A Genetic Algorithm Framework using Variable Length Chromosomes for Vehicle Maneuver Planning

by

Benjamin James Yu

B.S. Tufts University (2011)

Submitted to the The Center for Computational Science and Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Computational Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2022

Author .....
The Center for Computational Science and Engineering
May 10, 2022

Certified by .....
Olivier de Weck
Apollo Program Professor of Astronautics and Engineering Systems
Thesis Supervisor

Accepted by .....
Youssef Marzouk
Professor of Aeronautics and Astronautics
Co-Director, Center for Computational Science and Engineering

A Genetic Algorithm Framework using Variable Length Chromosomes for Vehicle Maneuver Planning

by

Benjamin James Yu

Submitted to the The Center for Computational Science and Engineering
on May 10, 2022, in partial fulfillment of the
requirements for the degree of
Master of Science in Computational Science and Engineering

Abstract

Incorporating reconfigurability demonstrates great potential in increasing the performance and/or lowering the cost of complex systems. Reconfigurability enables a system to adapt and dynamically respond to the specific objectives it encounters, rather than simply being optimized towards a general case. One such class of reconfigurable systems are fleets of maneuvering vehicles. Considering this class naturally leads to the question of how to generate the optimal set of maneuvers over an operational campaign. This thesis presents a genetic algorithm framework with Variable Length Chromosomes (VLC) to find this optimal set of maneuvers. Said framework generates Pareto optimal sets of maneuvers using non-dominated sorting genetic algorithm II (NSGA-II). The use of VLC removes the necessity for a human designer to impose a priori assumptions on the number and/or timing of vehicle maneuvers. Instead, the optimizer is freed to grow or reduce the number of maneuvers as needed. In addition, the use of a genetic algorithm approach enables the framework to evaluate problem domains and constraints which include non-linear behavior, discontinuities, and non-smoothness. A small simplified 1D abstract problem is formulated and solved with this framework to familiarize the reader, before two case studies: (1) a reconfigurable satellite constellation observing Earth targets, and (2) an ocean-going maneuvering platform completing a cross-Atlantic voyage while simultaneously offering itself as a calibration target to overhead Low Earth Orbit (LEO) satellites, are explored in-depth. The analysis shows that maneuver plans generated from the framework can increase the imaging performance of reconfigurable satellites by 25 to 35 percent, and the calibration metric for the ocean-going platform by up to 40 percent. Throughout this thesis, the key design decisions of the framework are discussed. The framework itself is available as Julia code, which has been written to take full advantage of any distributed computing cluster, particularly those managed by SLURM.

Thesis Supervisor: Olivier de Weck
Title: Apollo Program Professor of Astronautics and Engineering Systems

Acknowledgments

There are many people I have to thank not just for their support in this research and the writing of this thesis, but for their invaluable guidance, aid, and assistance rendered towards my entire graduate studies at MIT.

First, I need to thank several wonderful folks at MIT Lincoln Laboratory, which as an organization generously funded me under the Lincoln Scholars Program. Special thanks goes out to Jerry Augeri and Mohamed Abouzahra, both who have mentored and supported me not just in my educational aspirations, but throughout my entire career. A big thanks also to my most agreeable and supportive group management through the years, Greg Berthiaume, Anne Adamcyk, and Mary Girard, whom granted me extensive freedom in organizing my time between work and school, and Jonathan Birge, (a CDO/CSE alum!) who helped me get started along this path.

Second, a big thanks to Oli de Weck, my advisor in the Engineering Systems Lab. Despite all the strangeness of pursuing a degree during a global pandemic, with one whole year of remote classes, he always supported and guided my coursework and research, and allowed me to tailor my research very specifically to my interests. But also, he pushed me to expand outside my safe zone. (Frankly, a quick-turn two-month from conception to launch space experiment is not something that will come up often in my career given its nature, and bureaucracy, so it was definitely worth the ride this time around!)

And finally, a big loving thank you to my wife, who has supported me without fail throughout the entirety of the last two years. She certainly has had to deal with being sequestered with me far more than she might have expected when I told her I was going back to school. Spending these last two years studying with you by my side literally the vast majority of the time shows without a doubt why we work together.

This material is based upon work supported by the United States Air Force under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force.

Contents

List of Figures

List of Tables

Acronyms

AUV autonomous underwater vehicle

BBH building-block hypothesis

CONOPs Concept of Operations

CPU Central Processing Unit

CRLB Cramer-Rao lower bound

ECEF Earth-Centered Earth-Fixed

ECI Earth-Centered Inertial

EO Earth-observing

FOV field of view

GA Genetic Algorithm

GNSS Global Navigation Satellite Systems

GOES Geostationary Operational Environmental Satellite

GOM Global Observation Mode

GP Genetic Planning

GSD ground sampling distance

JIT just-in-time

LEO Low Earth Orbit

LLSC Lincoln Laboratory Supercomputing Center

LLVM Low Level Virtual Machine

NDVI Normalized Difference Vegetation Index

NSGA-II non-dominated sorting genetic algorithm II

PCA Principle Component Analysis

PEARL Platform for Expanding AUV ExploRation to Longer ranges

POCA point of closest approach

PSF point spread function

ReCon Reconfigurable Constellation

RGT repeating ground track

RK4 Runge-Kutta Fourth-Order

ROM Regional Observation Mode

RTOFS Real-Time Ocean Forecast System

SAAL Standardized Astrodynamic Algorithm Library

SATCOM satellite communication

SGP4 Simplified General Perturbation Model 4

SPARC Specular Array Radiometric Calibration

TLE two-line element set

VLC Variable Length Chromosomes

VLG Variable Length Genotypes

Chapter 1

Introduction and Background

In this chapter, I will present the motivation for the design of this optimization framework. Afterwards, I will present background on the the optimization techniques used in the framework, before describing the background on Reconfigurable Constellations (ReCons) and the Platform for Expanding AUV ExploRation to Longer ranges (PEARL), the two systems used as case studies in this thesis. Finally, this chapter ends with a roadmap which describes the outline of the remaining chapters of the thesis in detail.

1.1 Motivation

For any system beyond even trivial complexity, it is often impossible for the designer to craft a solution that is optimal in all expected operating environments. Either the exact specifics of the environment are simply unknown, or the variance between different environments pulls the design in competing and/or incompatible directions. Designers can approach this challenge in many ways. Some simply choose to build a design which is most performant in the most frequently encountered environment. Others choose to accept a lower level of peak performance in exchange for a guaranteed level of performance across the whole spectrum of operating environments.

A third approach is to defer the choice as much as possible, building in the ability for the system to reconfigure itself. One simply waits for the system objectives and

operating environment to become clear before adapting the system into a physical configuration tailored toward whichever reality has materialized. The ability to reconfigure, *reconfigurability*, is a form of *flexibility*, both of which are one of the many *-ilities*, properties of complex systems which become apparent over the system lifetime [34]. Incorporating flexibility into a system design allows responding to the changing external environment, precisely the enabling system quality one needs to follow this third approach.

Such flexibility and reconfigurability can be seen in systems both old and new. Perhaps the archetype of such a system is the Swiss-Army knife, famous for its ability to change its function and adapt to the need at hand that its name has practically become a shorthand descriptor in the English language. A user might not know whether they need a flat-head screwdriver or a Philips-head, but the Swiss Army Knife can simply be changed at the time of use, which is its claim-to-fame.

But more modern and high-tech situations are often encountered. Of particular interest to this thesis is the design of Earth-observing satellite constellations. The first two approaches can be implemented with static, non-maneuvering constellations. One could easily design a satellite constellation which provides good general coverage. For example, the constellations of Global Navigation Satellite Systems (GNSS) like GLOSNASS, GPS, Beidou, and GALILEO are explicitly designed such that multiple satellites are visible from any spot in their terrestial operating region on the surface of the Earth. Alternatively, one could tailor a constellation to maximize coverage over a specific area. Such an example would be the Soviet Molniya satellites constellation [50], which utilized the eponymous Molniya orbits (highly elliptical orbits which provided long periods of observation over high latitudes but poor coverage over other areas) in their mission.

In the last decades, significant thought has been spent designing and analyzing the use of reconfigurable satellite constellations, which exemplify the third, dynamic approach. These constellations are designed with the ability to perturb orbit in-mind, and they agilely respond to the specific objectives they are tasked with as those objectives arise. More background on ReCons will be given in section 1.2, as

such a constellation will be used as a case study for this thesis.

To take full advantage of the flexibility included in large complex systems like ReCons, human planners and designers need tools to identify the optimal plan for reconfiguration. In some cases, say choosing the correct tool out of a Swiss Army Knife, the choice may be obvious and unambiguous, but in large complex systems, with many agents, options, and complicated metrics of performance, the problem can become intractable for human intuition. That's where this thesis aims to improve the body of knowledge. It explores the use of Genetic Algorithms (GAs) with variable length chromosomes to find optimal maneuvers for single or multiple vehicles. ReCons were the particular motivating problem. They contain many agents, with constraints and objectives that can be non-linear, non-smooth, and discontinuous, which motivated the choice of genetic algorithms. The framework is generic though, and can be adapted to other maneuver problems, and as an example, a second case study exploring a maneuvering ground vehicle interacting with non-maneuvering space vehicles is explored.

The use of variable length chromosomes is motivated by a desire to limit the amount of assumptions a human designer might implicitly impose on the domain of potential vehicle maneuvers. A more traditional approach to formulating the problem would limit the potential number of maneuvers or their distribution among the constituent vehicles in the system. Conversely, a designer may try to side-step this problem by formulating a large number of design variables corresponding to more maneuvers than could reasonably be expected to be needed. The idea being, the optimizer could then simply "zero-out" some amount of maneuvers, but this approach has its own problems, particularly with the memory and time problems that come with searching such a high-dimensionality space. Using VLCs, the number of maneuvers is variable, and the optimizer is free to grow or shrink that number as is necessary, without searching an unnecessarily oversized space.

1.2 Reconfigurable Constellations (ReCon)

In this section, I will give a brief overview of the ReCon concept, before discussing recent work in the optimization of ReCons.

1.2.1 The ReCon Concept

Earth-observing satellite constellations have been common from the inception of the Space Age. The utility of viewing the Earth from an uncontested domain such as space, with wide area coverage, and the sensor out of harms way from any hostile actor is immense for both government and commercial spheres. There is now a plethora of both government constellations and commercial constellations which routinely image the Earth.

