $\theta_{\boldsymbol{z}}$ in eq. (3.41) are defined in eq. (3.29). Thus, we can see that an aircraft's heading and pitch directly influence the objective function.

In order to understand how the antenna steering angles impact the scene coverage and resolution, we will consider the geometry of the problem, seen in figure 3.5. The steering angles will primarily affect the antenna footprint, which will in turn affect the objective function by deter-

mining over which targets to integrate. The two steering angles we will consider are the elevation angle, called β_{LOS}, and the boresight angle, called ϕ_B. The minor axes of the antenna footprint

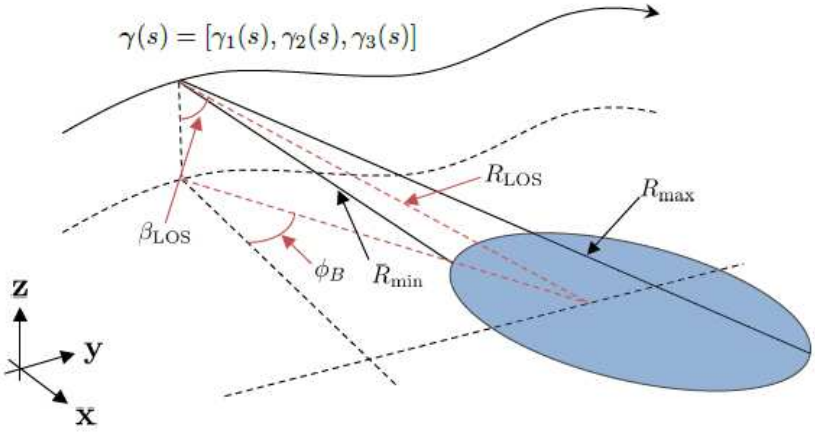

Figure 3.5: Scene Geometry

ellipse are determined by $\phi_B - \phi_{dy}$ and $\phi_B + \phi_{dy}$, where ϕ_{dy} is the antenna divergence angle. The major axes of the antenna footprint ellipse are determined by $\beta_{LOS} - \phi_{dz}$ and $\beta_{LOS} + \phi_{dz}$, where ϕ_{dz} is also antenna divergence. Below are figures illustrating the geometry of ϕ_{dy} and ϕ_{dz}.

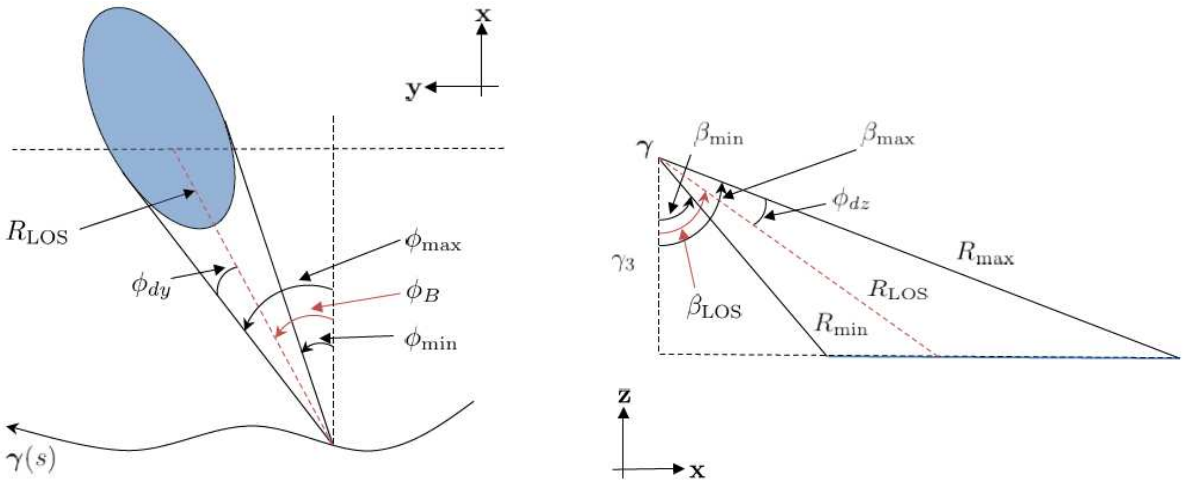

Figure 3.6: Left: Scene Geometry from Above, Right: Scene Geometry from Side

We are able to vary both β_{LOS} and ϕ_B in order to better maximize (3.41) which leads to maximizing scene coverage and resolution.

3.2.3 Constraints

We added a number of constraints to our problem in order to make it more realistic. First, we constrained our heading and pitch in order to mirror the capabilities of an actual aircraft. Additionally, we restricted our flight path to a minimum and maximum height.

Other practical considerations are the amount the steering angles are allowed to move. In order to avoid left-right artifacts, the antenna must remain side-looking. Thus, we add this constraint to our problem.

Lastly, in order for a target DCM to contribute to the objective function, it must lie within the antenna footprint. We assume the antenna footprint is an ellipse, and thus it follows the standard equation for ellipses. Specifically, a target lies within an antenna footprint if the following quantity is satisfied:

$$\left(\frac{x - c_x}{a}\right)^2 + \left(\frac{y - c_y}{b}\right)^2 \leq 1, \tag{3.45}$$

where (x, y) is the location of the target, a and b are the radii of the antenna footprint ellipse, (c_x, c_y) is the center of the antenna footprint ellipse, and ϕ_B is defined in figure 3.5.

In the case of spotlight and circular SAR, the antenna is steered towards the center of the target scene. Thus, we know (c_x, c_y) since this is the center of the target scene. Then we compute a by the following, with $h =$ the aircraft height:

$$a = (x_{max} - x_{min})/2, \tag{3.46}$$

where

$$x_{max} = h[\tan(\beta_{LOS} + \phi_{dz})] \tag{3.47}$$

$$x_{min} = h[\tan(\beta_{LOS} - \phi_{dz})], \tag{3.48}$$

with β_{LOS} and ϕ_{dz} defined in figures 3.5 and 3.6. We compute b by the following:

$$b = R_{center} \tan(\phi_{dy}), \tag{3.49}$$

where ϕ_{dy} is defined in figure 3.6 and R_{center} is the distance from the aircraft to $(c_x, c_y, 0)$.

In stripmap SAR, β_{LOS} is constant and $\phi_B = 0$. In the optimized flight path, β_{LOS} and ϕ_B are two of the optimization parameters that we vary. In spotlight and circular SAR, β_{LOS} and ϕ_B are steered towards the center of the scene. For spotlight and circular SAR, we compute these in the following way:

$$\beta_{LOS} = \arccos\left(\frac{h}{R_{center}}\right) \tag{3.50}$$

$$\phi_B = \arctan\left(\frac{|c_y - y|}{|c_x - x|}\right). \tag{3.51}$$

After computing the major and minor axis of the antenna footprint ellipses, the ellipses are rotated so that the major axis lies along the line of sight. This follows the process defined in [50]. Then the DCM of each target that lies within the antenna footprint is included in the objective function. All targets lying outside of the antenna footprints do not contribute to the objective function.

3.3 Results

It is common to define a flight path by its waypoints, namely discrete locations through which an aircraft flies. Our objective function will be used to optimize these waypoints in order to create an optimal flight path. Since we are transmitting pulses at the pulse repetition frequency (PRF)

rate between these waypoints, this can be thought of as interpolating between waypoints with transmitted pulses. At each waypoint, we will allow the aircraft to change its heading, pitch, and steering angles. In Section 3.2.2C, we defined a way to parametrize the flight path. This is what we use to compute the following results.

We will compare the results from spotlight, stripmap, and circular SAR scenarios against a varied flight path scenario found from optimizing the objective function. We will assume the following parameters:

- There are 100 targets evenly spread over the target scene.
- The flight path is 50000 meters long.
- There are 50 way points.
- The targets are points on the $xy-$plane.
- The center of the target scene is located at [7500, 0, 0].
- The bandwidth is [200, 800] MHz.

A greedy algorithm was used to maximize the objective function. A greedy algorithm for this problem looks at each waypoint, calculates the objective function for each possible heading, pitch, and steering angles (4-tuple), chooses the 4-tuple that scores the highest, then repeats for the next waypoint.

Here, a random 4-tuple is chosen if the objective function is zero for all possible 4-tuples at a given waypoint. This situation will occur if the aircraft is unable to see any targets for any 4-tuple at that waypoint.

In order to maintain consistency with the previous SDA resource allocation problem terminology, an optimal flight path will consist of an assigned task at each waypoint that maximizes the objective function. A task will consist of a heading, pitch, and steering angles 4-tuple. Below is

pseudo-code for this algorithm:

Result: *Optimal Flight Path*

generate all potential tasks;

for *waypoints* **do**

  compute objective function with tasks;

  if *at least one objective function value > 0* **then**

    choose task with highest objective function value;

  else

    choose random task;

  end

end

return *greedy path*;

Algorithm Greedy Algorithm

In all of the figures below, the flight path is in blue, the antenna footprints are represented by ellipses, and the targets are represented as the black dots. Additionally, the aerial view of the target scene with the antenna footprints for each of these flight types are shown in figures 3.8, 3.10, 3.12, 3.14.

In both the stripmap and spotlight modes, the aircraft flies in a straight path. In the stripmap case, the antenna remains fixed with respect to the aircraft. In the spotlight case, the antenna is steered towards the center of the target scene. The flight paths for these can be seen in figure 3.7 and Fig 3.9. It should be noted that the full flight path in these figures is not shown for scale purposes. Additionally, the asymmetry in the antenna footprints in figure 3.10 is due to the flight path not being centered around $y = 0$.

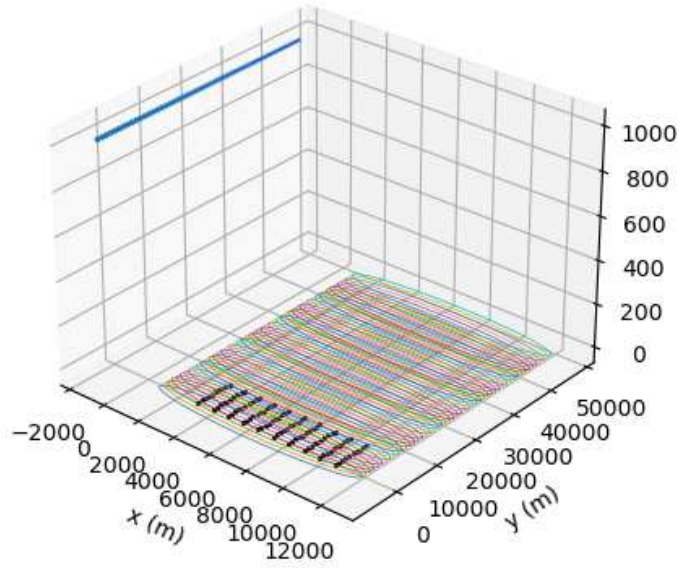

Figure 3.7: Stripmap mode flight path and corresponding footprints

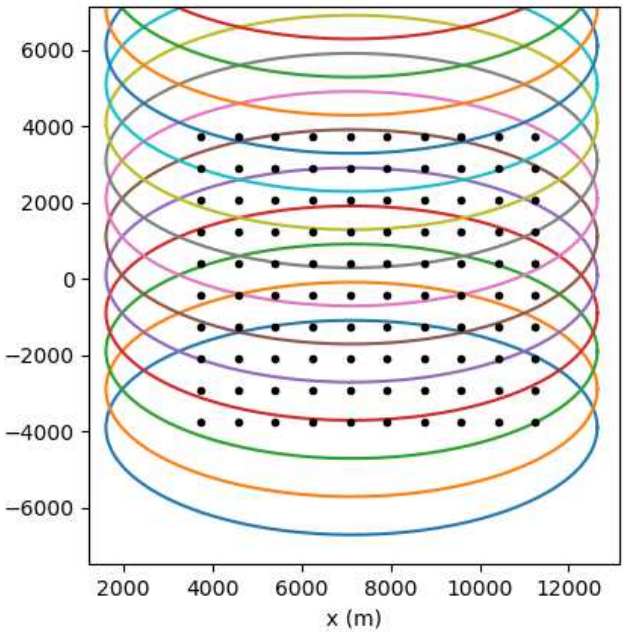

Figure 3.8: Antenna footprints in stripmap mode

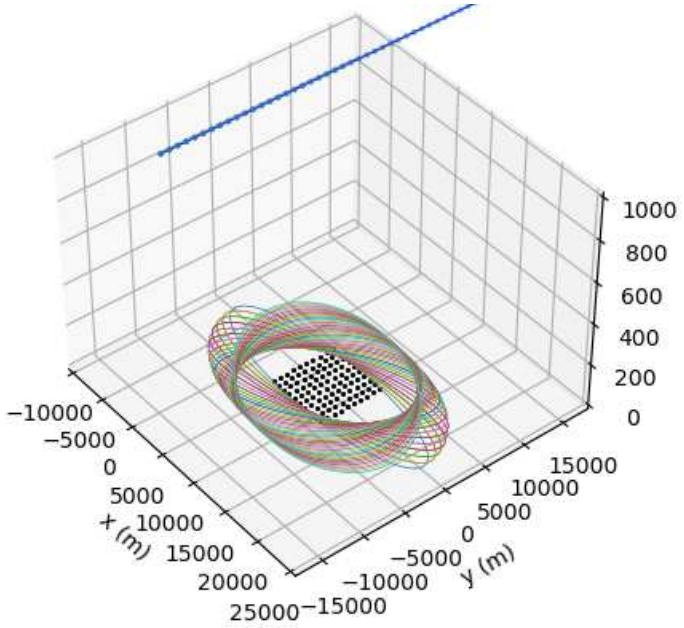

Figure 3.9: Spotlight mode flight path and corresponding footprints

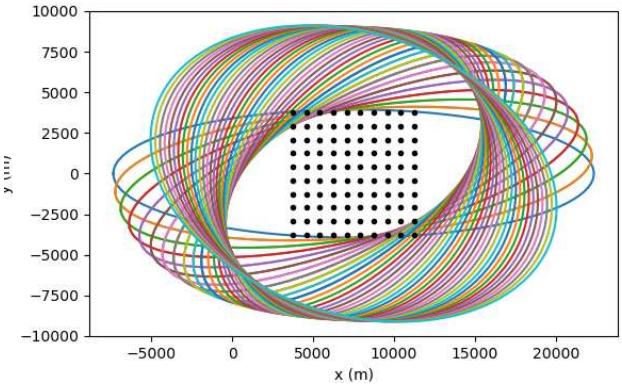

Figure 3.10: Antenna footprints in spotlight mode

In circular SAR, the aircraft flies in a circular path and steers the antenna to the center of the scene. This can be seen in figure 3.12. There is slight asymmetry in the antenna footprints in 3.12. Recall, we assumed we had a rectangular antenna. This asymmetry is due to this assumption.

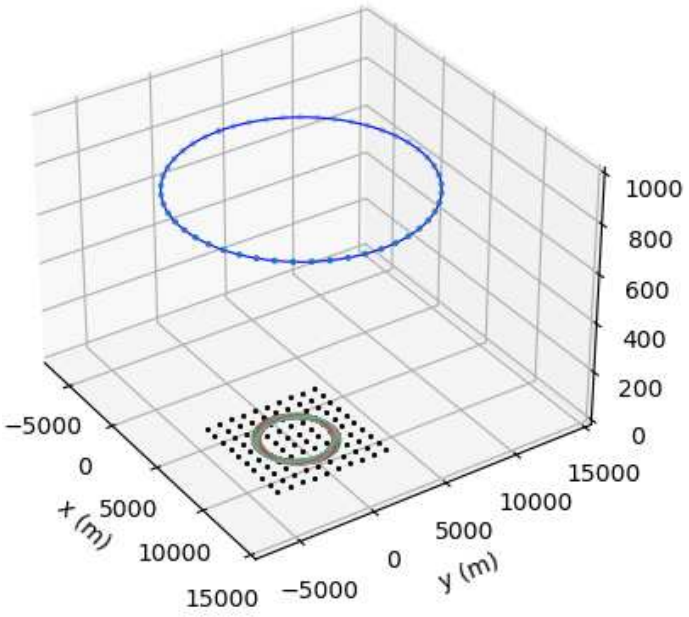

Figure 3.11: Circular SAR flight path and corresponding footprints

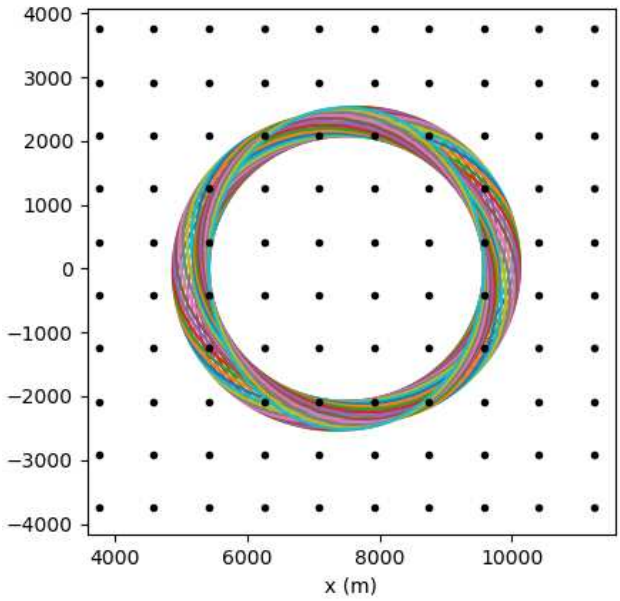

Figure 3.12: Antenna footprints in circular mode

Lastly, the optimized flight path and corresponding footprints are shown in figures 3.13 and 3.14. We see that the aircraft flies around the scene, but not in a circular path. The path generated flies in the best way to maximize scene coverage and image resolution.