But there are plenty of inherent trade-offs a designer must make when creating an Earth-observing constellation. There is a conflict between the quality of imagery a satellite can collect, and the size of its field of view (FOV). The lower the altitude of the space platform, the smaller details the on-board camera can resolve, but the smaller the FOV. Alternatively, the same camera at a higher orbit altitude may see more of the Earth, but at the cost of ground resolution.

Thus, designers of Earth-observing (EO) space constellations must weigh carefully the altitude of the orbits, the cost and quality of the cameras, and the total number of platforms. These decisions will determine how often a site on the Earth can be expected to be in the FOV of a sensor (the revisit rate) and the expected quality of the imagery, often quantified by the ground sampling distance (GSD). In addition, constellations may be asymmetric with regards to the geometry of the Earth, meaning that different areas of the Earth's surface will experience different revisit rates and GSD.

Many of the imaging constellations of Earth-imaging satellites are static constellations. That is, the constituent satellites are not designed to significantly perturb their orbits under their own power in the course of normal operations. Examples include Planet's Dove constellation [27], NASA's A-Train constellation of meteoro-

logical satellites [23], and NOAA's Geostationary Operational Environmental Satellite (GOES) systems [23].

Of course, there is no requirement that an imaging constellation be static. That is where the concept of ReCons comes in. In a ReCon, maneuvering of the satellites is expected in normal operation in order to improve imaging performance. The discussion here will focus on ReCons with optical Earth-imaging missions, as that is where a significant portion of academic study has focused.

In the Earth-imaging case, study has focused on the use of repeating ground track (RGT) orbits to lower the revisit rate of a satellite over a specific point on the Earth's surface. Generally, a satellite is not expected to pass over the same points on Earth between subsequent orbits. The Earth, of course, is rotating with a sidereal period of about 86164 seconds, meaning that one orbit later, even if the satellite had returned to the same exact spot in inertial space relative to the center of the Earth, the surface of the Earth has rotated a significant amount. Between subsequent orbits, the path the satellite takes over the Earth will drift.

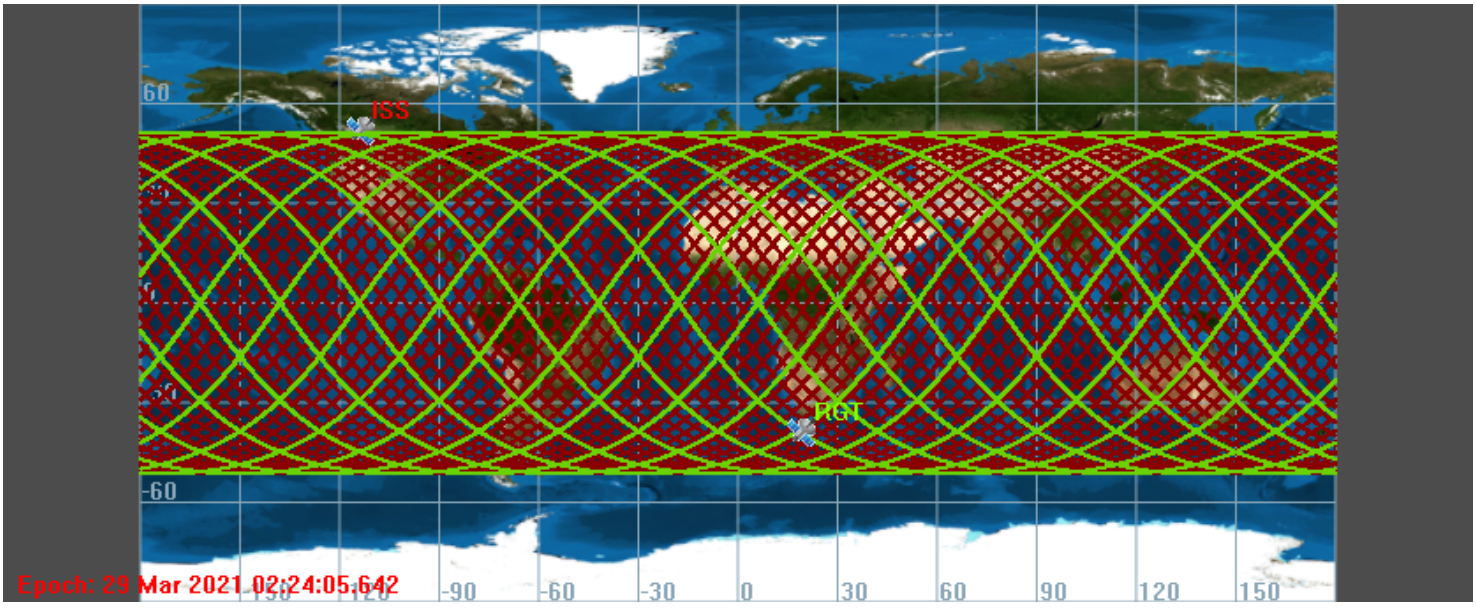

Figure 1-1: A comparison of the ISS orbit and a very similar repeating ground track orbit. Due to the rotation of the Earth, the ISS ground trace drifts over the surface of the Earth, while the RGT orbit stays stable. This image was created with the General Mission Analysis Tool from NASA.

However, it is possible to specifically design orbits which will "sync" with the rotation of the Earth such that the satellite will return to the same spot over the surface of Earth in some small number of sidereal days. One approach to achieve this effect is to match the period of the satellite orbit to the sidereal period of the Earth. This is the case with geosynchronous and geostationary orbits. Unfortunately, the height of these orbits which sync with the sidereal period are 35786 kilometers in altitude, which makes imaging the Earth at high resolution difficult. Of course, it is in fact possible to image the Earth in "high" quality from this altitude, but the cost of the optics is significant and still may not approach the spatial resolution of LEO satellites. For example, the new generation of GOES satellites (the so-called GOES-R series) for meteorological observation are geostationary and host the new Advanced Baseline Imager instrument. This instrument improves upon the spatial resolution of the previous generation of GOES satellites by a factor of four, but still only obtains a resolution of 500 meters [1].

It significantly lowers the cost of high-resolution imagery for the satellites to be at lower orbital altitudes. The problem is then how to ensure an area is imaged regularly with the LEO platform's smaller view of the Earth. In fact, LEO orbits can be designed which exploit the nodal precession induced by the non-spherical shape of the Earth to sync up a satellite with Earth's sidereal period over some integer number of orbits. The satellite then follows a ground track which repeats, and these orbits are called repeating ground track (RGT) orbits. Figure 1-1 compares such an RGT orbit at similar altitude to the ISS orbit. These RGT orbits make use of the non-uniform gravity of the Earth. Because the Earth's gravitational field is not perfectly spherical, (it bulges along the equator) a satellite in orbit will experience non-uniform gravitational acceleration over its orbit, beyond just the effects of its changing altitude. This difference in gravity induces the orbital plane of the satellite to precess. The true gravity field of the Earth is usually described using large series of spherical harmonics matched to large amounts of experimentally collected data [70], but the majority of the "bulge" around the Earth's equator, most responsible for nodal precession, is described in a single spherical harmonic term, called J_2, which

represents the zonal effect of the gravity field.

A small introduction on the mathematics of orbits is required to discuss the mathematics of RGT orbits. A satellite's orbit and its position therein are described with 6 classical orbital elements:

1. a: Semi-major axis. This is the radius from the center of the orbit to the apogee.

2. e: Eccentricity. This describes the shape of the orbit. An eccentricity of 0 refers to a circular orbit and numbers between 0 and 1 describe ellipses.

3. i: Inclination. This is the angle the orbital plane makes with the equatorial plane of Earth.

4. Ω: Right ascension of the ascending nodes. This quantity describes the rotation of the orbital plane around the Earth's axis.

5. ω: Argument of Perigee. This orients the ellipse described with the semi-major axis and the eccentricity within the plane described by the inclination and the right ascension of the ascending node.

6. ν: True anomaly. This is the location of the satellite along the track of the ellipse.

This coordinate system is shown in Figure 1-2.

For a circular orbit and a given J_2 value, the induced precession in the orbital plane is described by three rates:

$$\dot{\Omega} = \frac{3nR_E^2 J_2}{2p^2} \cos i \tag{1.1}$$

$$\dot{\omega} = \frac{-3nR_E^2 J_2}{4p^2} \left(5\cos^2 i - 1\right) \tag{1.2}$$

$$\dot{M} = \frac{-3nR_E^2 J_2}{4p^2} \left(3\cos^2 i - 1\right) \sqrt{1-e^2} \tag{1.3}$$

with $p = a\,(1-e)$. Here we use M, the *mean anomaly*, instead of ν, the true anomaly. For the circular orbits we will handle in this thesis, the mean anomaly is simply the

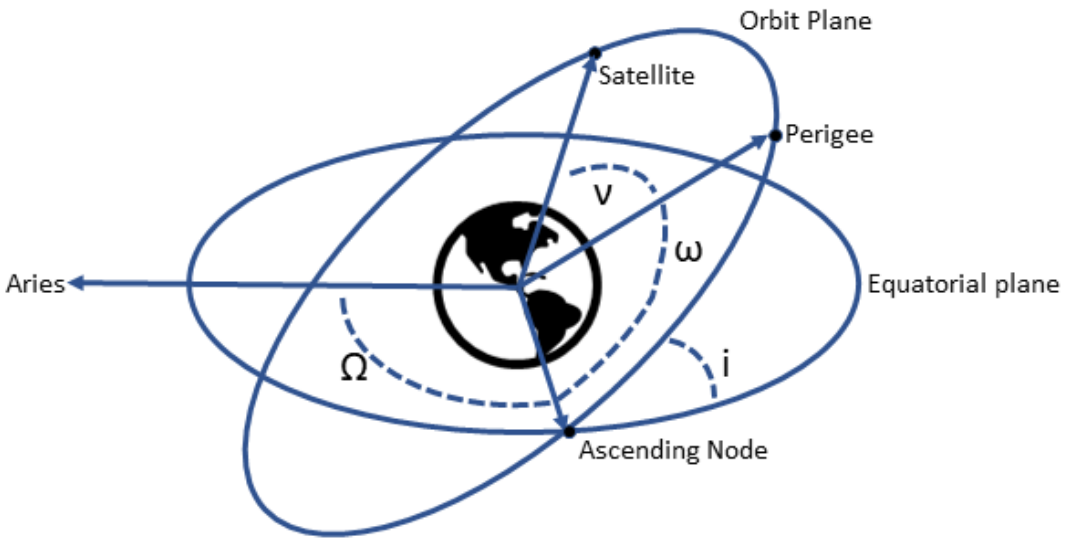

Figure 1-2: The Keplarian coordinate system which to describe the orbit of a satellite. The absolute reference of the system is the direction of the vernal equinox.

angle from the perigee to the current location of the satellite. The mean anomaly changes over time as the satellite moves. The rate of change is the *mean motion*:

$$n = \sqrt{\frac{\mu}{a^3}} \tag{1.4}$$

where μ is the gravitational constant of the Earth. Note, $\dot{M}$ is the additional change in mean anomaly from J_2 effects, separate from the mean motion.

By syncing the induced orbital precession ($\dot{\Omega}$) to the sidereal motion of the Earth, it is possible to place satellites in orbits whose ground tracks repeat on the order 1 or 2 days. The period of such an orbit is given by [65] as

$$T = \frac{2\pi N_d}{\omega_E N_o}\left(1 + 2\xi\frac{n}{\omega_E}\cos i\right)^{-1}\chi \tag{1.5}$$

with

$$\chi = 1 + \xi\left[4 + 2\sqrt{1-e^2} - \left(5 + 3\sqrt{1-e^2}\right)\sin^2 i\right]$$

$$\xi = \frac{3R_E^2 J_2}{4a^2\left(1-e^2\right)}$$

$$T = \frac{2\pi}{n}$$

N_d is an integer number of days, N_o is an integer number of orbits and $p = a(1-e)$, the semilatus rectum. Equation 1.5 can be solved easily with a non-linear solver for a semi-major axis. Figure 1-3 shows several families of RGT orbits in LEO.

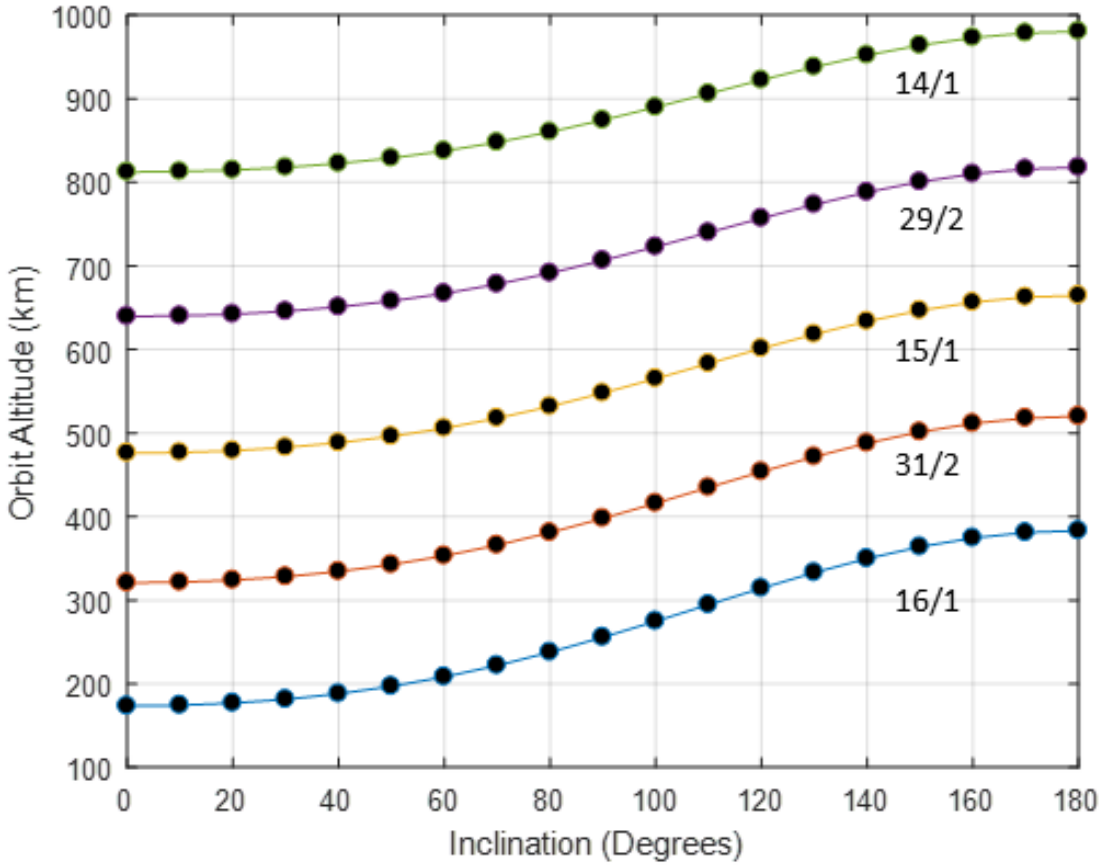

Figure 1-3: Families of RGT orbits. The ratios labelling the lines indicate the ratio of orbit periods to the sidereal day. Results are shown for circular orbits ($e = 0$).

The current ReCon concept for Earth-imaging constellations involves maneuvering satellites into these RGT orbits over locations of interest on the Earth. These satellites will then revisit these high-value locations every one or two days, decreasing the average revisit time. A satellite in a ReCon has two major modes of operation. The mode just discussed, wherein the satellite has been placed in an RGT orbit to minimize revisit time for some ground location, is called Regional Observation Mode (ROM). The other mode is called Global Observation Mode (GOM). In this mode,

the satellite is not in an RGT orbit, and uses the drift of its ground track to image different areas of the Earth over time. In this mode, the constellation will periodically image a large section of the globe.

The Concept of Operations (CONOPs) of the constellation as a whole is to idle in GOM, providing imagery over large areas. When a target of interest is identified which would benefit from imagery with enhanced temporal resolution, some subset of satellites will use in-plane maneuvers to enter ROM over the target. Maneuvering is limited to in-plane maneuvers as out-of-plane maneuvers are much more fuel-inefficient. The maneuver is required to both place the orbit at the required semi-axis to enter a RGT orbit, but also phase the satellite such that the RGT actually passes over the target of interest. After the event of interest has ended, the satellite maneuvers again to return to GOM, where it idles until the next time an entry into ROM is required.

There are of course, a large amount of design factors and decisions which go into the transitions between ROM and GOM. For example, some work [64] has suggested the use of hybrid propulsion systems on ReCon satellites. The combination of chemical and electric propulsion on a single platform would enable a fast transition from GOM to ROM (where the ability to quickly respond to an event on the ground is key) but a fuel-efficient transition back to GOM, though slower. Discussion over design trades in the physical construction of satellites is out of scope for this thesis, so will not be mentioned further.

ReCons fill a need for relatively quick high-resolution imagery over specific locations of the Earth. ReCon satellites reside at a relatively low altitude, and thus can provide high quality, high resolution imagery with relatively inexpensive optical systems. While a static LEO constellation would then require many platforms to ensure high temporal resolution over any specific spot of the Earth, ReCons exploit maneuvering into RGT orbits in order minimize revisit periods with less total satellites. Indeed, even with the added expense of maneuvering fuel and equipment, previous work has shown that ReCons can provide more cost-value than static constellations in many scenarios [58].

1.2.2 Optimization of ReCons

The design and operation of ReCons for Earth-imaging tasks naturally leads to a number of optimization problems:

- What is the optimal amount of fuel and propulsive ability for the platforms?
- How many satellite platforms should there be?
- What GOM orbits should the platforms idle in?
- For a given imaging task, which platform(s) should be assigned to the task?
- What ROM orbits should be used for the task?
- How should the platforms maneuver into and out of ROM?

There exists a substantial amount of previous work on these issues. Legge [58] developed a framework for co-optimizing the design of the satellites with the GOM orbits, using GAs. As a subproblem, to evaluate a given orbit/satellite configuration, Legge was required to optimally select RGT orbits and formulate a maneuver plan for a series of imaging events. After calculating potential passes over a target area and removing dominated solutions, he applied a dynamic programming approach to create an optimal trade-off curve between required delta-V and gained utility. This involved choosing the optimal subset of satellites to maneuver as well as some parameters for the RGT orbit. Notably, the parameters were reduced to a discrete sampling of orbit parameters, rather than a continuous space. It should also be noted that Legge simulated sequential imaging tasks, and did not investigate responding to multiple simultaneous ground events.

Straub [74] expanded upon Legge's work, providing additions to the scheduling framework and the ability to include agile satellites which incorporate the ability to slew the camera for off-nadir imaging. In addition, Straub introduced metrics to quantify image quality for each pass, and combine multiple passes into a cumulative performance metric. A whole set of optimization techniques was applied to this problem, including ant colony, simulated annealing, and genetic simulated annealing.

Mcgrath [61] investigated the imaging of mobile targets with ReCons. Of course, with a mobile target, the use of RGT ROM orbits is not a given, as the mobile target may simply travel away from the ground track. A graph theory approach was used to find optimal maneuvers for imaging mobile targets, providing several discrete selections of maneuvers for multiple platforms at multiple time steps. The graph of possible selections is the explored for the optimal selection. Morgan [64] followed on to this work, applying a genetic algorithm to replace the discrete maneuver options at each time-step with a continuous selection.

It is the optimal selection of satellite maneuvers that this thesis will investigate as a case study. This aforementioned works applies implicit designer decisions on the selection and formulation of possible choices. Legge chose from a discrete set of maneuver parameters rather than a continuous domain. Mcgrath and Morgan enforced discrete times at which the maneuvering can occur. In the case study, the VLC GA framework of this thesis will attempt both to remove some these implicit assumptions as well as optimize a more complex performance metric.

1.3 PEARL and Vicarious Calibration

Now, I will give an overview of the PEARL concept, as well as vicarious calibration for orbiting satellites. The hypothetical use of a PEARL ocean-going platform hosting vicarious calibration equipment will be used as the second case study.