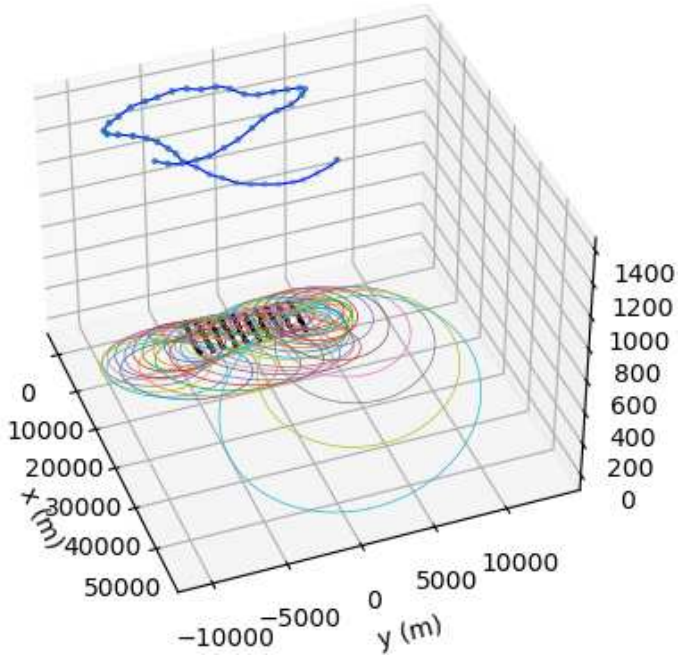

Figure 3.13: Optimized flight path with corresponding footprints

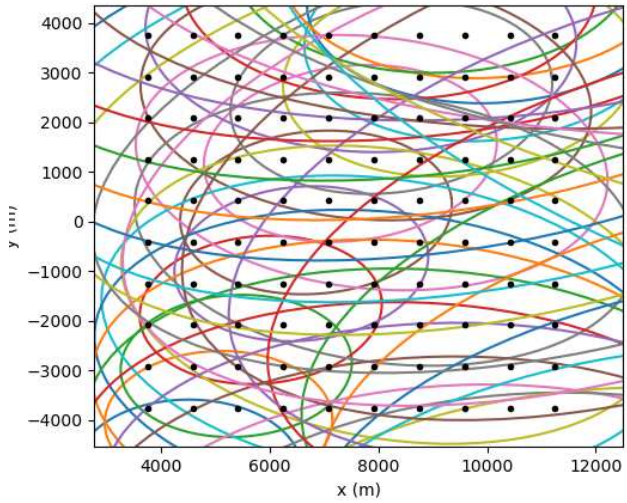

Figure 3.14: Antenna footprints in optimized flight path

The objective function values and percent of targets seen for each flight path are shown in figures 3.15 and 3.16, respectively. We see that the optimized flight path obtained the highest objective function value of all the other flight paths and was able to see all of the targets in the target scene.

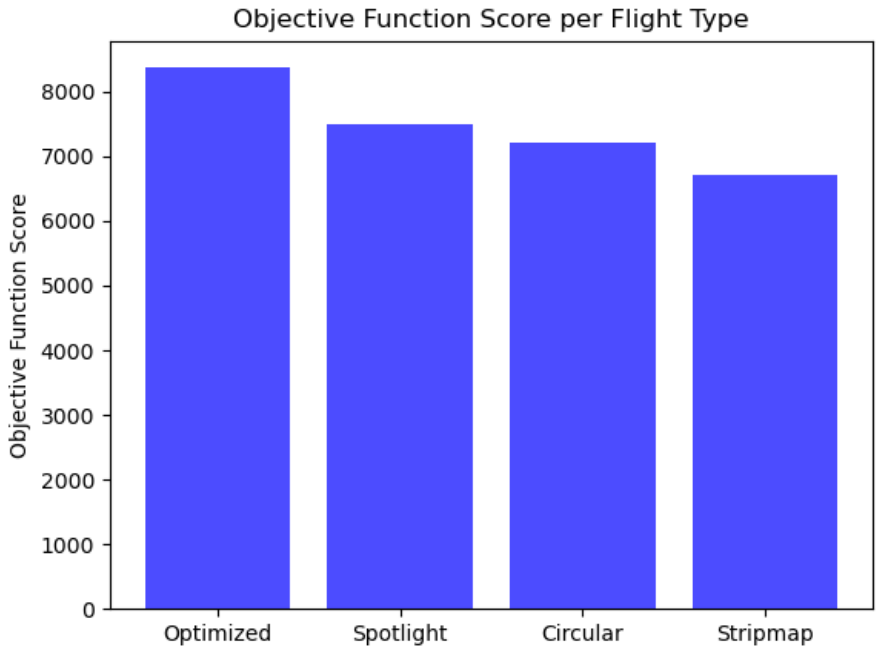

Figure 3.15: Flight Path Performance

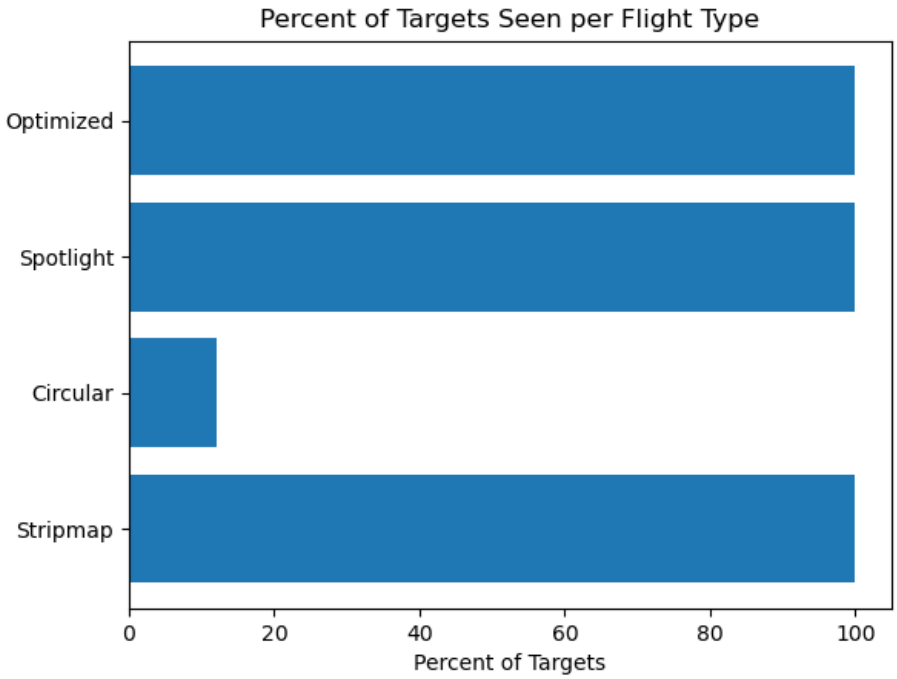

Figure 3.16: Percent of Targets Seen

These results are consistent with what we know about these stripmap, spotlight, and circular SAR modes. Namely, stripmap SAR tends towards better scene coverage than circular SAR and sometimes spotlight SAR. Spotlight SAR and circular SAR tend toward better resolution than stripmap SAR.

With the optimized flight path, the objective function obtained the highest value. Additionally, the aircraft was able to see all of the targets, whereas the circular scenario was not able to do this. Thus, this flight path is better than those generated in the stripmap, spotlight, and circular SAR scenarios.

This flight path accurately maximizes both scene coverage and image resolution. Our goal was to find a tool for finding the balance between maximizing scene coverage and image resolution. We can see from our generated flight path that this is what we accomplished.

3.3.1 Conclusion

In this chapter, we discussed a mathematical tool that takes both target resolution and scene coverage into consideration [10]. We began by looking at the data set determined by the data-collection manifold. It was noticed that the size of this manifold was directly related to the attainable resolution for a target. We followed the process developed by Dr. Scott Altrichter to create a formula for calculating the surface area of this manifold which quantifies the resolution for a point target.

After this, we looked at the scene geometry. By looking at the antenna footprint and antenna steering, we could quantify scene coverage. Combining these two developments into a single objective function gave us a tool that quantifies the trade-off between resolution and coverage. The main difference from Dr. Altrichter's work here is that we considered the 2D components of the DCM, instead of the 3D components.

We extended Dr. Altrichter's work by optimizing the objective function with respect to the aircraft's steering angles as well as its heading and pitch. We compared the results of this generated flight path with other common SAR flight paths. We showed that we were able to generate a better flight path than traditional SAR methods by maximizing the objective function.

It should be noted that we restricted the pitch rate so as to ensure the aircraft did not fly indefinitely upward. Without this restriction, the aircraft will fly upward in order to increase the antenna footprint and generate a high objective function value. The further the aircraft is from the targets, the more likely it is to have a low SNR value. More detailed consideration of this effect on SNR is addressed in Section 4.1.

We needed to determine a simple and general way to parameterize a flight path in order to exploit this objective function. In Section 3.2.2C, we defined a recursive generator that relied upon changing pitch and heading angles at each step of the path to direct the aircraft to the next step of the path [10]. This formulation presented us with parameters that we could adjust to yield different paths and different objective values.

With the objective function defined, we were able to use the objective function as a way to determine optimal flight paths. Not only did we optimize for an aircraft's heading and pitch, we were also able to optimize for the aircraft's antenna steering angles. The simulations demonstrated that this objective function can be used to define flight paths that provide more scene coverage and resolution than in traditional SAR modes.

This lays the foundation for much future study. The objective function not only quantifies the trade-off between target resolution and scene coverage, but also provides a flexible tool that allows us to determine flight paths in a wide range of scenarios.

In the next chapter, we show the versatility of the objective function. We start by considering the effects of SNR on our objective function. We then add a priority to visit specific targets or target scenes. We also develop an extension of the objective function to plan flight paths for multiple unmanned aerial vehicles (UAVs). Lastly, we implement two additional optimization algorithms to our objective function and compare the results.

Chapter 4

Flight Path Planning Extensions and Applications

In this chapter, we will show the flexibility of our approach in planning an optimal SAR flight path. We will start by adding an SNR consideration. Backscattered power decays as $1/R^4$. Thus, the further from the target scene the aircraft flies, the lower the SNR. We will compare various modifications to the objective function that account for the SNR per target.

We will also observe how the flight path changes when we add a priority to see specific targets. In addition to this, we will add a target scene with a higher priority. We will see that the flight path is modified to see the higher priority targets in both of these cases.

After this, we will show how this objective function may be used to plan flights for multiple vehicles. We will specifically implement this for multiple UAVs. We will start by generating flight paths for two UAVs. This formulation may be extended to an arbitrary number of UAVs.

Lastly, we will implement additional algorithms in order to maximize the objective function. We will compare the flight path resulting from using the greedy algorithm of Chapter 3 with a genetic algorithm and a random search algorithm.

Ultimately, we will show that this formulated objective function may easily be modified for various applications and extensions. The specific applications and extensions we will describe are only a few of the plethora of possibilities we can explore with this objective function.

4.1 SNR Effects on Flight Path

4.1.1 SNR Overview

Recall our objective function:

$$V = 2(k_{\text{max}}^2 - k_{\text{min}}^2) \int_{\boldsymbol{Z}} \int_{s_i}^{s_f} \left| \frac{\partial \theta_z}{\partial s} \right| \cos^2 \psi_z(s) ds dz, \ \forall \boldsymbol{z} \in \boldsymbol{Z}. \tag{4.1}$$

With the integral over the targets in the antenna footprint, the aircraft had a tendency to fly very far upward. In doing this, the antenna footprint was large and contained many, or all, possible targets. However, since backscattered power decays as $1/R^4$, SNR decreases as the aircraft flies further from a target scene. Thus, we want to penalize the aircraft from flying indefinitely upward.

We will determine how exactly to create this penalizing factor by starting with the equation for SNR for a general SAR scenario. The derivation of this, by Armin W. Doerry, is detailed in [51]. Doerry defines the SAR SNR equation as the following:

$$SNR_{image} = \left(\frac{P_{avg} G_A^2 \lambda^3 \rho_y \sigma_{0,ref}}{2(4\pi)^3 R^3 v_x (kTF_N) a_{wr}}\right) \left(\frac{1}{L_{radar} L_{atmos}}\right) \left[\frac{a_{wr} a_{wa}}{L_r L_a}\right] \left[\frac{f}{f_{ref}}\right]^n. \tag{4.2}$$

According to Doerry, the terms in the brackets are usually nearly unity. Thus, we will assume they are equal to 1. Then we are left with the following equation for SNR:

$$SNR_{image} = \left(\frac{P_{avg} G_A^2 \lambda^3 \rho_y \sigma_{0,ref}}{2(4\pi)^3 R^3 v_x (kTF_N) a_{wr}}\right) \left(\frac{1}{L_{radar} L_{atmos}}\right), \tag{4.3}$$

where we have the following definitions:

	Definitions
P_{avg}	average signal power
G_A	transmitter antenna gain factor
λ	wavelength
ρ_y	range resolution
$\sigma_{0,ref}$	target Radar Cross Section (m^2)
R	range vector from target to antenna (m)
v_x	platform velocity (m/s), horizontal and orthogonal to the target direction
k	Boltzmann's constant = 1.38×10^{-23} J/K
T	nominal scene noise temperature ≈ 290 K
F_N	system noise factor for the receiver
a_{wr}	azimuth impulse response broadening factor ≈ 1.2
L_{radar}	microwave transmission loss factor due to miscellaneous sources
L_{atmos}	microwave transmission loss factor due to atmospheric sources

Now we will examine each element in equation (4.3) to determine which elements have a dependency on the flight path.

- **Signal power**: This does not rely on the flight path, this will be the same regardless of the flight path.
- **Wavelength**: This will remain the same regardless of the flight path.
- **Transmitter antenna gain**: The equation for the transmitter antenna gain factor is the following:

$$G_A = \frac{4\pi A_e}{\lambda^2} = \frac{4\pi \eta_{ap} A_A}{\lambda^2}, \tag{4.4}$$

where A_e is the effective antenna aperture, A_A is the physical area of the aperture, and η_{ap} is the aperture efficiency [51]. The aperture efficiency is approximately 0.5. Since the physical area of the aperture remains constant throughout the flight path, the transmitter antenna gain factor will not change with the flight path.

- **Range resolution**: This depends on the bandwidth which does not depend on the flight path.
- **Radar cross section**: We will assume all targets are the same and are point sources, so we will assume the target RCS will not change with respect to the aircraft's flight path.
- **Range vector**: This will depend on the flight path.
- **Platform velocity**: This will change with respect the aircraft's steering angles and flight path. We will define this as the following:

$$v_x = v\sin(\theta), \tag{4.5}$$

  where v is the velocity of the aircraft and θ is the angle between the velocity vector and the look direction.
- **Boltzmann's constant**: This is constant by definition.
- **Nominal scene noise temperature**: This is assumed to be constant (approximately 290 K).
- **System noise factor for the receiver**: This will not be dependent on the aircraft's flight path.
- **Azimuth impulse response broadening factor**: The is assumed to be constant (approximately 1.2).
- **Microwave transmission loss**: We will ignore this factor due to both the atmospheric and miscellaneous sources.