1.3.1 PEARL Concept

The Platform for Expanding AUV ExploRation to Longer ranges (PEARL) is an autonomous sea-going vehicle developed by the Engineering Systems Laboratory at MIT. The main design goal of PEARL is to increase the range and endurance of autonomous underwater vehicles (AUVs). These AUVs are limited by the amount of battery capacity they carry, and require frequent support by crewed vessels for recharge and data export. PEARL aims to be an autonomous platform which AUVs can dock, recharge, and transfer data. The PEARL platform contains the required

docking apparatus, solar panels for power generation, and radio links to satellite communication (SATCOM) networks for data exportation. Further info on PEARL can be found in [43], [44], and [4].

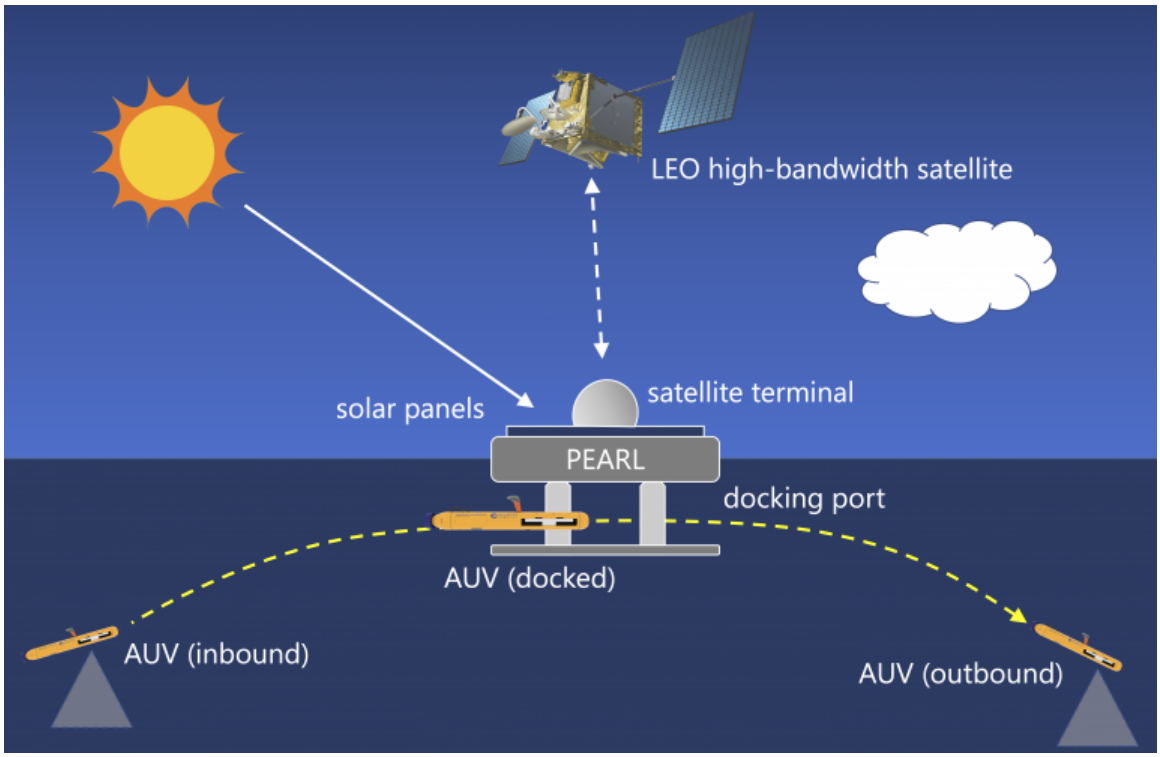

Figure 1-4: Potential CONOPs of a PEARL system, adopted from [4]. Here the PEARL system acts as mobile base of operations for AUVs, providing energy and communication links to overhead satellites.

1.3.2 Vicarious Calibration

Of particular interest for this thesis is the use of PEARL to host calibration sensors for Earth-imaging satellite platforms. Earth-imaging satellites need periodic calibration data to convert the raw data collected via their optics to real-world values. All optical imaging optics will introduce distortion into an image. In particular, the error introduced from this distortion can become quite large when imaging small targets, especially of subpixel sizes. Such distortion can be characterized with the impulse response of the optical system. The point spread function (PSF) is the the actual curve of light measured by an imager system when presented with an

infinitely small point of light. By characterizing the PSF, the recorded image can be corrected for these optical distortion effects. In addition, radiometric calibration (that is, an accurate conversion of raw digital counts to real-world intensity) over various bands is necessary for many EO applications. For example, LANDSAT provides an estimate of vegetation coverage called the Normalized Difference Vegetation Index (NDVI) which requires the use of two bands of reflectance measurements and three bands of brightness measurement [2]. These bands and measurements therein must be accurately calibrated for the proper calculation on this metric.

Of course, focal planes and optics can be calibrated on the ground, but it is not given that the sensor parameters will not drift over the course of the mission or that new mission parameters will require calibration in different regimes. Thus, it is often necessary for instruments to be calibrated on-orbit. Finding convenient calibration targets for in-orbit satellite platforms is not a trivial matter. It is possible for a satellite to include a calibrated source on-board, but this is an additional cost levied onto the development of the system, often too large for smaller satellites. Another approach is to collect sample imagery of the Earth and compare the data to published final products from science missions with precise calibrations. LANDSAT is a popular choice for this type of "second-hand" calibration as it is well-calibrated and publicly releases data. This approach is sometimes called a "Big Satellite" method, as data from large (both in project dollars and generally size) satellites are used to bootstrap calibration for smaller satellites. There are some obvious disadvantages though. Periodic re-calibration requires the orbit of the system to match up with one of these big satellites, sometimes with simultaneous viewing of a given land area required. In addition, there is no guarantee that the data provided by such "big satellites" will match the specific instrument parameters, and thus calibration may be incomplete or imperfect.

As a solution to this calibration problem, industry has begun to offer "calibration-as-a-service," wherein sites on the ground provide well-calibrated stimuli for imaging by satellites passing overhead. An overhead satellite can image one of these sites, and with the stimulus data provided by the calibration operator, tune their own sensors.

One such method for providing calibrations for on-orbit platforms is the Specular Array Radiometric Calibration (SPARC) developed and patented by Raytheon [72]. This method uses an array of mirrors reflecting the sun towards the satellite platform in question. The sun then becomes a "virtual point source" enabling the calibration of imaging optics.

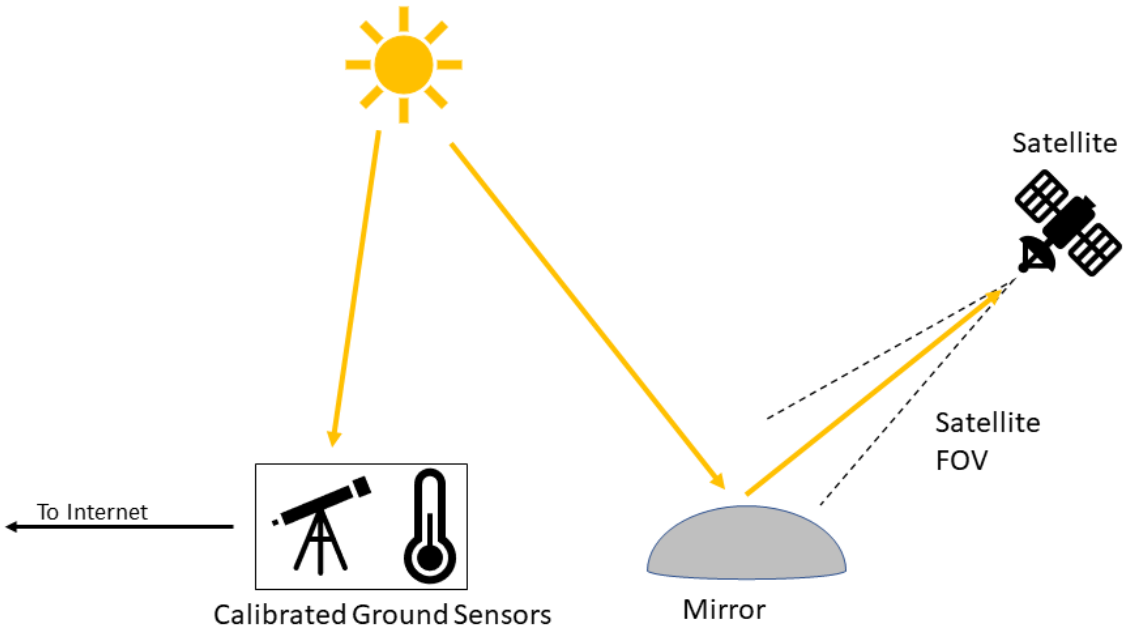

Figure 1-5: A FLARE site consists of a steerable mirror which reflects the image of the sun into a passing satellite's FOV. As the satellite measures the reflected image, calibrated ground instruments co-located at the site record environmental data for distribution over the internet.

Raytheon has licensed out this technology to LabSphere Inc. which has deployed a network of ground sites specifically for use in vicarious calibration with the SPARC method. This network of calibration sites is called FLARE [19]. At a FLARE site, multiple arrays of convex mirrors with pointing authority can be automated to reflect the sun as a point source towards a satellite in need of calibration. Calibration can be provided on-demand and off-nadir at a more flexible schedule than a "Big Satellite" method or naturally occurring calibration sites. The overall CONOPs are shown in Figure 1-5.

Such a calibration network provides numerous advantages. The sites can be scat-

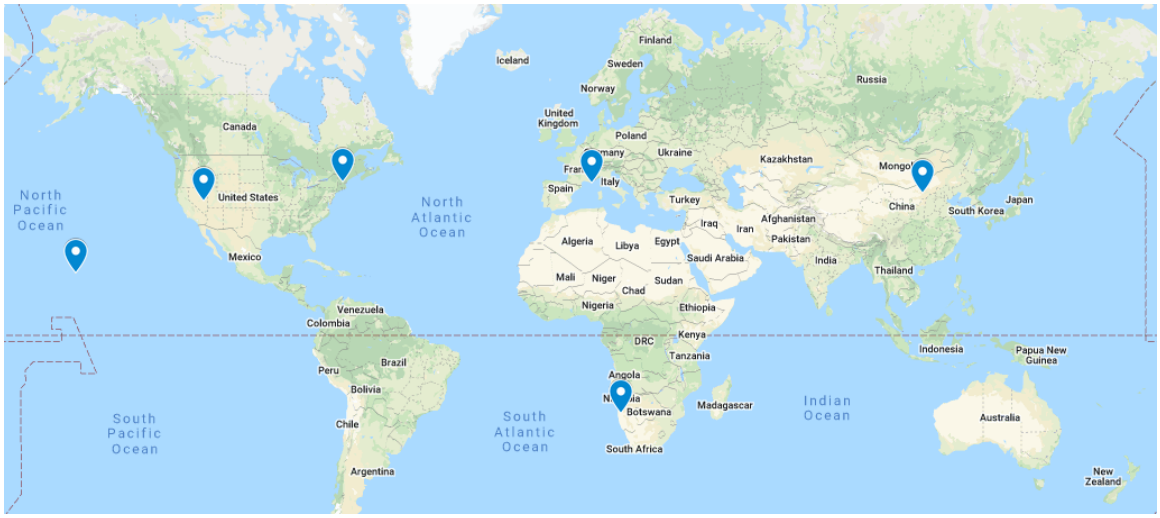

Figure 1-6: A hypothetical lay down of calibration for of a large satellite. The sites include stations at the RADCALNET [15] locations as well as LabSphere headquarters and Hawaii. This is the lay down used as an example in [19]

tered and geographically diverse, removing the need to sync up with the orbit of a "Big Satellite" to obtain calibration data. In addition, the environmental measurements needed for calibration, such as various meteorological quantities such as humidity, temperature, and pressure, are made directly at the calibration site. Calibration is also on-demand. As such, there can be realized budgetary gains by tailoring the calibration schedule to the exact needs of a specific sensor platform.

The PEARL project has already experimented with prototyping a similar mirror target. PEARL operators have measured the response from such such a mirror using overhead drones. The concept can be expanded from overhead drones to satellites in LEO orbit. PEARL's mobility, as compared to the inflexibility of static sites, provides an obvious advantage to any entity attempting to provide calibration targets, especially to imagers in LEO orbits with their comparatively small field of view. PEARL can proactively move into the FOV of a satellite and/or into an area with more sunlight to aid calibration. The operation of such a calibration site in the marine environment has particular opportunity to aid in solving the Ocean Leaving Radiance problem. Work by LabSphere has already suggested that the use of mirrors on the ocean surface, or even submerged under the surface, can provide a solution to the Ocean Leaving Radiance problem beyond the ability of current methods [19] [68] [69]

[73].

It is this proposed mode of operation that will serve as the second case study for the optimization framework. The VLC GA framework is used to generate Pareto optimal sets of maneuvers which best allow a PEARL platform to complete a fuel-efficient transit across the ocean while maximizing its calibration-availability to a select set of overhead LEO satellites.

1.4 Genetic Algorithms

Genetic Algorithms (GAs) are a class of heuristic optimization techniques. What is meant by *heuristic* is that these optimization algorithms are not guaranteed to find a globally optimal solution, and even if one is found, the algorithm has not proven that said solution is globally optimal. The benefit of such a heuristic technique over more traditional techniques is that they can handle more complex constraints, objectives, and design domains. The formal representation of a non-linear optimization problem is

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to } g_i(x) \leq 0 \quad \forall\, i \in \{1, \ldots, m\} \\ &\qquad\qquad h_j(x) = 0 \quad \forall\, j \in \{1, \ldots, n\} \\ &\qquad\qquad x \in X \end{aligned}$$

Here, we try to minimize an objective function $f(x)$ over over the domain of X while satisfying inequality constraints $g_i(x)$ and equality constraints $h_j(x)$. The general problem can be quite complex, as both constraints and the objective function may be non-linear, non-convex or ill-behaved at points.

GAs attempt to find the minimizing design vector by taking inspiration from biology, specifically natural selection. From a population of individuals, each representing a potential design vector, GAs apply operators inspired from the natural world to gen-

erate the next generation of solutions. Notably there is *selection*, akin to survival of the fittest, where only better-performing individuals are allowed to participate in the breeding of the subsequent generation, *crossover*, the inheritance of traits from multiple parent individuals, and *mutation*, the random perturbation of design variables within a single individual.

Although there are previous publications on using computers to simulate the evolution of "organisms," Fraser described all the quintessential elements present in modern GA implementation in 1970 [39]. By the 1980s, commercial software incorporating genetic algorithms was available for use by the public [26]. GAs generally find their use in problems with complex fitness domains, that is when the optimization metric to be minimized as a function of parameters and design variables contains many local minima, as well as possible discontinuities and non-smooth regions. Traditional hill-climbing algorithms are liable to become "stuck" in the local optima, or fail around regions of the fitness space that are not well-behaved. While heuristic, GAs have the potential to escape local optima, and function in a derivative-free manner; avoiding many of the problems that can come with a fitness domain that is not totally well-behaved.

1.4.1 Single Objective Optimization with Genetic Algorithms

A run of a GA starts with developing an initial population of individuals. Each individual represents one potential set of values for the design variables of the problem, and these values are encoded into its *genome*. Generating the initial population is often done with random selection, and it is not required that all individuals actually exist in the feasible portion of the design space. Designers can attempt a more targeted approach than uniformly sampling the entire search space, placing more individuals in areas of the design space that are thought to be near optimal.

Once generated, each population member is scored for *fitness*. For each individual, the values for the design variables and parameters are passed to a fitness function, which returns the performance metric ($f(x)$ in the formal problem definition) to be minimized. Since this fitness function is evaluated for every individual in every gen-

eration, GAs undertake a relatively high amount of functional evaluations in a single optimization run, a problem that considerable effort was spent on in this thesis to keep the problem computationally tractable. After fitness is scored, the *selection* operator is applied to the population in order to determine which individuals will breed the next generation. This selection operator is based on the fitness of the individuals, with better-performing individuals more likely to enter the breeding population. Infeasible solutions (those that violate the given constraints of the problem) are penalized relative to the feasible solutions. There are numerous different schemes for selection operators, but this thesis in particular will use *tournament selection* [62], wherein individuals are paired up in head-to-head competitions in multiple rounds until the population is pared down to the desired level. In this scheme, it is thus possible for low-performing individuals to make their way into the breeding population, as long as they are paired with even lower-performing individuals in each round.

With the breeding population decided, the *crossover* and *mutation* operators are used to generate new individuals. Crossover refers to generating a genome of a child individuals using sections from each parent. This parallels the type of genetic recombination that occurs in sexually reproducing species, and as such, the design vectors are often referred to as chromosomes. The mutation operator is also analogous to the biological world, referring to random changes in the values of a child's genome.

The specifics of mutation and crossover are dependent on how the chromosomes are represented in digital space. A chromosome can, for example, be treated as just a long string of binary 0s and 1s. In this case, crossover may select sections of each parent's binary string whose boundaries don't sync up with the boundaries of individual design variables. It is possible for example, if a design variable is stored as a 64-bit precision floating point number that the first 10 bits come from one parent, while the remaining 54 bits come from the other. Alternatively, other crossover operators enforce such boundaries between different design values, forcing recombination of chromosomes to treat individual values as atomic, and indivisible. Similarly, the specifics of the mutation operator depend on the representation of the chromosomes. If treated as a binary string, a simple random flip of some subset of bits is sufficient. However, a more

sophisticated mutation operator may redraw individual design values (or apply small perturbations to such values) from variable-dependent distributions. An example would be to only redraw a design value within the constraints given in the problem.

Regardless of the specifics of the operators, child individuals are bred by randomly grouping parent individuals together and using the crossover operator to create a design vector, at which point a mutation operator may perturb this vector. These children become the population for the next generation. Some GAs will introduce the concept of *elitism* into this process. With elitism, the best performing individuals from the previous generation are passed directly into the next generation. This keeps the best solutions in the population despite the randomness of crossover and mutation.

This selection and breeding process continues until a termination criteria is met. There are a large amount of choices for this criteria, such as a certain absolute level of fitness, a number of generations without a given level of improvement or simply a maximum number of generations to breed. Once termination criteria is met, the designer can inspect the best performing individuals of the population. Figure 1-7 outlines the high-level flow of a notional GA.

There is no proven rigorous answer that absolutely explains why GAs are successful at finding optimal solutions. I will briefly discuss one high-profile explanation here, as I will contextualize some of the design choices I describe in chapter 2 in terms of this narrative.

The "building-block hypothesis" posits that GAs find individuals which are fitter-than-average partial solutions. Each of these individuals is one of the eponymous building-blocks, which can then be fit together in crossover to form an even better solution. GAs outperform random search because they attempt to fit together partial solutions rather than just trying every conceivable possibility [40]. This explanation is not accepted as given by the entire community, and one does not have to look far for examples of disagreement [30] [67], however, it is a useful mental narrative for contextualizing results.