4.1.2 Incorporating into Objective Function

Based on this analysis, the aircraft's flight path only affects the range vector and platform velocity. Thus, in order to account for SNR in our flight path planning, we need only consider the range vector and platform velocity components.

The platform velocity only appears when we use steering angles. First we will ignore the platform velocity and focus on the range. We will later add the platform velocity into our objective function.

The range appears in equation (4.3) as $1/R^3$. Thus, we will modify our objective function by a factor of $1/R^3$. We will look at how the flight path varies when we both multiply the objective function by this factor and when we add this factor to the objective function.

The two modified objective functions we will test are below:

$$V = 2(k_{\max}^2 - k_{\min}^2) \int_{\boldsymbol{Z}} \frac{C}{R_z^3} \int_{s_i}^{s_f} \left| \frac{\partial \theta_z}{\partial s} \right| \cos^2 \psi_z(s) ds dz, \ \forall \boldsymbol{z} \in \boldsymbol{Z} \tag{4.6}$$

$$V = 2(k_{\max}^2 - k_{\min}^2) \int_{\boldsymbol{Z}} \left(\int_{s_i}^{s_f} \left| \frac{\partial \theta_z}{\partial s} \right| \cos^2 \psi_z(s) ds + \frac{C}{R_z^3} \right) dz, \ \forall \boldsymbol{z} \in \boldsymbol{Z}, \tag{4.7}$$

where R_z is the range to target z and C is a constant.

After we test these objective functions, we will add the platform velocity consideration. Namely, we will also test the following objective functions:

$$V = 2(k_{\max}^2 - k_{\min}^2) \int_{\boldsymbol{Z}} \frac{C}{v_x R_z^3} \int_{s_i}^{s_f} \left| \frac{\partial \theta_z}{\partial s} \right| \cos^2 \psi_z(s) ds dz, \ \forall \boldsymbol{z} \in \boldsymbol{Z} \tag{4.8}$$

$$V = 2(k_{\max}^2 - k_{\min}^2) \int_{\boldsymbol{Z}} \left(\int_{s_i}^{s_f} \left| \frac{\partial \theta_z}{\partial s} \right| \cos^2 \psi_z(s) ds + \frac{C}{v_x R_z^3} \right) dz, \ \forall \boldsymbol{z} \in \boldsymbol{Z}. \tag{4.9}$$

In order for C to maintain correct units, we first will perform dimensional analysis on equations (4.7) and (4.9). First, the term $\int_{s_i}^{s_f} \left| \frac{\partial \theta_z}{\partial s} \right| \cos^2 \psi_z(s) ds$ is dimensionless. The term $\frac{1}{R_z^3}$ has dimensions $\frac{1}{length^3}$ and the term $\frac{1}{v_x R_z^3}$ has dimensions $\frac{1}{\frac{length}{time} length^3} = \frac{time}{length^4}$. Thus, in equation (4.7), C has dimensions of $length^3$. In equation (4.9), C has dimensions of $\frac{length^4}{time}$. In this context, $length =$ meters and $time =$ seconds. Thus, C has units of either m^3 in (4.7) or $\frac{m^4}{s}$ in (4.9).

It should be noted the objective function is dimensionless and thus unit-free. In order to ensure the objective function is dimensionless in equations (4.6) - (4.9), the units of C in equations (4.6) and (4.8) are m^3 and $\frac{m^4}{s}$, respectively.

For comparison, the following figures represent a flight path from using the original objective function, equation (4.1). The objective function value was 7311.64 in this scenario.

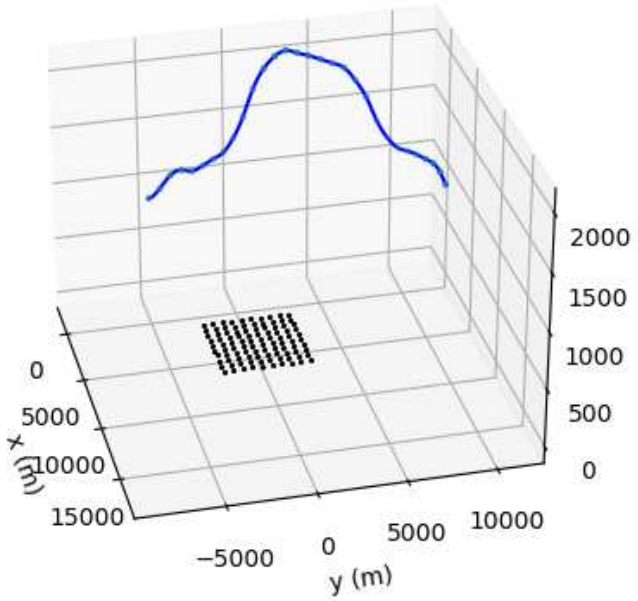

Figure 4.1: Flight path and targets

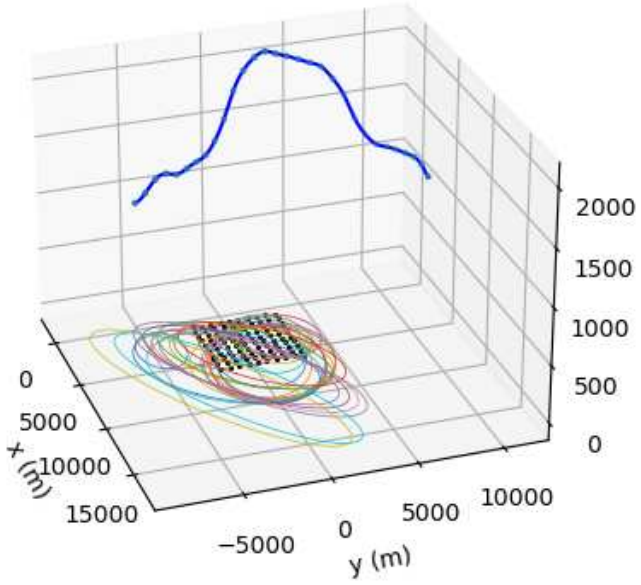

Figure 4.2: Flight path, targets, and antenna footprints

4.1.3 Results from (4.6)

The following are results from applying equation (4.6). The objective function in this scenario was small since R was large. The objective function value was $6.12e \times 10^{-8}$.

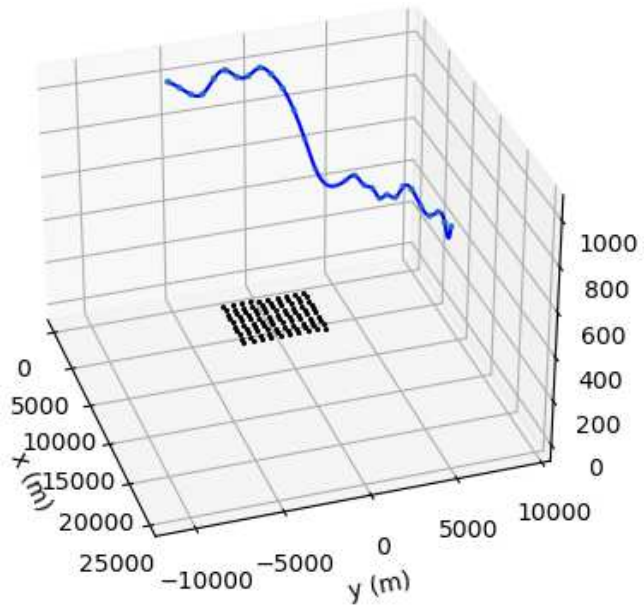

Figure 4.3: Flight path and targets

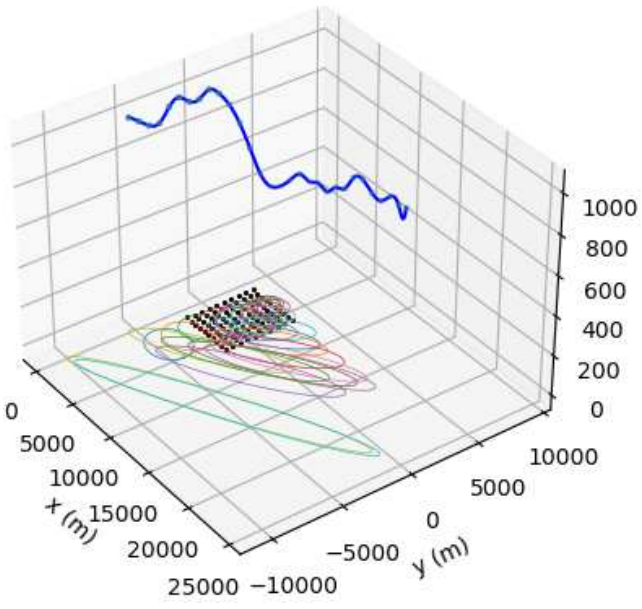

Figure 4.4: Flight path, targets, and antenna footprints

4.1.4 Results from (4.7)

Next, figures 4.5 and 4.6 are results from using equation (4.7) as the objective function with different constant values, C.

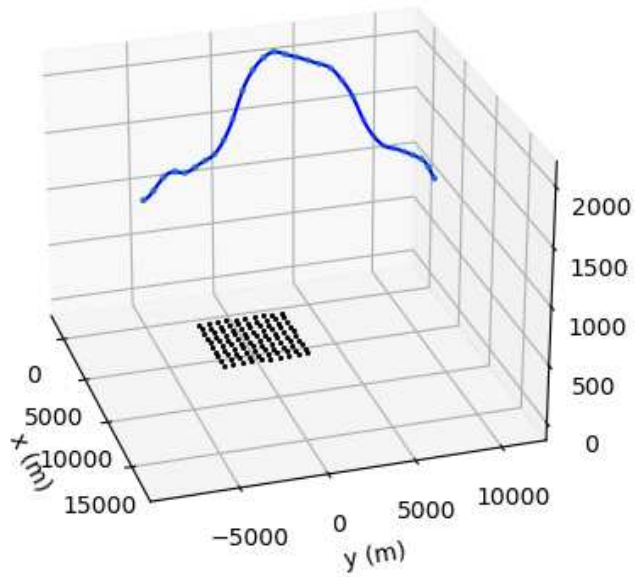

Figure 4.5: Flight path and targets with $C = 1\ m^3$, objective function value is 7311.64

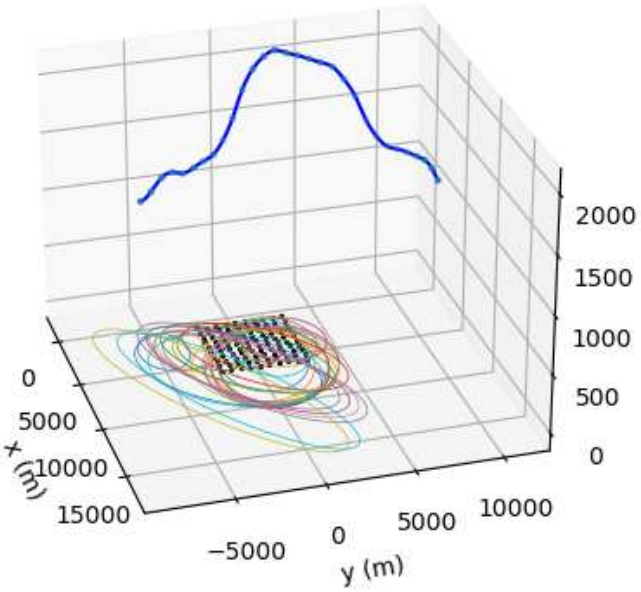

Figure 4.6: Flight path, targets, and antenna footprints with $C = 1\ m^3$, objective function value is 7311.64

Since $1/R^3$ is such a small value, the objective function value barely changes until $C \approx 1 \times 10^9\ m^3$. This is shown in figures 4.7 - 4.12:

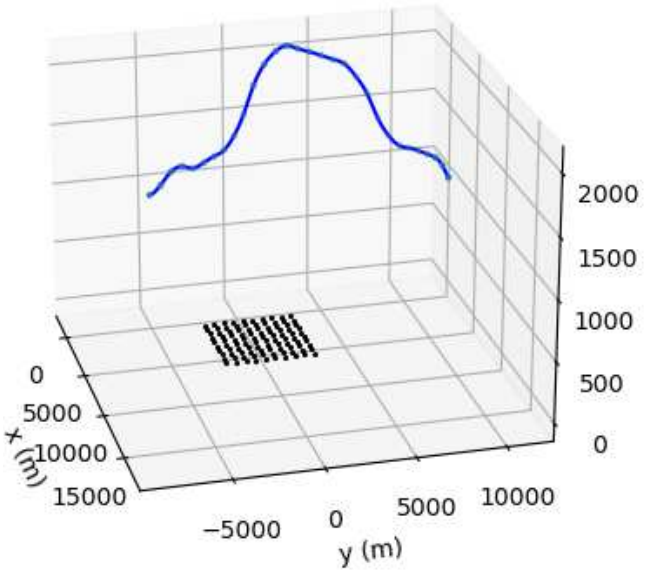

Figure 4.7: Flight path and targets with $C = 1 \times 10^9\ m^3$, objective function value is 7317.12

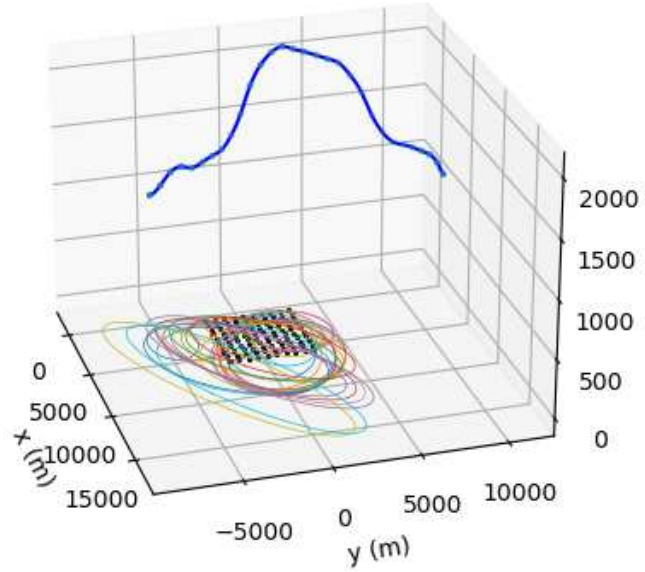

Figure 4.8: Flight path, targets, and antenna footprints with $C = 1 \times 10^9\ m^3$, objective function value is 7317.12

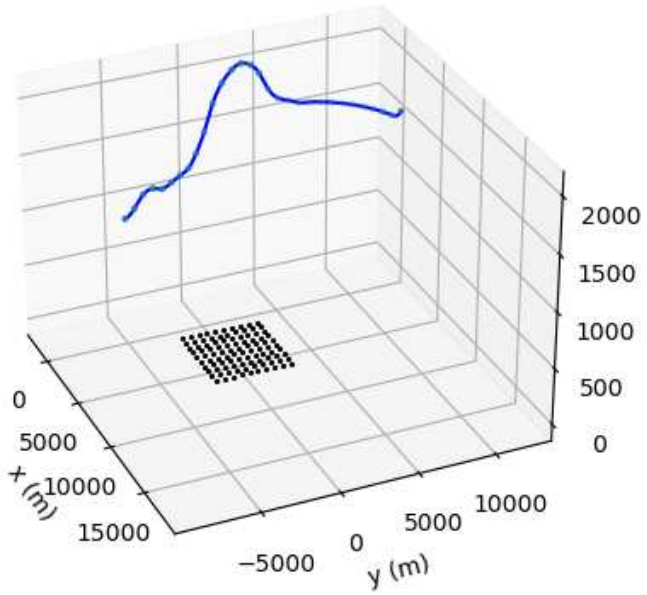

Figure 4.9: Flight path and targets with $C = 1 \times 10^{10}\ m^3$, objective function value is 7193.37