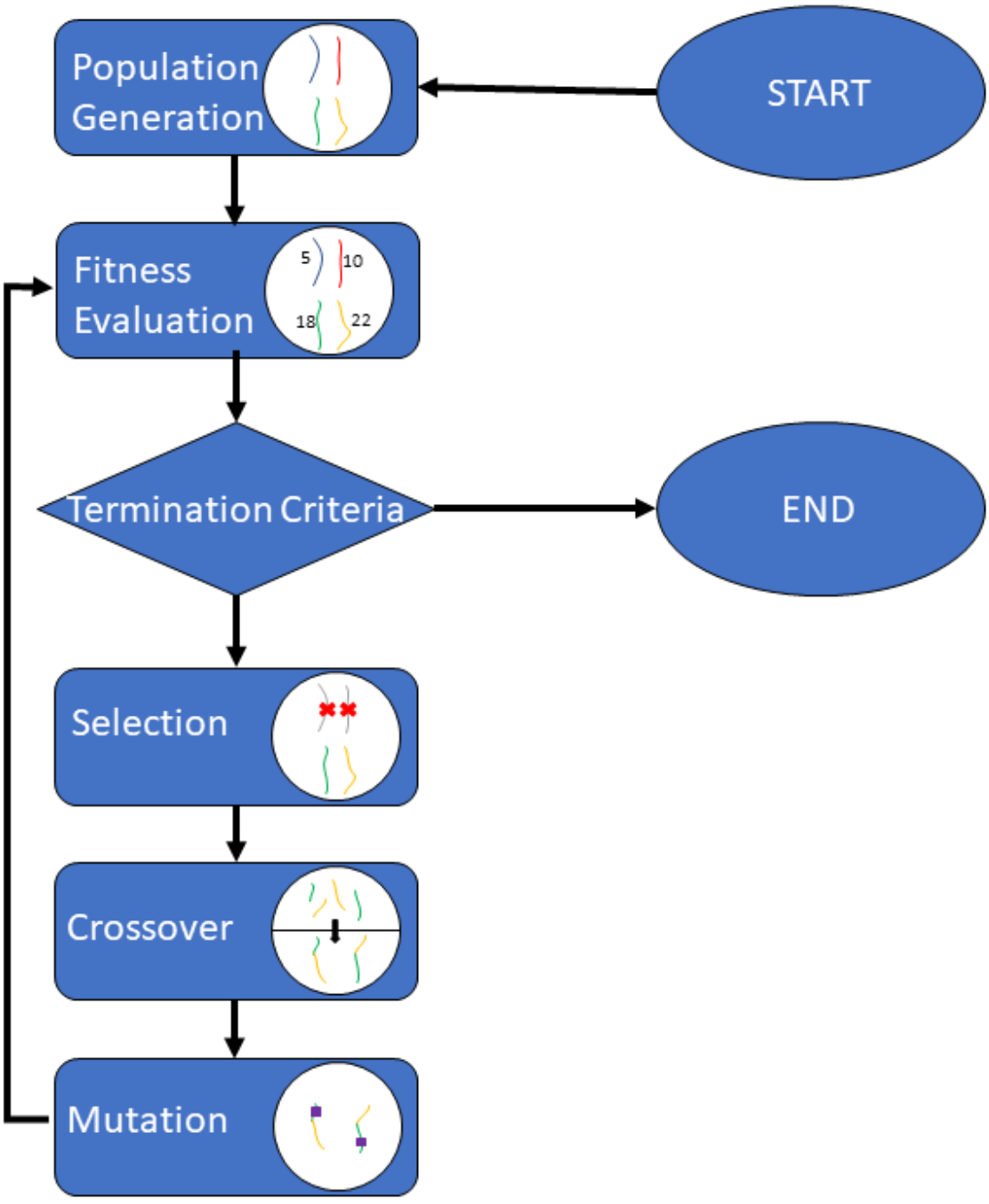

Figure 1-7: An overview of the steps of the process flow of a notional genetic algorithm. After an initial population is generated, the algorithm iterates over applications of selection, crossover, and mutation to breed new generations until termination criteria is met.

1.4.2 Multi-Objective Optimization Non-dominated Sorting Genetic Algorithm II

What was just previously described was the use of a GA to optimize a single performance metric. It is often the case that there exist multiple, sometimes competing,

performance objectives in an optimization scenario. For example, take the case of finding the optimal route a vehicle takes from origin to destination. A designer may wish to take into consideration not just the travel time of a given route, but also how fuel efficient the route is. A traditional single-objective optimization would only be able to minimize time or maximize fuel efficiency, but it would be more useful to present the designer with the optimal trade-off curve between the two.

This is now a multi-objective optimization problem. If there is no known weighting or preference between the multiple objectives, it is not usual to produce only a single solution. Instead, multi-objective optimization algorithms seek to find the set of *non-dominated* solutions. A solution is non-dominated if there are no other solutions with better performance in at least one objective without at least one other objective degrading. The set of non-dominated solutions will form a trade-off curve (in 2D, but a subspace in higher dimensional objective spaces) between the multiple measures of performance. This curve is known as the Pareto front, and it can be presented to decision-makers for a final selection based on outside input.

Single objective optimization methods can be used in larger algorithms to trace out a Pareto front. One example would be a weighted sum approach, wherein multiple objectives are combined into a single objective with a set of weights. A single objective optimizer can then optimize this weighted combination to find a single point on the Pareto front. To trace out the entire front, the optimizer needs to be run with many different objective weights, making the process computationally intensive. In addition, this method, and similar methods like adaptive weight [52] and normal boundary intersection [33] can be problematic when approximating complex Pareto fronts [37].

But there are algorithms which attempt to map the entire Pareto front simultaneously, a much more computationally tractable solution. The non-dominated sorting genetic algorithm II (NSGA-II) is a GA extended to trace out the front of non-dominated solutions [35]. NSGA-II uses a modified fitness function for rating individuals. Once scored on the multiple objectives, NSGA-II sorts the individuals into different *fronts* based on how dominated a solution is. The first front is the set of

non-dominated solutions. The second front is those solutions only dominated by individuals in the first front. The third front contains solutions only dominated by those in the first or second front, and so on. Within each front, individuals are scored based on distance to neighboring solutions, and are penalized for being close. Sorting by front number and then crowding metric is the fitness metric for NSGA-II. Thus, the population faces selection pressure to both push out towards the real Pareto front and spread across it. This fitness metric is used both for selection of parents (usually by a tournament method) and then for deciding which individuals from the combined parent and child population survive to the next generation. Figure 1-8 demonstrates how this non-domination sorting works at the culling stage.

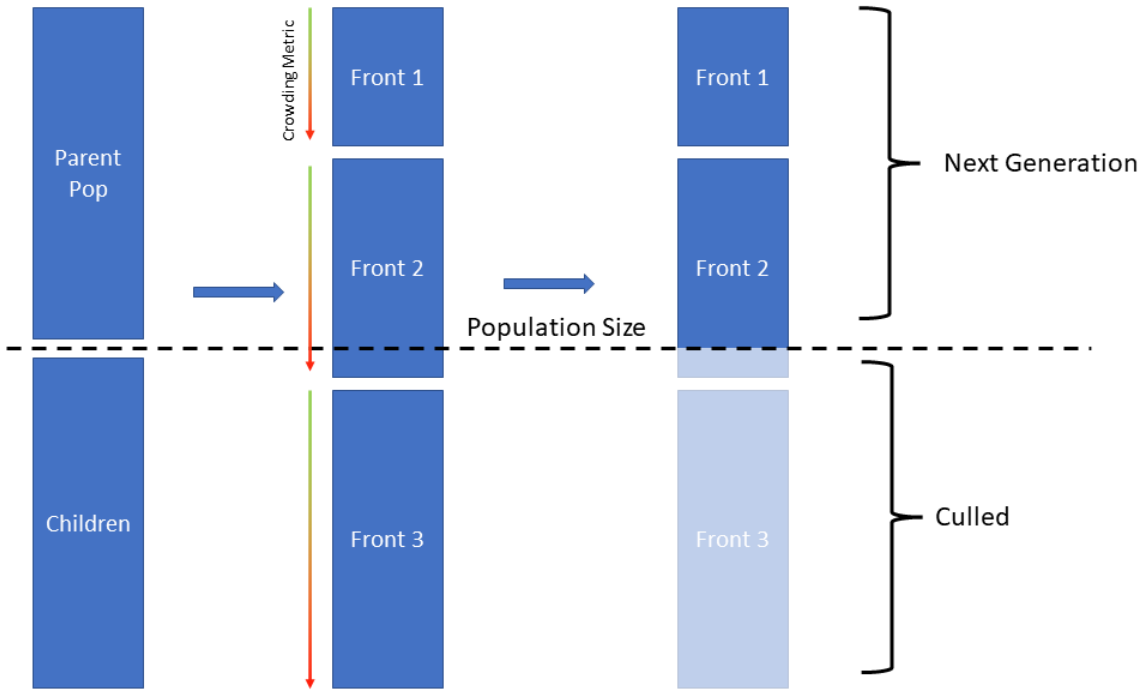

Figure 1-8: The Pareto sorting used when simulating a generation in NSGA-II. The individuals are first sorted into different *fronts* based on domination. Within each front, individuals are sorted by distance to their nearest neighbor in the front. Only the top-performing half survives the cull to the next generation. As a form of *elitism*, both the children and their parents are allowed to compete for a slot in the next generation.

The authors of NSGA-II have provided an efficient front sorting algorithm of $O(MN^2)$ complexity (M being the number of objectives, and N the number of indi-

viduals) [35]. Each individual in the population is compared to every other individual. Pairings wherein one individual dominates another are noted and book-keeping tracks: (1) how many solutions each individual is dominated by, and (2) for each individual, a list of other individuals it dominates. Once this info is known, all the non-dominated solutions are placed in the first front (the approximation to the Pareto front). One by one, each member in the first front has its list of dominated individuals checked, and each of those individuals has their count of dominating solutions decremented. Those individuals now newly finding themselves with a domination count of zero (meaning they were only dominated by individuals in the first front) are placed in the second front. This process continues onward for as many fronts as needed until the population is exhausted.

```
function non_domination_sort(Population P)
# P an array of individuals
# F[n] A set holding the members of the nth front
# domination_set[n] The set of solutions dominated by P[n]
# domination_count[n] The quantity of solutions which dominate P[n]
for i = 1 to length(P)
    domination_count[i] = 0 #Number of solutions which dominate P[i]
    domination_set[i] = {}  #Set of solutions P[i] dominates
    for j = 1 to length(P)
        if P[i] dominates P[j]
        then
            add_to_set(domination_set[i], j)
        else if P[j] dominates P[i]
            domination_count[i] += 1
        end if
    end for
    if domination_count[i] = 0
        # P[i] is in the first front
        # Add it to the set of individuals in
        # the first front
        add_to_set(F[1], i)
        rank[i] = 1
end for
```

```
i = 1
while F[i] is not empty
    for every p in F[i]
        for every q in domination_set[p]
            domination_count[q] -= 1
            if domination_count[q] = 0
            then
                # P[q] is a member of the next
                # front
                rank[q] = i + 1
                add_to_set(F[i+1], q)
        end for
    end for
    i = i + 1
end while
```

As previously mentioned, inside each front individuals are rated on a crowding metric. This is the distance to their nearest neighbors (in objective space) in the same front. Specifically, a cuboid is drawn using the nearest neighbors as two of the vertices (across a diagonal of the cuboid). The average length of a side of the cuboid is the crowding metric. The individuals at the extrema of the front (specifically one individual at each extrema) are assumed to be an infinite distance away from other solutions, as this guarantees their success in both the selection operator and surviving to the next generation.

With front and crowding distance calculated, the individuals can be ranked. Individuals in a lower front always have a superior rank to those in a higher front, and within a front, individuals further away from others are superior to those closer. Selection, crossover, and mutation are then applied as before, but NSGA-II explicitly includes elitism. Once bred, the children are combined with their parents into one large population, which is then sorted by the front-distance metric. Then, only the top-performing individuals are passed onto the next generation.