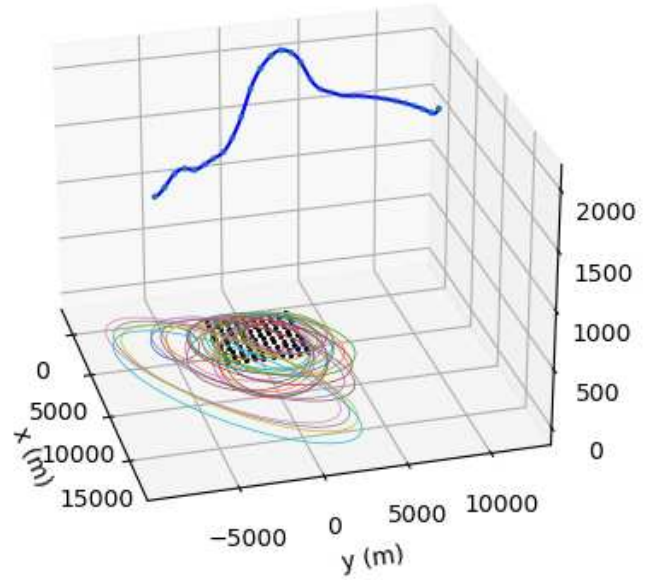

Figure 4.10: Flight path, targets, and antenna footprints with $C = 1 \times 10^{10}\ m^3$, objective function value is 7193.37

When equation (4.7) is optimized, the flight path is not affected until C is a very large value. At $C \geq 4 \times 10^{10}\ m^3$, all the flight paths look nearly the same. This is due to the size of the remaining terms of the objective function. Thus, the objective function is dominated by the C/R^3 term and resolution is no longer a leading factor in the objective function. Since one of our main goals with the objective function is

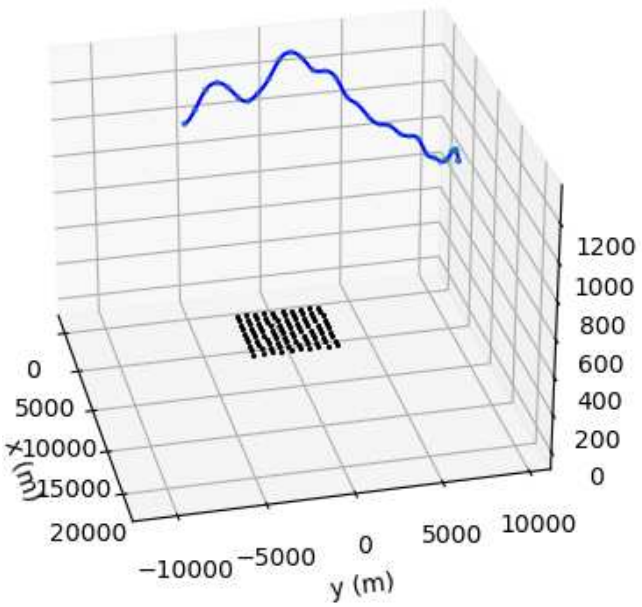

Figure 4.11: Flight path and targets with $C = 4 \times 10^{10}\ m^3$, objective function value is 7939.82

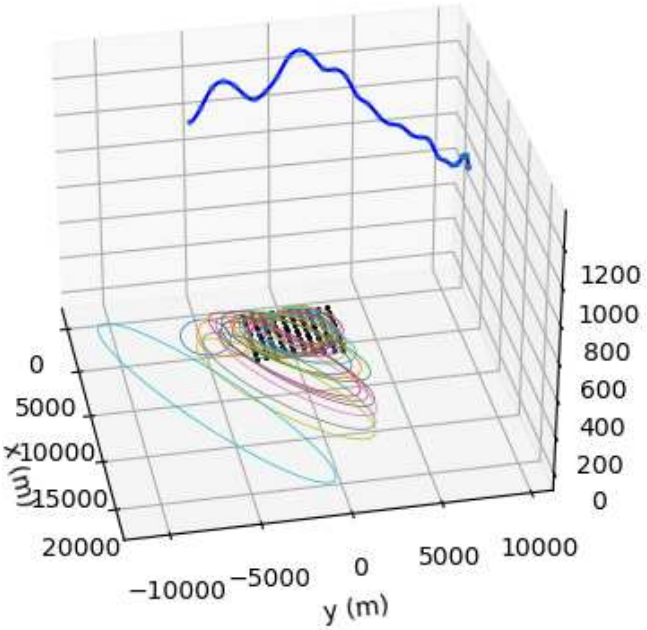

Figure 4.12: Flight path, targets, and antenna footprints with $C = 4 \times 10^{10}\ m^3$, objective function value is 7939.82

to maximize resolution, we will ensure that $C \leq 4 \times 10^{10}\ m^3$. In both cases, the flight path tends to a lower flight path. This is what we expected - the SNR is higher the closer the antenna is to the target scene.

4.1.5 Results from (4.8)

Now we will look at results from using equation (4.8) as the objective function, shown in figures 4.13 and 4.14.

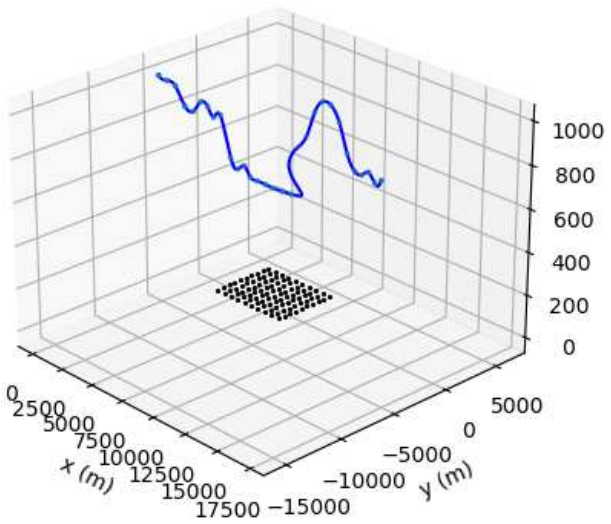

Figure 4.13: Flight path and targets with an objective function value of 4.21×10^{-8}

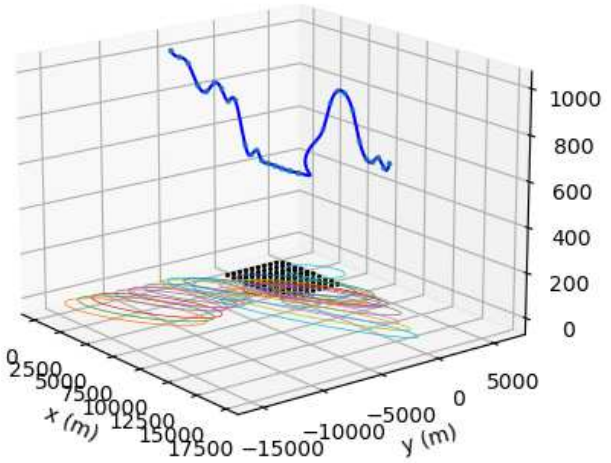

Figure 4.14: Flight path, targets, and antenna footprints with an objective function value of 4.21×10^{-8}

4.1.6 Results from (4.9)

Lastly, the results from using equation (4.9) are seen in figures (4.15) - (4.17):

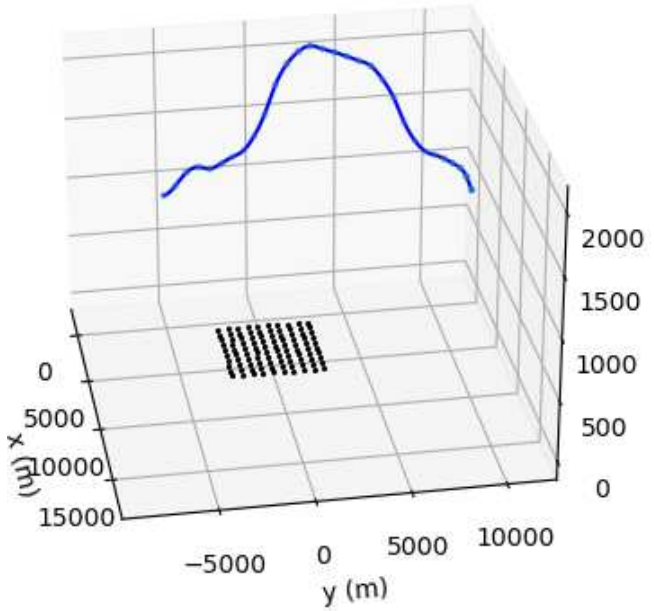

Figure 4.15: Flight path and targets with $C = 1\ \frac{m^4}{s}$, objective function value is 7311.64

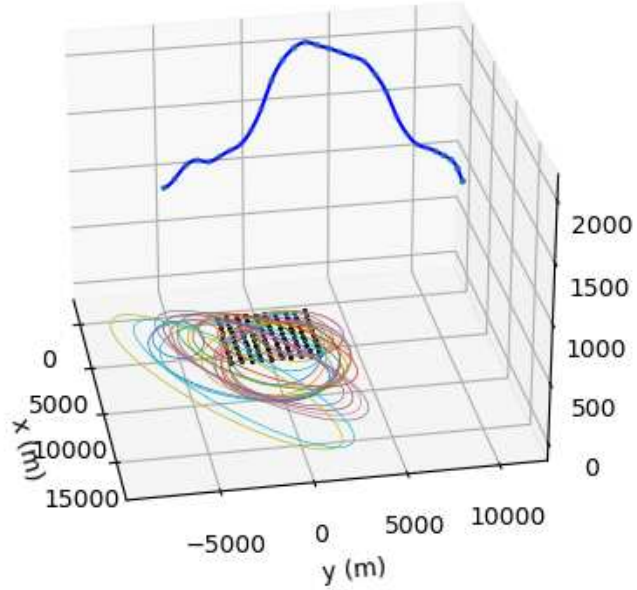

Figure 4.16: Flight path, targets, and antenna footprints with $C = 1\ \frac{m^4}{s}$, objective function value is 7311.64

Once again, the flight path does not change very much until after C is very large, shown in figures 4.17 and 4.18.

In summary, we used the expression for SAR SNR in [51] to incorporate the effect of SNR in our objective function. We looked at multiple ways to incorporate this term in our objective function. All the

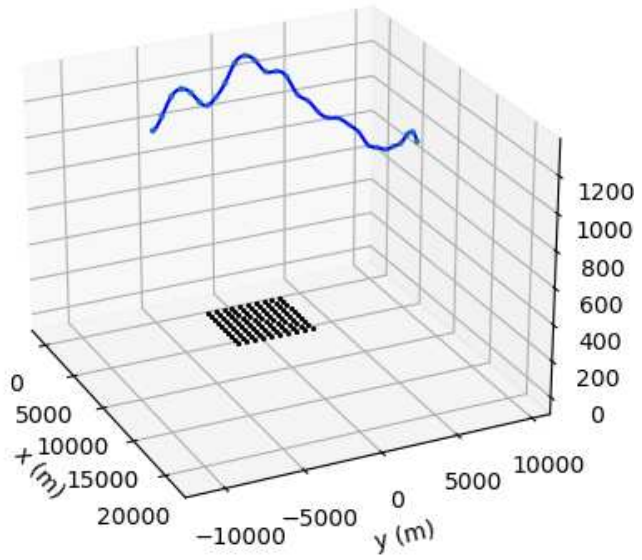

Figure 4.17: Flight path and targets with $C = 4 \times 10^{10}\ \frac{m^4}{s}$, objective function value is 7925.74

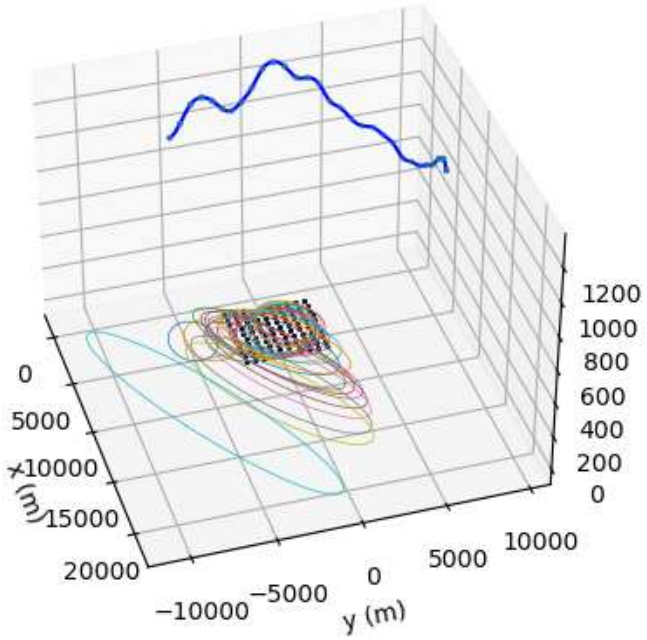

Figure 4.18: Flight path, targets, and antenna footprints with $C = 4 \times 10^{10}\ \frac{m^4}{s}$, objective function value is 7925.74

objective functions that include SNR result in flight paths with limited altitude. This is what we hoped to accomplish by adding this term.

Once we determined the SNR term needed for this scenario, including it in the objective function was quite simple. This is just one example of the versatility of this objective function. Next, we will see that it is straightforward to add a weighting term to various targets in a target scene or multiple target scenes.

4.2 Adding Target Priority

In a realistic scenario, some targets will be more important to visit than others. We will add a priority of visitation function to the objective function in the following way:

$$V = 2(k_{\text{max}}^2 - k_{\text{min}}^2) \int_{\boldsymbol{Z}} p(z) \int_{s_i}^{s_f} \left| \frac{\partial \theta_z}{\partial s} \right| \cos^2 \psi_z(s) ds dz, \ \forall \boldsymbol{z} \in \boldsymbol{Z}, \tag{4.10}$$

where $p(z)$ is the priority of visiting target z. As in the SDA scheduling problem, this priority function may be set in various ways. We can make this a function of time since the target was last seen. This may be set as a constant value. There is much flexibility in this definition.

First, we examine the results from setting only a few targets at a higher priority than the rest. In figure 4.20, the three magenta targets on the top left corner had a constant and equal priority greater than the constant and equal priorities of the black targets.

In figure 4.19, all of the targets have an equal and constant priority of visitation. In the equally weighted target scene, the antenna footprints gravitate towards the center of the target scene to see the most targets for as long as possible. This is to maximize coverage and resolution per target equally.

In contrast, when the three corner targets are weighted heavier, the antenna footprints are steered towards the heavier weighted targets. This is to maximize the resolution of those specific targets more than the rest of the targets.

In addition to prioritizing individual targets, it may also be important to prioritize an entire target scene over another target scene. In figure 4.21, the targets in the magenta target scene have a heavier weight than the targets in the black target scene.

We can see that when the aircraft starts its flight in the middle of the scenes, it flies around the heavier weighted target scene.

This modification to our objective function has many benefits. This allows us to focus on a particular target or patch of targets more than the remaining targets. Additionally, every target may have its own

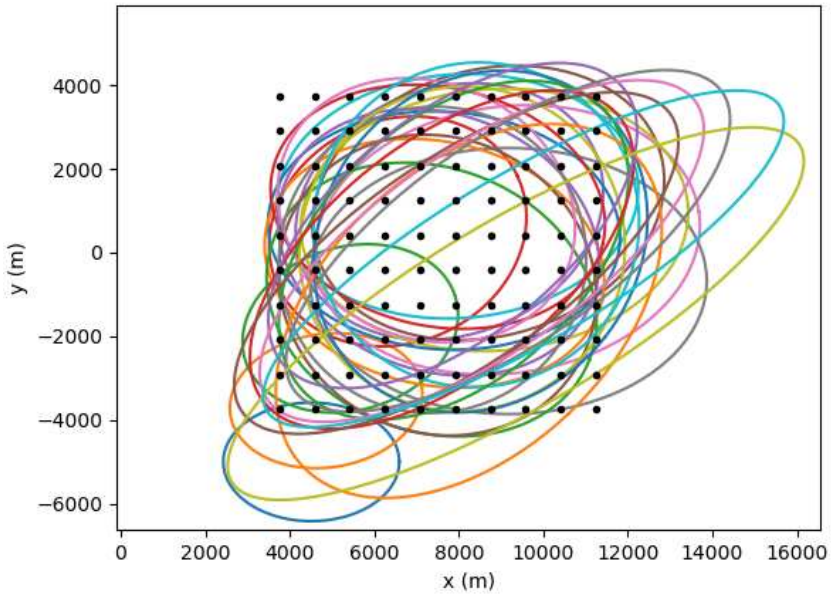

Figure 4.19: Antenna footprints with equal target priority

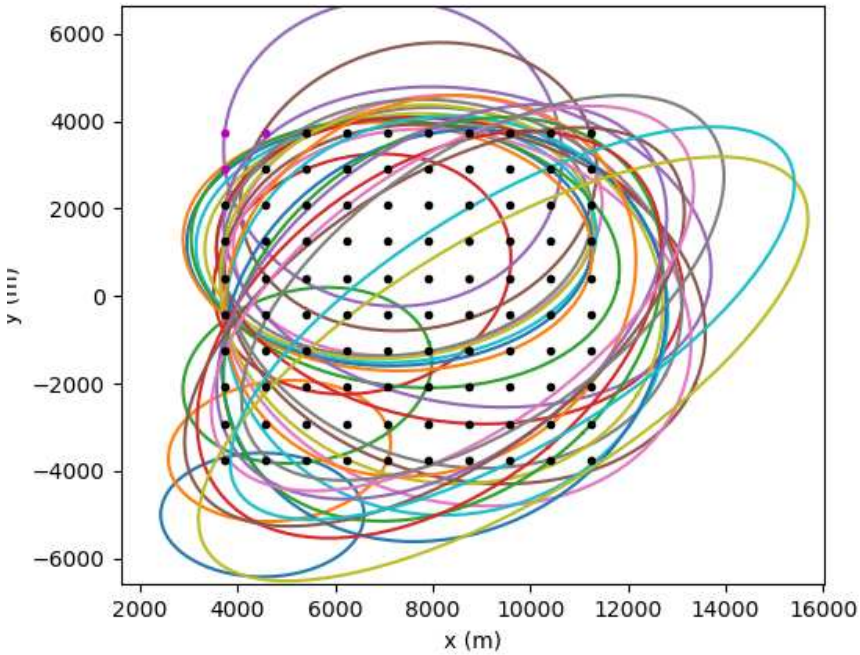

Figure 4.20: Antenna footprints with three weighted targets, colored magenta

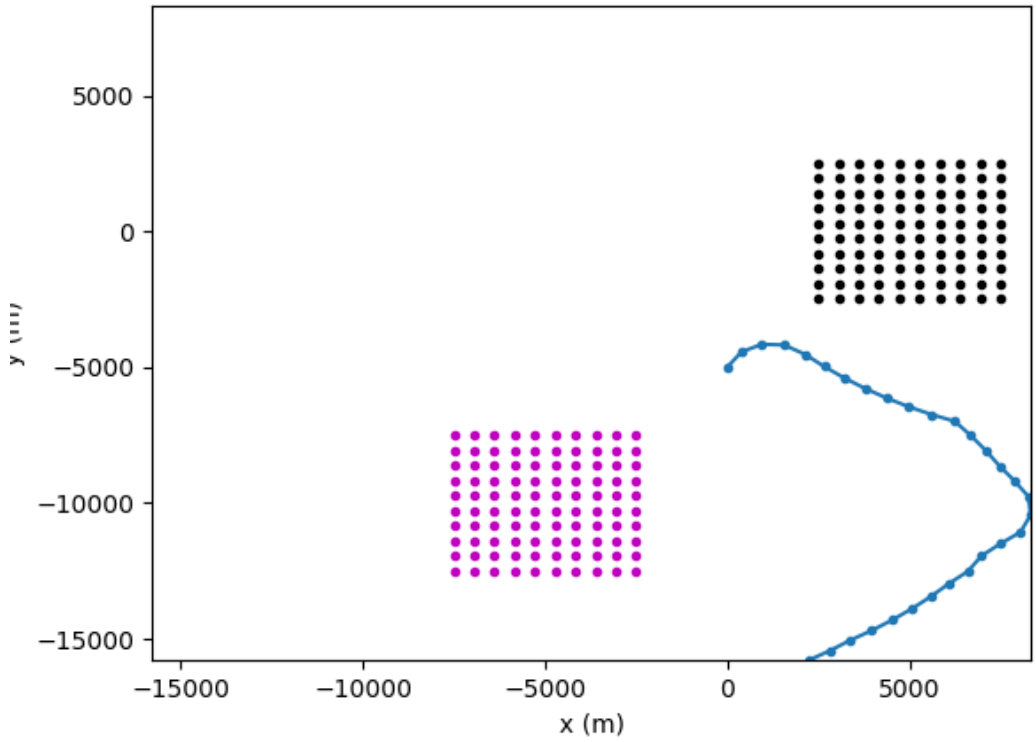

Figure 4.21: Two target scenes. Magenta targets are weighted heavier than black targets.

unique weight to it, thus allowing one to look at higher priority targets and gradually look at lower and lower priority targets.

4.3 Multiple Vehicle Path

The last extension to the objective function that we will pursue is adding an application to unmanned aerial vehicles (UAVs). UAVs have existed since before manned aircraft [52]. Their potential military combat application was realized in the two World Wars. By the Vietnam War, the interest in UAVs turned to surveillance and stealth. Today, UAVs are still used for military purposes. They are also used for applications such as archaeology, filmmaking, search and rescue, and scientfic research, along with a plethora of other applications.

Since the purpose of SAR is imaging, UAV imaging applications are pertinent. One application is stitching together images taken from multiple UAVs [53] to create one large image. This provides the ability to search a greater region of interest. Another application is the ability to generate 3-dimensional images using multiple images taken from UAVs.

In SAR, we have monostatic and multistatic sensors. With a multistatic sensor, the transmitter and receiver are in different locations. With a monostatic sensor, the transmitter and receiver are in the same location. In the work presented here, we will use monostatic sensors.

Below we use the objective function formulated in the previous sections to plan an optimal flight path for multiple UAVs. The optimization process will find the optimal waypoints for the UAVs to pass through. Using UAVs in this scenario allow for greater maneuverability than in the previous sections where we only considered a single, traditional aircraft. In the next section, we will use the work completed in the previous sections to optimize against each UAVs heading, pitch, and antenna steering angles.

4.3.1 Mathematical Formulation

We use equation (4.1) in order to build an objective function for optimizing the flight paths of multiple UAVs. We will modify the objective function in the following way:

$$V = 2(k_{\text{max}}^2 - k_{\text{min}}^2) \sum_{\boldsymbol{U}} \int_{\boldsymbol{Z}} \int_{s_i}^{s_f} \left| \frac{\partial \theta_{z,u}}{\partial s} \right| \cos^2 \psi_{z,u}(s) ds dz, \ \forall \boldsymbol{z} \in \boldsymbol{Z}, \ \forall u \in \boldsymbol{U}, \tag{4.11}$$

where $\boldsymbol{U}$ is the set of UAVs and the subscript u refers to each UAV. This modification to the objective function assumes that data can be combined coherently. This requires accurate synchronization of timing systems.

Below are results from multiplying the objective function by $1/R^3$, where the blue and green ellipses are the corresponding antenna footprints of the blue and green UAVs:

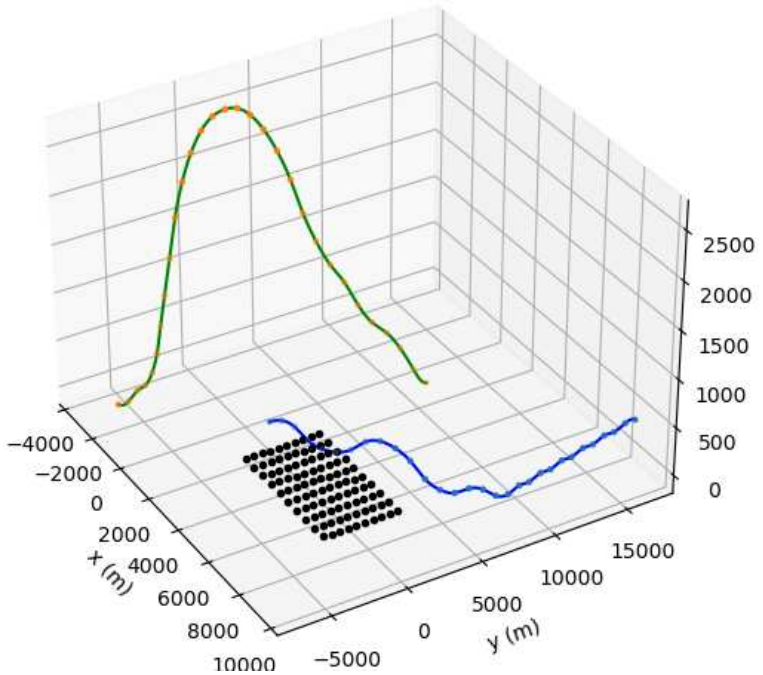

Figure 4.22: Two UAVs' flight paths

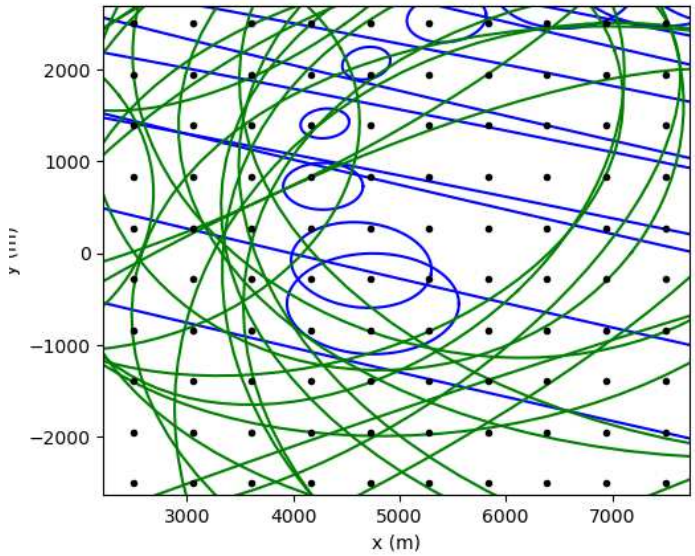

Figure 4.23: Antenna footprints of UAVs.

In a scenario with two target scenes, we get the following flight paths:

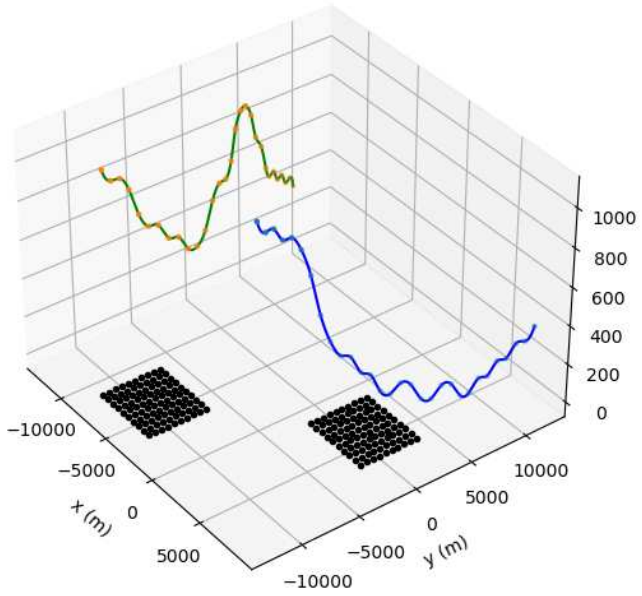

Figure 4.24: Greedy flight path

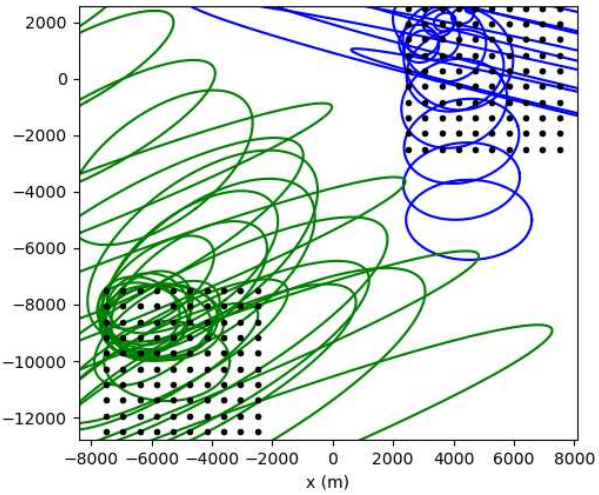

Figure 4.25: Antenna footprints from greedy flight path

In these examples presented, we can see how versatile this objective function is. We are able to use it as a framework for various scenarios. By making slight modifications to the objective function, we can add a higher priority of seeing specific targets or target scenes. This is a realistic concern for many scenarios. It is likely that in a given scenario, some targets or target scenes are more important to observe than others. When we added a target or target scene priority, we see from the results that the aircraft flies or steers its antenna more heavily towards those targets or target scenes.

We were also able to add an SNR consideration to the objective function by either adding or multiplying the SNR term to the objective function. Considering SNR in a SAR scenario is an important real-life concern. Without this consideration, the aircraft tends to a higher altitude in order to increase the antenna footprint. However, realistically, as the range from the target increases, the backscattered power decreases. Thus, incorporating SNR into the objective function is crucial.

Additionally, by again making a slight modification to the objective function, we were able to plan a flight path for multiple vehicles. This is especially beneficial to a scenario with multiple UAVs. We looked at scenarios with a single target scene and multiple target scenes. In both of these cases, the objective function was able to plan flight paths for both UAVs.

We have only begun to touch upon all of the possibilities with this objective function. Given its versatility, this objective function provides the foundation to many other avenues. By making various modifications, it may be used in an abundance of applications.

Chapter 5

Additional Algorithms

One factor that limits the quality of the flight path is the algorithm used to maximize the objective function. The optimization algorithm implemented in the results of Chapter 4 is the same as in Chapter 3. This is a simple greedy algorithm. This only chooses the next best place for the aircraft to fly and steering angles to point. It does not consider any future possibilities. Consequently, there are cases where the flight path gets “stuck". Namely, the path can end up in a position where it can only fly lower than the specified height constraint.

Additionally, it can be seen from some of the figures in Chapter 4 that there are some parts of the flight path that do not see any targets. This is also due to the algorithm not being able to predict future possibilities. It simply chooses the next best position and steering angles. From some of these positions and steering angles, there is no way for the aircraft to see a target.

The current greedy algorithm works well most of the time. However, there are drawbacks to using this algorithm. Ultimately, we want to use an algorithm that considers future possibilities when it assigns a specific flight position and steering angles. Thus, we implement some additional algorithms in order to better optimize the flight path.

We used some of the same scheduling algorithms as in Chapter 2 with the Space Domain Awareness sensor tasking problem. Namely, in addition to the already implemented greedy algorithm, we implement a random search algorithm and a genetic algorithm. For all of the algorithms implemented, we use all of the same constraints outlined in Section 3.2.3.

5.1 Block Greedy and WTA Algorithms

We did not implement a weapon-target-assignment (WTA) algorithm. Recall, this algorithm is performed by finding all possible options for the aircraft to move and steer its antenna across a designated block of waypoints. For more detail on this algorithm, see Chapter 2. In Chapter 2, we saw that this algorithm was extremely computationally intensive. It performed well in terms of generating a good schedule. However, the trade off between optimality and run time with this algorithm made it less than desirable. Be-

low is pseudo-code of this algorithm:

Result: *Regionally Optimal Flight Path*

define waypoint block size;

for *(waypoints / block size)* **do**

  for *waypoint blocks* **do**
    find all possible tasks per waypoint

  end

  compute every combination of tasks for waypoint block;

  choose highest scoring combination block;

end

return *WTA path*;

Algorithm Weapon-Target-Assignment Algorithm

The complexity of this algorithm lies at the line '*compute every combination of tasks for waypoint block*'. For example, let m_w, n_w, p_w, q_w be the number of possible headings, pitches, and steering angles for waypoint w. Then the total possible combination of flight paths is the following:

$$total\ paths\ = \prod_w m_w n_w p_w q_w. \tag{5.1}$$

Suppose there are 20 possibilities of each heading, pitch, and steering angles for each waypoint. Also suppose there are 50 waypoints. Then there would be $(20^4)^{50} \approx 1.606938 \times 10^{260}$ total possible paths. As in Chapter 2, this is too many possibilities to compute the global optimum by brute force.

In the WTA algorithm, we break this into smaller pieces so that instead of computing this combination over 50 waypoints, we compute this over a smaller block of waypoints. However, even breaking this into smaller waypoint blocks for the WTA algorithm is highly computational. Thus, we did not apply this algorithm for this application.

Additionally, we did not implement a block greedy algorithm in this case. In the block greedy algorithm in Chapter 2, we computed the possible look angles for each sensor at each time window in the time window blocks, as opposed to computing the possible combination of look angles for each sensor across that time window block. In the SDA case, we knew the location of the sensor at each time based on formulas for

propagation. Thus, we knew how many targets the sensors would see for each look angle for each time. This was important since we scheduled sensor look angles out of temporal order. We chose the highest scoring look angle for each sensor out of order of time in each time block.