There are other similar types of GAs that are used in multi-objective optimization, such as SPEA [77], PAES [53], and MOGA [38]. I chose NSGA-II for its history of

application to numerous problem areas, and my familiarity with it (notably, it is included in MATLAB). The framework in this thesis is based on NSGA-II.

1.4.3 Variable-Length Chromosomes

The traditional GA uses a fixed length genome with a set number of design parameters. However, it is clear there are problems which GAs can solve where the optimal number of design parameters is not known *a priori* and it would be beneficial to allow the optimizer to explore solutions of different chromosome lengths. In the realm of GAs, chromosomes that are allowed to grow and shrink in length are known as Variable Length Chromosomes (VLC), or sometimes Variable Length Genotypes (VLG). Such algorithms have been studied and characterized in literature [41] [45].

The use of VLCs requires modifications to the crossover and mutation operators. First, the crossover operator must be able to handle parents of different lengths. For fixed-length genomes, the crossover point(s) is the same in both parents, and the first half of one parent chromosome is combined with the second half of the other, resulting in a child chromosome of the same fixed length. Variable length implementations can choose points that do not line up between chromosomes. Indeed, since the parents might have chromosomes of different lengths, it can be impossible to even identify a corresponding point between the two. There are varied methods for choosing crossover points and then recombining the spliced chromosome fragments. Algorithms can range from simple random selection to involved sequence similarity comparisons between chromosomes [49].

The mutation operator is often augmented to allow the insertion or deletion of individual genes, alternatively increasing or decreasing the total chromosome length. These are often implemented as point operators, copying or deleting at randomly chosen points and such changes can accumulate over time into large size variations. However, there are some new methods which eschew such point operators in favor of folding in length changes entirely within the crossover operator [57].

In addition to the modifications to these traditional operators, some implementations (e.g. [66]) also include a *reduction* or simplification operator. This operator

removes inactive or redundant segments from a chromosome. While often times leaving these inactive sections can have some benefit (for example, there is always a chance an inactive segment mutates and becomes active with some utility), there are often computational considerations that come into play with large genomes. If the computational complexity of the fitness function is proportional to the length of the genome, the entire optimization process will slow. In addition, memory constraints can come into play with larger genomes, not to mention knock-on time effects from cache sizes and such.

The use of VLCs is particularly interesting to this thesis's focus on vehicle maneuver problems. Although some numerical relationship could be enforced between number of maneuvers and the number of objective sites, and/or the number of vehicles, in general, there is no reason to believe such a relationship is the optimal choice. Thus, VLCs are an attractive option to remove the this choice from the human and fold it into the domain of the optimizer.

1.4.4 Genetic Planning

Genetic Planning (GP) is the application of genetic algorithms to solve planning tasks. A related field is the area of Genetic Programming, the use of GAs to design computer programs which can solve problems or generate plans. The use of Genetic Programming is out of scope for this thesis, and is not explored (However, the optimization framework presented in this thesis could be construed as a very basic and limited form of Genetic Programming.)

GAs, have been applied to various sorts of planning problems, including traffic management [32], path-planning [66], satellite constellation design [47] and even space vehicle trajectory planning [59]. Many of these formulations make use of VLCs in their formulation. Notably though, I am not aware of a combination of VLCs and NSGA-II applied to vehicle maneuvering the way it is formulated in the optimization framework developed for this thesis.

1.5 Thesis Roadmap

The remainder of this thesis will be organized as follows. Chapter 2 will provide an overview of the VLC GA formulation for vehicle maneuver planning, describing the overall structure of the algorithm, how it includes NSGA-II, and the general structure of the chromosomes. Motivation for the use of genetic algorithms over more traditional optimization schemes will be presented. In addition, I will spend some time discussing the design and development of the framework from a computer science perspective, as the efficient performance of the code was key to keeping the problem tractable and solvable. Finally, a simplified one dimensional toy problem will be presented, in order to familiarize the reader with the framework by example.

Chapter 3 will present a hypothetical ReCon-like satellite constellation as a case study for the application and evaluation of the framework. This case study will seek to expand previous work on maneuver planning in ReCons, presenting a case with many simultaneous targets,a complex performance metric, and a continuous and less constrained domain of possible maneuvers/orbits.

Chapter 4 will present a hypothetical transit Atlantic of a PEARL platform while acting as a vicarious calibration target for overhead passing LEO satellites. The path across the Atlantic will be planned using the framework to meet travel constraints while maximizing calibration opportunities. This problem acts as a sort of inverse to the case study which will be presented in Chapter 3. In Chapter 3, space platforms move to interact with non-maneuvering Earth locations, while in Chapter 4, a maneuvering Earth-based platform maneuvers to interact with non-maneuvering space platforms.

Finally Chapter 5 will act as summary of conclusions and an outline for areas of potential future work.

This thesis has 3 main objectives, which these chapters will speak to:

- Develop a multi-objective genetic algorithm framework using variable length chromosomes to optimize sets of vehicle maneuvers
- Showcase the performance and applicability of this framework for ground-space

interactions with two case studies

- Draw some useful insights about those two real-world application using the framework, as well as general insights about the optimization approach

Chapter 2

An Extensible Variable Length Chromosome Genetic Algorithm Framework for Vehicle Maneuvers

In this chapter, I will present the high-level VLC GA framework I have implemented for this thesis. I will start with motivation for the choice of GAs, before moving onto a description of the generalized structure of the chromosomes and genes whose specifics must be tailored by the user for a specific problem. Then I will describe the specific design choices in the high-level algorithm based on NSGA-II. After the description of the data structures and high-level algorithm, I will discuss computer science considerations relevant to performance and tractability, both with respect to the high-level framework as well as the scenario-specific fitness functions which are used in the case studies. Finally, for clarity, I will present the toy problem I used during development, as well as details on the specifics of its implementation in the optimization framework. This chapter will end with significant design lessons I gleamed when iterating on this toy problem.

As a high-level overview, the framework formulated for this thesis is an instance of NSGA-II extended with variable length chromosomes. The individual genes in this case are descriptions of single maneuvers for individual vehicles. Each individual in the population has a vector of these genes, in what I am calling a chromosome.

Since the chromosomes are variable-length, they are allowed to grow and shrink as necessary, removing implicit relationships other schemes may impose between the number of maneuvers and the number of vehicles and/or collection targets.

2.1 Why Genetic Algorithms?

The choice of a framework with a GA as its core optimization method was motivated by several factors. First, many vehicle maneuver problems have complex fitness domains. The cases in Chapters 3 and 4 both have complicated performance metrics and constraints which are not well-behaved over their entire domains. The non-convexity of the fitness domain means that local search methods like hill-climbing algorithms will converge on local optimums. Of course, there is no guarantee that a heuristic algorithm like a GA will not also converge on a local optimum either, but there is more opportunity to explore the search space and escape such local optimums.

There is also the fact that many of these non-GA algorithms will require the gradient to exist over the fitness domain. The types of problems explored in this thesis have not only continuous design variables, but also integer and categorical. An obvious example is that for any multi-vehicle problem, any formulation which does not hard-code certain design variables to certain vehicles must have some sort of categorical variable to differentiate among those vehicles. The gradient simply will not exist over that design dimension. Even if the design variables were all continuous, the fitness function itself may not be. Many fitness functions contain binary valuations. For example, we will see in both case studies that utility is only gained when the vehicle and/or some objective area is sunlit. This type of yes/no decision creates non-continuous steps in the fitness function where the gradient will not exist.

The complicated constraints that could (and will be imposed in the case studies) also present challenges to traditional methods. For gradient-descent algorithms, a penalty function or a method to determine feasible descent directions would need to be devised. With a penalty function, the human designer is then required to determine penalty weights, which may be different between different types of constraints. The

relative weighting between different constraints would implicitly apply a judgement value on which constraints should be prioritized.

It should also be noted that the problems explored in this thesis are co-optimizations of several smaller problems. Take for example, attempting to optimize the maneuvers a ReCon constellation should undertake to image a set of targets. First, we must decide which subset of targets we should attempt to image, as it might not even be possible or optimal to image them all. Then, for each selected target, we must assign some subset of satellites to maneuver. The choice is complicated since the targets are diverse in time and space, thus multiple targets could all be assigned the same satellite, but only if the spatial and temporal considerations allowed. Then, finally we can begin to optimize the maneuvering of the satellites for each target, keeping in mind previous maneuvers can affect the performance and feasibility of future maneuvers, and that the optimal solution might not have a one-to-one relationship between targets and a maneuver. It is totally possible a single maneuver can be used to improve collection over multiple sites simultaneously. Conversely, a single target-satellite pairing might require multiple maneuvers to realize maximal utility. It is probable that each of these steps are tractable for traditional methods, but to co-optimize them together introduces significant complexity, if it is even possible. GAs, on the other hand, are relatively simple to code and design, and are less stymied by the co-optimization of selection, assignment, and maneuver. Figure 2-1 illustrates how the general maneuver problem is a combination of several sub-problems.

As discussed in section 1.4, there exist multi-objective optimization GAs which produce an approximation of the entire Pareto front of maneuver sets in a single optimization run, as opposed to having to fold a single-objective method into a higher-level algorithm such as weighted-sum. This can be significantly more time-efficient, and absent any guidance from a human designer, there is no way to choose a single non-dominated solution as "the" answer, so producing the entire trade-off curve is necessary.