In the flight path case, we could compute all possible headings, pitches, and steering angles for each waypoint in a block of waypoints. However, without knowing what the previous position of the aircraft is, we will not know where the next heading and pitch places the aircraft. Thus, we will not know which targets or how many targets it is able to see using that heading, pitch, and steering angles. Without knowing this, we will not know which heading, pitch, and steering angles produces the highest score. So we will not know which to schedule out of order of the flight path waypoints.

Recall the following description of the block greedy algorithm:

Figure 5.1: Block Greedy Algorithm

In this case, the tasks consist of 4-tuples of headings, pitches, and steering angles. In order to schedule the highest scoring task, we need to know where the aircraft is so we know how many targets lie in its antenna footprint. However, without knowing where the aircraft was at the previous time, we have no way of knowing where the 4-tuple will place the aircraft. Thus, we have no way of knowing what targets the aircraft is able to see and are not able to compute the highest scoring task out of flight path order.

In the next sections, we will compare a genetic algorithm and random search algorithm with our existing greedy algorithm. We used the objective function for all of these algorithms along with the constraints

detailed in Section 3.2.3. Since each of these algorithms contain a random component, we ran each of these algorithms multiple times to detect the overall trend. In the figures, we show the best flight generated after running the algorithms multiple times. When we compare the score and number of targets seen for each algorithm, we compare the averages of these values from these multiple runs.

5.2 Greedy Algorithm

The greedy algorithm implemented here is the same algorithm described in Section 3.3. The best flight path generated after these multiple runs with the greedy algorithm is shown in figures 5.2 and 5.3.

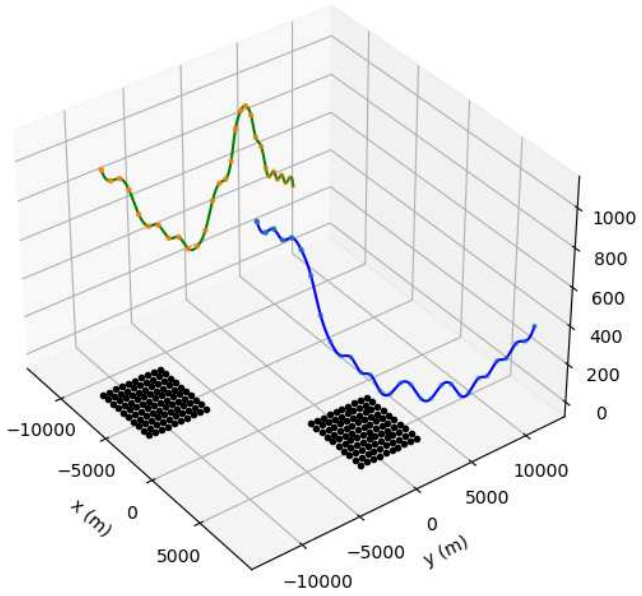

Figure 5.2: Greedy flight path

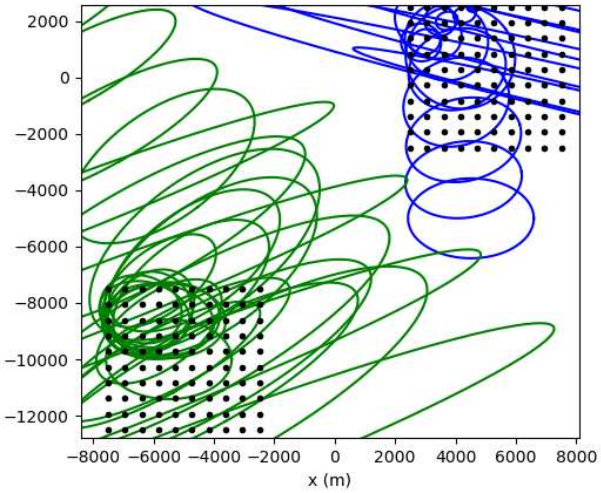

Figure 5.3: Antenna footprints from greedy flight path

5.3 Random Search

In the random search algorithm, at each waypoint, this algorithm randomly chooses each UAV's next heading, pitch, and steering angles. The pseudo-code for this algorithm is below:

Result: *Random Flight Path*

initialize empty path;

for *waypoints* **do**

> *choose random set of heading, pitch, steering angles*;
>
> *compute aircraft position resulting from random set*;
>
> *add aircraft position to waypoint*;

end

return *random path*;

Algorithm Random Search Algorithm

This algorithm was run many times. Figures 5.4 and 5.5 show the highest scoring schedule after these runs. Since this algorithm is entirely random, figures 5.6 and 5.7 show an additional assortment of these random algorithm runs.

Recall, the randomly chosen parameters are the aircraft's heading, pitch, and steering angles. We can see in these figures that none of the flights or antenna footprints look the same. It should be noted that all of these parameters are constrained to realistic aircraft movements. Thus, the flights appear relatively smooth.

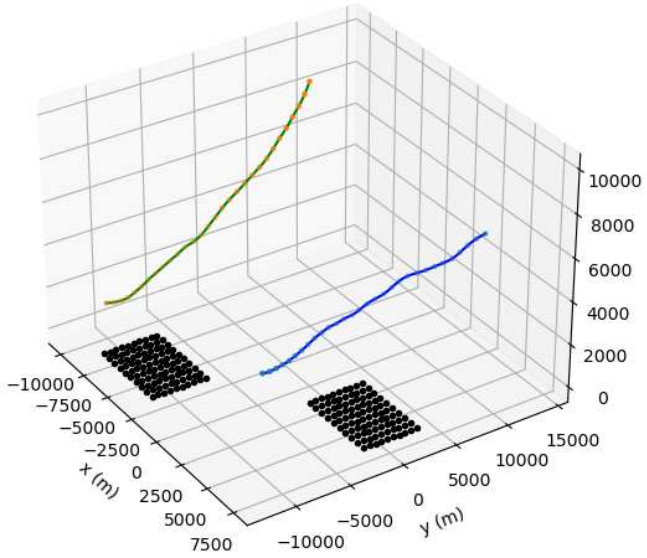

Figure 5.4: Randomly generated flight paths for two UAVs

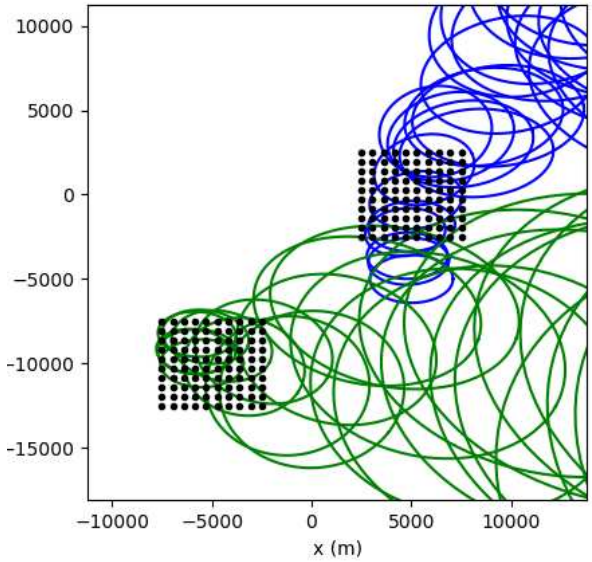

Figure 5.5: Corresponding antenna footprints

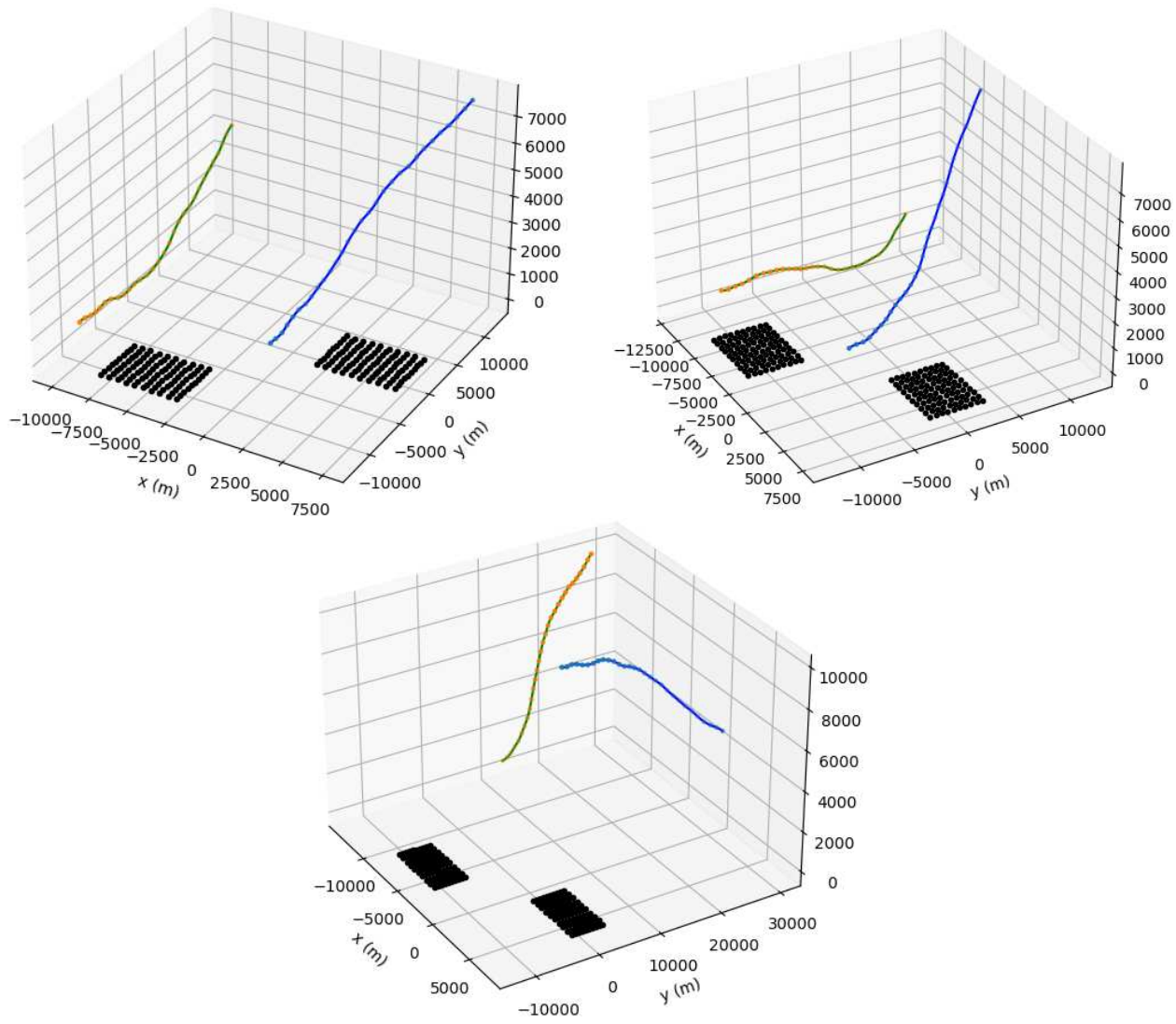

Figure 5.6: An assortment of randomly generated flight paths

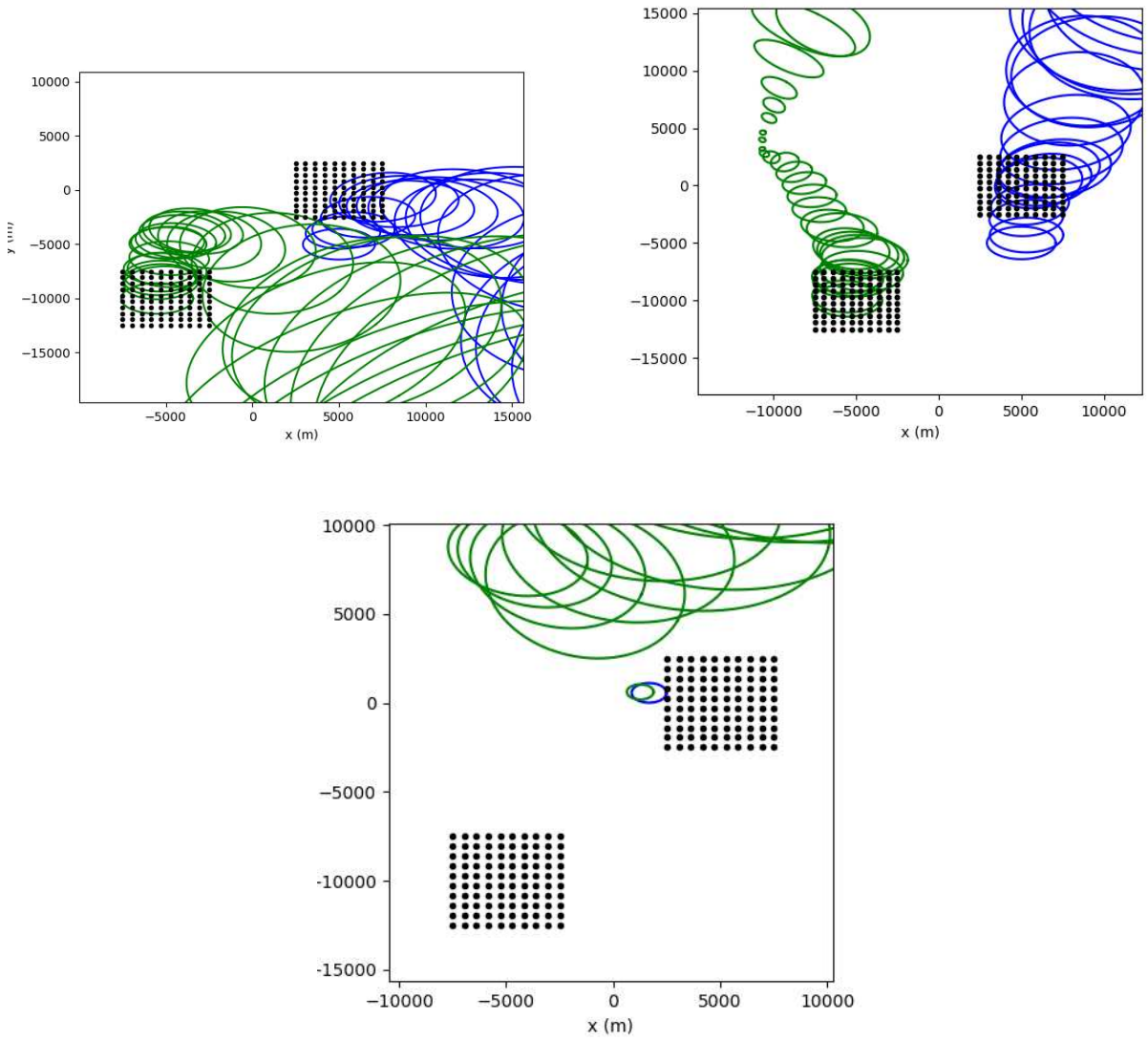

Figure 5.7: The corresponding footprints of the flight paths in the figure above

5.4 Genetic Algorithm

After implementing a random search algorithm, we looked at a genetic algorithm. This algorithm behaves the same way as in Chapter 2. Recall from Chapter 2, the genetic algorithm consists of the following steps: initialize population, crossover for each generation, and mutate for each generation. The pseudo-code for this algorithm is below:

Result: *Genetic Flight Path*
generate initial population;
for *generations* **do**
    crossover population;
    mutate population;
end
path = max crossed and mutated population;
return *genetic path*;

Algorithm Genetic Algorithm

In this scenario, a population consists of a set of heading, pitch, and steering angles for each waypoint. The first step in the genetic algorithm is to randomly choose n populations to initialize. After this, we crossover and mutate this population for m generations.

In order to perform the crossover step, random population pairs are generated until each population in a pair satisfies a specific amount of fitness. Namely, the scores of the populations must be greater than a set threshold. Once a satisfactory population pair is chosen, they crossover to form a new population. This is done by randomly choosing waypoints or waypoint groups from each population pair to use in the new population until the new population has a heading, pitch, and steering angle for each waypoint.

After the crossover step, a given percent of waypoints are mutated. This is done by randomly selecting these waypoints to be mutated. We randomly select a set of heading, pitch, and steering angles to replace the existing heading, pitch, and steering angles.

After repeating the crossover and mutated steps over the m generations, we choose the highest scoring path from the latest population set. See Chapter 2 for more detail on the genetic algorithm.

Figures 5.8 and 5.9 are results from using the genetic algorithm.

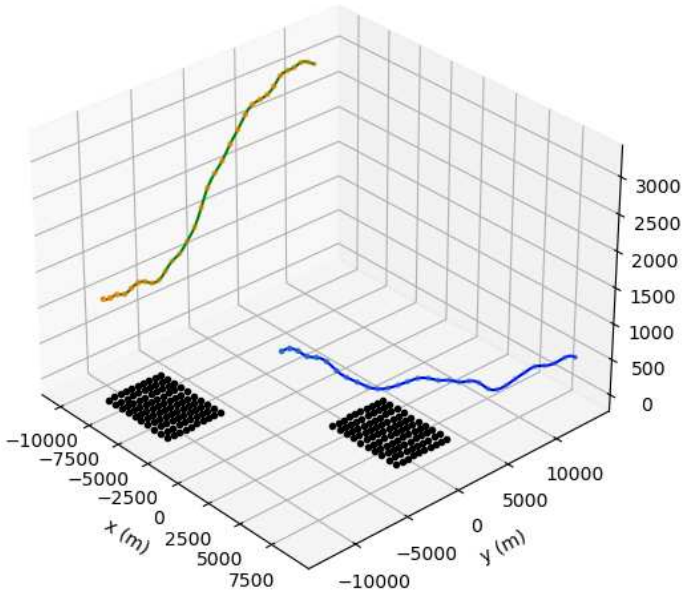

Figure 5.8: Flight produced from a genetic algorithm.

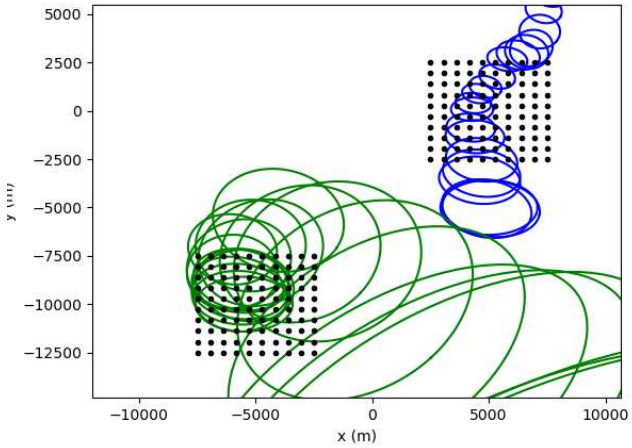

Figure 5.9: Antenna footprints from genetic algorithm generated flight path.

In summary, we have implemented a greedy, random search, and genetic algorithm on this flight path planning problem. In figure 5.10, we see how these algorithms compare based on the objective function score achieved. As we can see from this result, the genetic algorithm has the highest objective function score. Thus, it generates the best flight path out of all of the algorithms implemented.

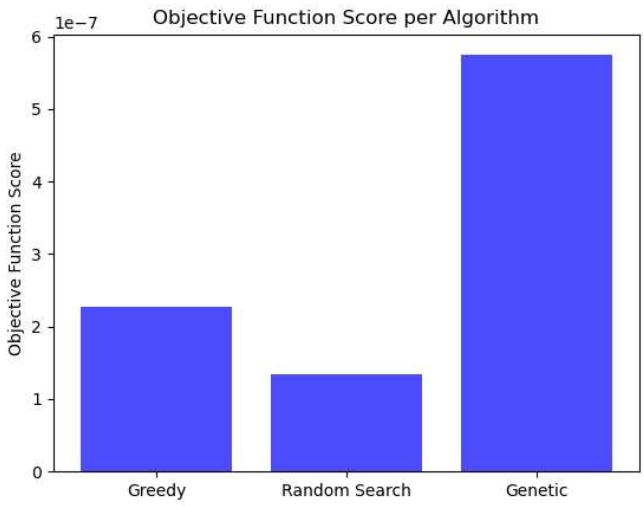

Figure 5.10: Objective function score by algorithm

5.5 Conclusion and Future Work

In Chapter 3, we followed the logic from Altrichter [10] to develop an innovative objective function to maximize image resolution and scene coverage. The results from this objective function were compared with the results from common SAR flight paths. Namely, we compared our flight path with a stripmap mode, spotlight mode, and circular mode. Ultimately, the flight generated from the objective function outperformed the other flight paths.

We then extended this work by first optimizing against the aircraft's antenna steering angles in addition to its heading and pitch. This created a more realistic flight path planning scenario and generated a more robust SAR flight path planning ability. We were able to obtain a higher objective function value by optimizing against these additional parameters.

In Chapter 4, we explored many applications and extensions that may be extrapolated from our formulated objective function. We first added an SNR consideration to our flight path. We were able to include this in our objective function. This kept our aircraft from flying indefinitely upward.

After this, we added target priority and target scene priority. As we expected, this led to a flight path that focused specifically on higher priority targets or target scenes. This added an ideal flexibility to how we searched an area.

We then used our objective function to generate flight paths for multiple vehicles. We specifically looked at flight path planning for a collection of multiple, mono-static UAVs. The objective function was instrumental in creating an optimal flight path for multiple UAVs. This work could be useful to flight path planners for platforms carrying SAR systems.

Lastly, in Chapter 5, we analyzed the outcome of maximizing the objective function using various optimization algorithms. In Chapters 3 and 4, we used a greedy algorithm to maximize the objective function. In this chapter, we also implemented a random search algorithm and a genetic algorithm. Ultimately, we found that the genetic algorithm produced the best flight path out of all of the algorithms we implemented.

We explored many extensions and applications of using this objective function. There are many more possibilities. Some future work may include creating a more robust flight path for UAVs. As the scenario becomes more realistic, more constraints must be considered. Some additional considerations may be to keep the UAVs from flying into each other's antenna beams. In most of the scenarios we tested, the UAVs looked at different targets so this concern was not applicable. However, there are many situations in which this is an important consideration. It may also be important to ensure the UAVs fly a certain distance apart. These modifications are all possible given the flexibility of our current framework.

Additional future work with UAVs might include gathering 3D information on a target scene. In order to generate a flight path that considers this 3D information, we may modify the objective function to include the 3D component of the data-collection manifold. See Appendix D for a derivation of the objective function using the 3D component of the data-collection manifold.

Ultimately, this objective function has been found to be incredibly versatile. We have shown many possible extensions and applications for this objective function. These are just a few of many possibilities. This objective function opens many avenues for future research.

Bibliography

[1] E. Mazareanu. Number of satellites launched by year 2019. *Statista*, Jun 2020.

[2] Thomas G. Roberts and Todd Harrison. Popular orbits 101. *Aerospace Security*, Oct 2020.

[3] David A. Vallado. *Fundamentals of astrodynamics and applications*. The space technology library ; v. 12. Microcosm Press, El Segundo, Calif, 2nd ed. edition, 2001.

[4] Phoenix. Earth fixed - polar view. *Wikipedia*, 2021.

[5] Orbital frames of reference. *Space Universities Network*, 2015.

[6] Mission Data Processing team at Ball Aerospace. 2020.

[7] Margaret Cheney and Brett Borden. *Fundamentals of radar imaging*. SIAM, 2009.

[8] Margaret Cheney and Brett Borden. Synthetic aperture radar imaging. *Handbook of Mathematical Methods in Imaging*, page 763–799, 2015.

[9] Seok Kim, Jiwoong Yu, Se-Yeon Jeon, Aulia Dewantari, and Min-Ho Ka. Signal processing for a multiple-input, multiple-output (mimo) video synthetic aperture radar (sar) with beat frequency division frequency-modulated continuous wave (fmcw). *Remote Sensing*, 9(5):491, 2017.

[10] Scott Altrichter. *Flight path optimization for resolution and coverage in synthetic aperture radar*. PhD thesis, Rensselaer Polytechnic Institute, 2015.

[11] K. Sundar and S. Rathinam. An exact algorithm for a heterogeneous, multiple depot, multiple traveling salesman problem. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 366–371, 2015.

[12] NASA Official: Brian Dunbar. Sputnik: the dawn of the space age. *NASA 60 Years & Counting*, Jan 2021.

[13] Carolin Frueh, Hauke Fielder, and Johannes Herzog. Heuristic and optimized sensor tasking observation strategies with exemplification for geosynchronous objects. *Journal of Guidance, Control, and Dynamics*, 41(5):1036–1048, 2018.

[14] Olfa Dridi, Saoussen Krichen, and Adel Guitouni. A multi-objective optimization approach for resource assignment and task scheduling problem: Application to maritime domain awareness. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

[15] Christopher G Valicka, Deanna Garcia, Andrea Staid, Jean-Paul Watson, Mark Daniel Rintoul, Gabriel Anton Hackebeil, and Lewis Ntaimo. Sensor network scheduling under uncertainty: Models and benefits. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2016.

[16] R Scott Erwin, Paul Albuquerque, Sudharman K Jayaweera, and Islam Hussein. Dynamic sensor tasking for space situational awareness. In *Proceedings of the 2010 American Control Conference*, pages 1153–1158. IEEE, 2010.

[17] Bryan D Little and Carolin Frueh. Ssa sensor tasking: Comparison of machine learning with classical optimization methods. In *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference*, pages 1–17, 2018.

[18] Bryan D Little. *Optical Sensor Tasking Optimization for Space Situational Awareness*. PhD thesis, Purdue University, 2019.

[19] Michael S Felten. Optimization of geosynchronous space situational awareness architectures using parallel computation. *Air Force Institute of Technology*, 2018.

[20] Andreas Hinze, Hauke Fiedler, and Thomas Schildknecht. Optimal scheduling for geosynchronous space object follow-up observations using a genetic algorithm. In *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*. Maui Economic Development Board Maui, HW, 2016.

[21] Naomi Owens Fahrner. A regional greedy algorithm for space domain awareness resource allocation. *AMOS*, 2021.

[22] Synthetic aperture radar: Round the clock reconnaissance. *Lockheed Martin*, n.d.

[23] Wireless Public Safety Networks 1. Synthetic aperture radar. *ScienceDirect*, 2015.

[24] The electromagnetic spectrum. *Boundless Physics*, 2014.

[25] BD Rigling and RL Moses. Flight path strategies for 3-d scene reconstruction from bistatic sar. *IEEE Proceedings-Radar, Sonar and Navigation*, 151(3):149–157, 2004.

[26] Fanyun Xu, Rufei Wang, Lu Zhao, Yongchao Zhang, Yin Zhang, Yulin Huang, and Jianyu Yang. Uav intelligent optimal path planning method for distributed radar short-time aperture synthesis. In *IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium*, pages 469–472. IEEE, 2020.

[27] Jincheng Li, Jie Chen, Pengbo Wang, and Chunsheng Li. Sensor-oriented path planning for multiregion surveillance with a single lightweight uav sar. *Sensors*, 18(2):548, 2018.

[28] Markus Schartel, Ralf Burr, Winfried Mayer, Nando Docci, and Christian Waldschmidt. Uav-based ground penetrating synthetic aperture radar. In *2018 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, pages 1–4. IEEE, 2018.

[29] Dipl.-Ing. (FH) Christian Wolff. Airborne radars. *Radartutorial*.

[30] Wojciech Stecz and Krzysztof Gromada. Uav mission planning with sar application. *Sensors*, 20(4):1080, 2020.

[31] Sophie Lentilhac. Uav flight plan optimzed for sensor requirements. *IEEE Aerospace and Electronic Systems Magazine*, 25(2):11–14, 2010.

[32] Fanyun Xu, Rufei Wang, Yongchao Zhang, Yin Zhang, Yulin Huang, and Jianyu Yang. Uav intelligent optimal path planning method for distributed radar short-time aperture synthesis. *IEEE Xplore*, 2020.

[33] Sgp4. *Python Package Index*, 2020.

[34] Tarunraj Singh. Optimal sensor tasking for space situational awareness. Technical report, RESEARCH FOUNDATION OF STATE UNIVERSITY OF NEW YORK THE BUFFALO United States, 2020.

[35] Edwin KP Chong and Stanislaw H Zak. *An introduction to optimization*. John Wiley & Sons, 2004.

[36] Margaret Cheney. Imaging frequency-dependent reflectivity from synthetic-aperture radar. *Inverse Problems*, 29(5):054002, 2013.

[37] Foley. Optimal algorithm development for uhf circular synthetic aperture radar (csar) applications. 2009.

[38] Mark E Davis. Developments in foliage penetration radar. In *2009 International Radar Conference" Surveillance for a Safer World"(RADAR 2009)*, pages 1–6. IEEE, 2009.

[39] Michael C Wicks, Braham Himed, JLE Bracken, Harry Bascom, and John Clancy. Ultra narrow band adaptive tomographic radar. In *1st IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, 2005.*, pages 36–39. IEEE, 2005.

[40] Joseph Schneible, James Paul Browning, Michael Wicks, and Yuhong Zhang. Geometric diversity versus frequency diversity an imaging example. In *2010 International Waveform Diversity and Design Conference*, pages 262–269. IEEE, 2010.

[41] LJ Cutrona. Synthetic aperture radar. *Radar handbook*, 2:2333–2346, 1990.

[42] Yee Kit Chan and Voon Koo. An introduction to synthetic aperture radar (sar). *Progress In Electromagnetics Research B*, 2:27–60, 2008.

[43] Samuel W McCandless and Christopher R Jackson. Principles of synthetic aperture radar. *Synthetic aperture radar marine user's manual*, pages 1–24, 2004.

[44] Mehrdad Soumekh. *Synthetic aperture radar signal processing*, volume 7. New York: Wiley, 1999.

[45] BR Mahafza and A Elsherbeni. High resolution tactical synthetic aperture radar, 2004.

[46] GA Showman. An overview of radar imaging. *Principles of Modern Radar*, 1:835–891, 2010.

[47] Scheer Cook, Melvin. Spotlight synthetic aperture radar. 2013.

[48] Modes & frequency bands of operation. *Sandia National Laboratory*, 2021.

[49] Aquisition modes. *The European Space Agency*, 2000-2021.

[50] Jeanne J. Plot ellipse (python). *Stack Overflow*, May 2018.

[51] Armin W Doerry. Performance limits for synthetic aperture radar. Technical report, SANDIA NATIONAL LABS ALBUQUERQUE NM, 2006.

[52] Nova, spies that fly, time line of uavs. *PBS*, 2002.

[53] Huai Yu, Tianheng Yan, Wen Yang, and Hong Zheng. An integrative object-based image analysis workflow for uav images. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLI-B1:1085–1091, 2016.

[54] P Richard et al. *Feynman lectures on physics*. Addinson-Wesley, 1963.

[55] Frank Lee. The intuitive derivation of maxwell's equations. *WordPress*, Apr 2016.

[56] Maxwell's equations. *Wikipedia*, Feb 2021.

[57] Bevelacqua. Maxwell's equations. *Maxwell's Equations*.

[58] Los Angeles University of California. Gauss' law. *UCLA Physics & Astronomy*.

Appendix A

Erf Function

The erf function is the error function. The error function is defined as the following:

$$erf(z) = \frac{2}{\sqrt{\pi}} \int_0^z \exp(-t^2) dt. \tag{A.1}$$

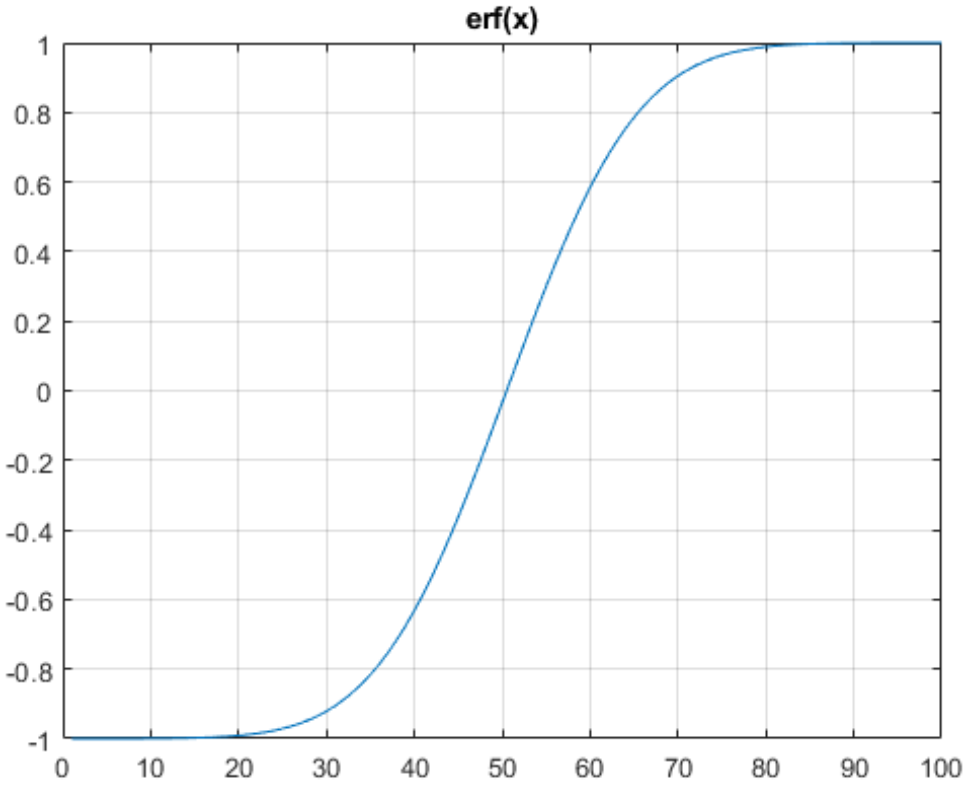

Figure A.1: Erf Function

Appendix B

Maxwell's Equations Derivation

(The following work is a combined effort of Naomi Owens-Fahrner and Colin Roberts, with references [54] [55] [56] [57] [58])

The equations we want to derive are the following:

Gauss' Law for Electric Fields:	$\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_0}$
Gauss' Law for Magnetic Fields:	$\nabla \cdot \mathbf{B} = 0$
Faraday's Law:	$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$
Ampere's Law:	$\nabla \times \mathbf{B} = \mu_0 \left(\mathbf{J} + \epsilon_0 \frac{\partial \mathbf{E}}{\partial t} \right).$

We can see that Faraday's Law and Ampere's Law describe the coupling between the electric field $\mathbf{E}$ and magnetic field $\mathbf{B}$ fields. Both of Gauss' Laws describe the nature of the sources for the two fields.

B.1 Derivation

To derive the following equations, we pose some physical requirements on the nature of these fields. First, we claim that these fields are sourced from charges q and the motion of these charges. We also require that these fields are additive in the sense that the field produced by a charge q permeates space without the knowledge that another charge exists or not. This is to say, for example, that if we have a charge q_1 that generates an electric field $\mathbf{E}_1$ and a charge q_2 that generates a field $\mathbf{E}_2$ that the resulting field is given by

$$\mathbf{E} = \mathbf{E}_1 + \mathbf{E}_2.$$

The same quality holds true for magnetic fields and for distributions of charges. This allows us to consider the case with a single charge as we can superimpose the solutions for a collection of charges later on. Given this, we expect our equations to be linear.

These equations were not originally *derived*. Instead, they were formulated to match results from experimentation. One particular observation is that there is no fundamental source for the magnetic field. In other words, we have charges that create sources for electric fields, but there is no notion of a “magnetic charge." Often, we say that there are no magnetic monopoles.

It is important to remember that these are dynamical equations and charges do influence motion of other charges. One must state whether charges are to be fixed in place or are free to move. An example would be a fixed nucleus of an atom and an orbiting electron.

B.1.1 Gauss' Law for Electric Fields

Fix a charge q in $\mathbb{R}^3$. We can enclose this charge with an arbitrary closed surface S. This charge creates a field at all points in space which we can measure indirectly by measuring the flux $\Phi_{\mathbf{E}}$ of this field through our surface.

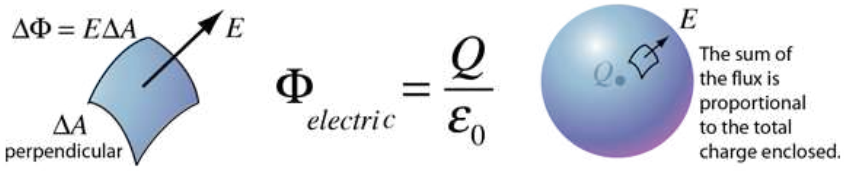

Figure B.1: A helpful picture to keep in mind for Gauss' law.

From experiment, it was observed that the total flux through S is proportional to the charge enclosed. Thus we can write

$$\Phi_{\mathbf{E}} = \iint_S \mathbf{E} \cdot \hat{\mathbf{n}} dS = kq.$$

If the charge was located on the exterior to S, then the total flux through S would be zero. This means, we care only about the distribution of charges inside S and thus if we instead had a charge distribution $\rho(\mathbf{x})$ contained inside S we could write

$$q = \iiint_V \rho(\mathbf{x}) dV.$$

With the relationship that $S = \partial\Omega$, we can apply the divergence theorem to arrive at

$$\iiint_V \nabla \cdot \mathbf{E} dV = k \iiint_V \rho dV.$$

Since this was true for our arbitrary surface S (and so an arbitrary region Ω), we must have that

$$\nabla \cdot \mathbf{E} = k\rho.$$

From experiment, it turns out that $k = \frac{1}{\epsilon_0}$ where ϵ_0 is the permittivity of free space. It seems that charges are fundamentally a source (or possibly sink) for electric field. The equation above is describing the outflow of a field that is based on the charge.

B.1.2 Gauss' Law for Magnetic Fields

This law seems to be more of a physical postulate. In fact, it is one that some physicists are still trying to find counterexamples to! Since we believe there are no fundamental magnetic charges (magnetic monopoles), we perform the exact argument as before but ρ must be identically zero. Hence,

$$\nabla \cdot \mathbf{B} = 0.$$

B.1.3 Faraday's Law

Through experimentation, Faraday noticed that changing the magnetic flux in a circuit produced an electromotive force (EMF). The EMF in an electric circuit is the electric potential between two points. This is measured in volts, so we will call the EMF V. Faraday described his observation by

$$V = -\frac{d\Phi_B}{dt} \tag{B.1}$$

where Φ_B represents the magnetic flux. Now we will derive the various pieces in order to better describe (1).

The magnetic flux is the amount of **B** passing through an arbitrary surface normal to **B**. We will call this surface S. In other words,

$$\Phi_B = \iint_S (\mathbf{B} \cdot \mathbf{n}) dS \tag{B.2}$$

This gives us information about the right hand side of the equation. For the left side of the equation, we need to determine the EMF. Since the EMF is the electric potential between two points, the total EMF is the

sum of the **E** field times the path between the points. We can write this by

$$V = \int_C \mathbf{E} \cdot \mathbf{t} dL, \tag{B.3}$$

where $\mathbf{t}$ is the unit tangent vector to our curve C. Using Stoke's Theorem,

$$V = \iint_S (\nabla \times \mathbf{E}) \cdot \mathbf{n} dS. \tag{B.4}$$

Substituting equations (2) and (4) into equation (1), we get

$$\iint_S \left(\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t} \right) \cdot \mathbf{n} dS = 0. \tag{B.5}$$

Since we have an integral over a finite area equalling zero, the integrand also equals zero. Thus we have

$$\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t} = 0. \tag{B.6}$$

This in turn yields Faraday's Law. Namely,

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}. \tag{B.7}$$

B.1.4 Ampere's Law

Again, through experimentation, Maxwell proposed that the strength of the magnetic field is proportional to the electric current, I. Specifically, the amount of magnetic field along a loop surrounding a wire with current I will be given by

$$\int_C \mathbf{B} \cdot \mathbf{t} dL = \mu_0 I.$$

We can then use Stokes' theorem to arrive at

$$\mu_0 I = \iint_S (\nabla \times \mathbf{B}) \cdot \mathbf{n} dS$$

where S is an arbitrary area bounded by the closed loop C.

Current I can be found by looking at the flux of the current density $\mathbf{J}$ through our surface S. That is to say,

$$\mu_0 I = \mu_0 \iint_S \mathbf{J} \cdot \mathbf{n} dS.$$

Given this, we would arrive at

$$\iint_S (\nabla \times \mathbf{B} + \mu_0 \mathbf{J}) \cdot d\mathbf{S} = 0.$$

Since S is arbitrary, we can safely say that

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J}. \tag{B.8}$$

However, this is not the exact equation we wanted. It turns out, by observation, Ampere noticed that the magnetic field from a capacitor in an AC circuit would vary. Current was not passing through the capacitor, but the plates were charging. Thus, he realized that the changing charge (and therefore a changing electric field) must also affect the magnetic field.

Recall that current is also the change in charge over time

$$I = -\frac{\partial q}{\partial t}.$$

This leads us to

$$\iint_S \mathbf{J} \cdot \mathbf{n} dS = -\iiint_\Omega \frac{\partial \rho}{\partial t} dV.$$

The divergence theorem then yields

$$\iiint_V (\nabla \cdot \mathbf{J}) dV = -\iiint_V \frac{\partial \rho}{\partial t} dV.$$

Our S was chosen arbitrarily, and so Ω is as well. Therefore

$$\nabla \cdot \mathbf{J} = -\frac{\partial \rho}{\partial t}.$$

From previous work, we have

$$\rho = \epsilon_0(\nabla \cdot \mathbf{E})$$

and thus

$$\nabla \cdot \mathbf{J} = -\epsilon_0 \left(\nabla \cdot \frac{\partial \mathbf{E}}{\partial t} \right).$$

Now, the cross product $\mathbf{u} \times \mathbf{v} = \mathbf{w}$ will always be orthogonal to both $\mathbf{u}$ and $\mathbf{v}$. In particular, we have

$$\nabla \cdot (\nabla \times \mathbf{B}) = 0.$$

Applying this to (8) shows us that we are missing a term! We must actually have that

$$\nabla \times \mathbf{B} = \mu_0 \left(\mathbf{J} + \epsilon_0 \frac{\partial E}{\partial t} \right),$$

so that the cross product identity shown above *always* holds.

B.1.5 Boundary Conditions

The fields generated from charges and currents permeate all of space. With that, we do not expect to assign boundary conditions on a given spatial domain. Instead, we impose charge densities and current densities and the resulting fields can be computed from there. These densities can vary over time but the fields that arise exist at all points in space. This is all to say that we work on $\Omega = \mathbb{R}^3$.

Appendix C

Data-Collection Manifold Derivation

We will follow the the data-collection manifold derivation in [7, 10]. In the phase of equation (3.23), we apply a change of variables and we get the first term of the Taylor expansion as the following:

$$2k(|R_{s,\boldsymbol{z}}| - |R_{s,\boldsymbol{x}}|) = (\boldsymbol{z} - \boldsymbol{x}) \cdot \frac{2\omega}{c} \hat{R}_{s,\boldsymbol{z}}. \tag{C.1}$$

Applying the Stolt change of variables, we obtain

$$(s, \omega) \longrightarrow \boldsymbol{\xi} = \frac{2\omega}{c} \hat{R}_{s,\boldsymbol{z}}, \tag{C.2}$$

which gives us the PSF with the desired phase:

$$K(\boldsymbol{z}, \boldsymbol{x}) = \int e^{i(\boldsymbol{z}-\boldsymbol{x})\cdot\boldsymbol{\xi}} Q(\boldsymbol{\xi}, \boldsymbol{z}) A(\omega(\boldsymbol{\xi}), s(\boldsymbol{\xi}), \boldsymbol{x}) \left| \frac{\partial(s, \omega)}{\partial \boldsymbol{\xi}} \right| d\boldsymbol{\xi}. \tag{C.3}$$

We choose a filter Q such that eq. (C.3) is as close to the δ-function as possible. Then (C.3) becomes the following:

$$\int_{\Omega_{\boldsymbol{z}}} e^{i(\boldsymbol{z}-\boldsymbol{x})\boldsymbol{\xi}} d\boldsymbol{\xi}, \tag{C.4}$$

where the data-collection manifold is given by (3.25).

Appendix D

Objective Function using 3D DCM

The following derivation comes from Dr. Scott Altrichter's PhD Dissertation [10]. If we consider the DCM with the vertical component, we get the following parametrization:

$$\begin{aligned}\boldsymbol{\xi}(\omega, s) &= [\xi_1, \xi_2, \xi_3] \\ &= \frac{2\omega}{c}[\cos\theta(s)\cos\psi(s), \sin\theta(s)\cos\psi(s), \sin\psi(s)],\end{aligned} \tag{D.1}$$

for $\omega \in [\omega_{min}, \omega_{max}]$ and $s \in [s_i, s_f]$.

We use multivariable calculus tools in order to find the surface element of the data-collection manifold associated with each target z. This is found by computing the magnitude of the cross product of $\boldsymbol{\xi_\omega}$ and $\boldsymbol{\xi_s}$

$$\begin{aligned}\boldsymbol{\xi_\omega} \times \boldsymbol{\xi_s} &= \begin{vmatrix} \hat{\boldsymbol{i}} & \hat{\boldsymbol{j}} & \hat{\boldsymbol{k}} \\ \frac{\partial\xi_1}{\partial\omega} & \frac{\partial\xi_2}{\partial\omega} & \frac{\partial\xi_3}{\partial\omega} \\ \frac{\partial\xi_1}{\partial s} & \frac{\partial\xi_2}{\partial s} & \frac{\partial\xi_3}{\partial s} \end{vmatrix} \\ &= \left(\frac{\partial\xi_2}{\partial\omega}\frac{\partial\xi_3}{\partial s} - \frac{\partial\xi_3}{\partial\omega}\frac{\partial\xi_2}{\partial s}\right)\hat{\boldsymbol{i}} \\ &\quad - \left(\frac{\partial\xi_1}{\partial\omega}\frac{\partial\xi_3}{\partial s} - \frac{\partial\xi_3}{\partial\omega}\frac{\partial\xi_1}{\partial s}\right)\hat{\boldsymbol{j}} \\ &\quad + \left(\frac{\partial\xi_1}{\partial\omega}\frac{\partial\xi_2}{\partial s} - \frac{\partial\xi_2}{\partial\omega}\frac{\partial\xi_1}{\partial s}\right)\hat{\boldsymbol{k}}\end{aligned} \tag{D.2}$$

$$\begin{aligned}\Rightarrow ||\boldsymbol{\xi_\omega} \times \boldsymbol{\xi_s}|| &= \sqrt{\left(\frac{\partial(\xi_2, \xi_3)}{\partial(\omega, s)}\right)^2 + \left(\frac{\partial(\xi_1, \xi_3)}{\partial(\omega, s)}\right)^2 + \left(\frac{\partial(\xi_1, \xi_2)}{\partial(\omega, s)}\right)^2} \\ &= \frac{4\omega}{c^2}\sqrt{(\psi'(s))^2 + (\theta'(s))^2\cos^2(\psi(s))}.\end{aligned} \tag{D.3}$$

Now, in order to formulate an expression for the surface area of the DCM at target z, we compute the following:

$$
\begin{aligned}
A(\Omega) &= \int_{s_i}^{s_f} \int_{\omega_{min}}^{\omega_{max}} ||\boldsymbol{\xi_\omega} \times \boldsymbol{\xi_s}|| d\omega ds \\
&= 2(k_{max}^2 - k_{min}^2) \int_{s_i}^{s_f} \sqrt{(\psi'(s))^2 + (\theta'(s))^2 \cos^2(\psi(s))} ds.
\end{aligned}
\tag{D.4}
$$

Finally, in order to consider all target DCM, we integrate over all targets within the aircraft's antenna footprint. We are left with the following objective function:

$$
V = 2(k_{max}^2 - k_{min}^2) \int_{\boldsymbol{Z}} \int_{s_i}^{s_f} \sqrt{(\psi'(s))^2 + (\theta'(s))^2 \cos^2(\psi(s))} ds d\boldsymbol{z}, \ \forall \boldsymbol{z} \in \boldsymbol{Z}. \tag{D.5}
$$

List of Abbreviations

- CDF = Cumulative Distribution Function
- DCM = Data-Collection Manifold
- ECEF = Earth Centered Earth Fixed
- ECI = Earth Centered Inertial coordinate frames
- EW = Extra Wide Swath
- FOV = Field of View
- GEO = Geosynchronous Earth orbit
- IFT = Inverse Fourier Transform
- IOD = In Orbit Demonstration
- IW = Interferometric Wide Swath
- LEO = Low Earth Orbit
- MEO = Medium Earth Orbit
- PRF = Pulse Repetition Frequency
- PSF = Point Spread Function
- RSO = Resident Space Object
- SAR = Synthetic Aperture Radar
- SDA = Space Domain Awareness
- SDARA = Space Domain Awareness Resource Allocation
- SNR = Signal-to-Noise Ratio
- TLE = Two-line Element Set

- TSP = Traveling Salesperson Problem
- UAV = Unmanned Aerial Vehicle
- WTA = Weapon-Target-Assignment Algorithm