Finally, the existence of schemes like variable-length chromosomes as described in section 1.4.3 is very attractive. In order to allow a more traditional method access

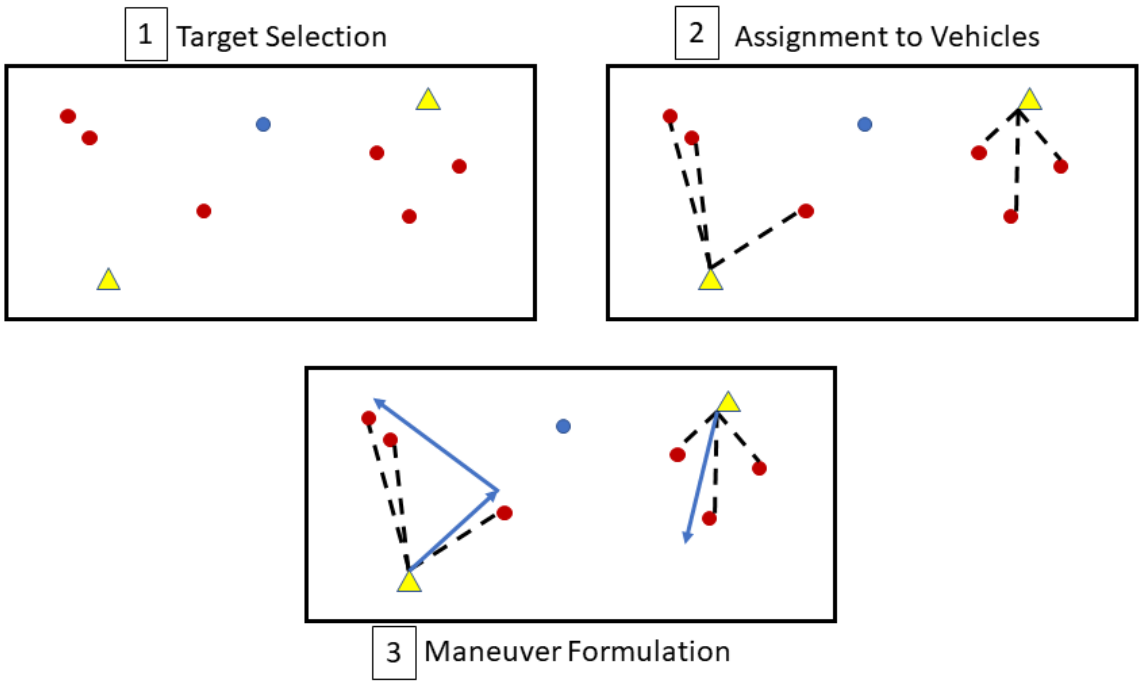

Figure 2-1: A notional overview of how a non-GA approach might break up the vehicle maneuver problem. Circles represent target sites which relate to objectives, and the triangles stand in for vehicles. In step 1, a subset of the target sites are chosen to be considered. Step 2 is an assignment problem; deciding which vehicles should collect on which sites. Step 3 is the actual maneuver problem. A planner must not only determine how many maneuvers to undertake but also the optimal parameters for said maneuvers.

to different numbers of maneuvers, a designer may add in large numbers of design variables, with the thought that the design variables corresponding to superfluous maneuvers will simply be zeroed out and ignored by the optimizer. This is wasteful in terms of both memory and computation time. The fact that GAs can allow the optimizer to choose the number of design variables, allows the designer to remove some human judgement on the form of the solution without sacrificing memory and computation time.

2.2 VLC Formulation

In the framework developed for this thesis, an *individual* in the *population* is described by a *chromosome*, a vector of *genes*. The goal of the framework is to produce a

Pareto front of chromosomes which optimize some set of objectives specific to the problem. Each gene in a chromosome describes a maneuver the network of vehicles can undertake. It is not required that a maneuver in a single gene only involve a single vehicle, but all the problems explored in this thesis formulate their genes in that manner. As the specifics of a maneuver are very problem-dependent, the framework itself does not rely on any specific form. A designer simply need to wrap whatever data structure they have devised in a Julia (the computer language used for the framework) type and declare it a subtype of the Gene type provided by the framework. The fields inside a single gene will only be touched by other problem-specific functions.

The designer will also need to provide an ordering to the gene type. The ordering is providing by defining a "less-than" operator. The reasoning for the ordering requirement is discussed in section 2.6.4, and relates to the performance arising from crossover. A problem-specific mutation operator is also required which acts on single genes. The ability to take in a problem specific gene mutation operator allows the designer to tailor mutation to problem specifics. I use this specialization in both case studies to limit mutation to the feasible subspace of maneuvers.

Finally, it is also necessary to define equality operators for genes. This comparison operator should identify when two genes are redundant, that is, when removing one would have no effect on the simulation and/or performance of an individual. Often, this equality comparison will simply be a check of each gene field one by one for individual equality, however, there can be formulations where the combination of two identical genes actually does effect the phenotype of an individual, so this equality operator may not be straightforward. The purpose of this equality comparison is for use in a reduction operator described in the next section.

2.3 NSGA-II and Multi-Objective Formulation

The main optimization loop of the framework roughly follows NSGA-II [35]. Given a population (a vector of chromosomes, each itself a vector of genes), and a fitness

function, the framework will score the population, before applying the non-dominated sort described in section 1.4.2. The fitness function must return a type that is a subclass of the "FitnessScore" type provided by the framework. This return structure must conform to certain structural requirements, as it contains the objective values that the non-dominated sort uses as input.

The selection operator used is tournament selection, with pairs of individuals being matched up. The number of tournament rounds is a parameter provided by the user. After tournament selection, the winners are used as parents to breed children equal in number to the original population. The crossover operator pairs up random parents (with replacement), and each pairing generates two children. A random point in each parent's chromosome is drawn, and the chromosome is divided into two. The division is always done at a gene boundary, i.e. no vehicle maneuver is split. It is possible for the zeroth position to be selected in this division, in which case one of the halves will be an empty vector. At this point, the first segment of one parent is concatenated with the second segment of the other, producing two children. As mentioned previously, genes are kept in a sorted order defined by the designer. After crossover, the chromosomes of the children are sorted according to this order (as while the individual contributing segments from each parent are internally sorted, their concatenation may not be). This crossover operator is shown in figure 2-2.

A mutation operator is then applied to the new children. There are three different parameters passed to the framework by the user: (1) chance of a gene mutation, (2) chance of a gene being deleted, and (3) chance of a gene being copied. Each gene in the child chromosome is trialed against these three chances, with copying happening before mutation. When a gene is selected for gene mutation, the problem-specific mutation function is called on that specific gene to perturb its design values. The chance of mutation is corrected for the length of the chromosome, so the value provided by the designer to the framework describes the chance of a single gene mutation in the entire chromosome. After the mutation operator is applied, the genes are re-sorted according to the problem-specific gene ordering. The mutation operator is illustrated in Figure 2-3.

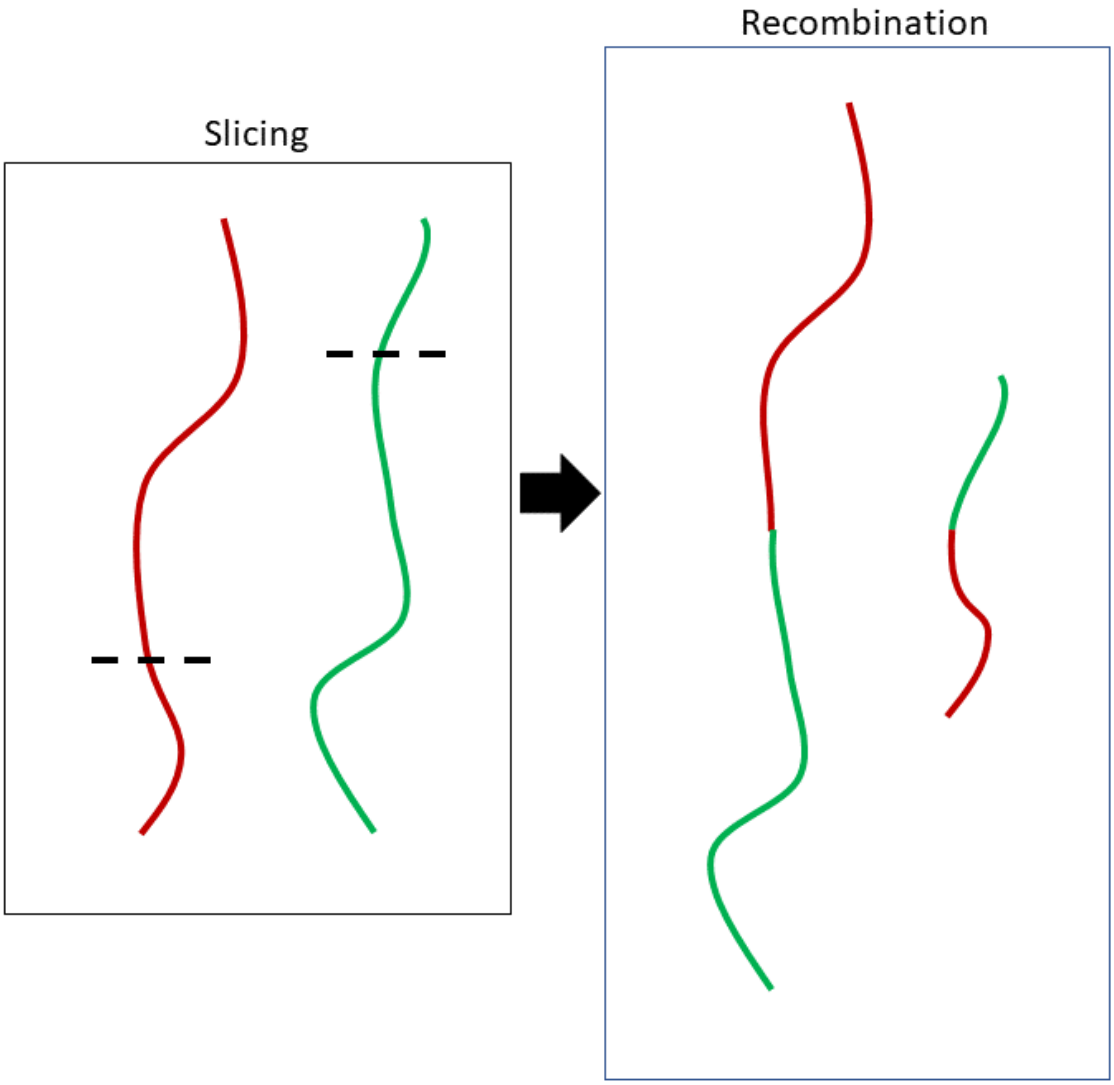

Figure 2-2: The VLC compatible crossover operator used in the optimization framework. Crossover points are chosen randomly along the length of both parents. Then, a first child is made by concatenating the first portion of the first parent's genome with the second portion of the second parent's genome. A second child is made by concatenating the first portion of the second parent's genome with the second portion of the first parent's genome.

Now, as with NSGA-II, the child and parent populations are combined and the combination is put through a non-dominated sort. In this way, elitism is included in the optimization process. Only the top-half (based on performance) of the population is passed onto the next generation. Since the scores for the surviving individuals have already been calculated, the next iteration of the optimizer will be warm-started with

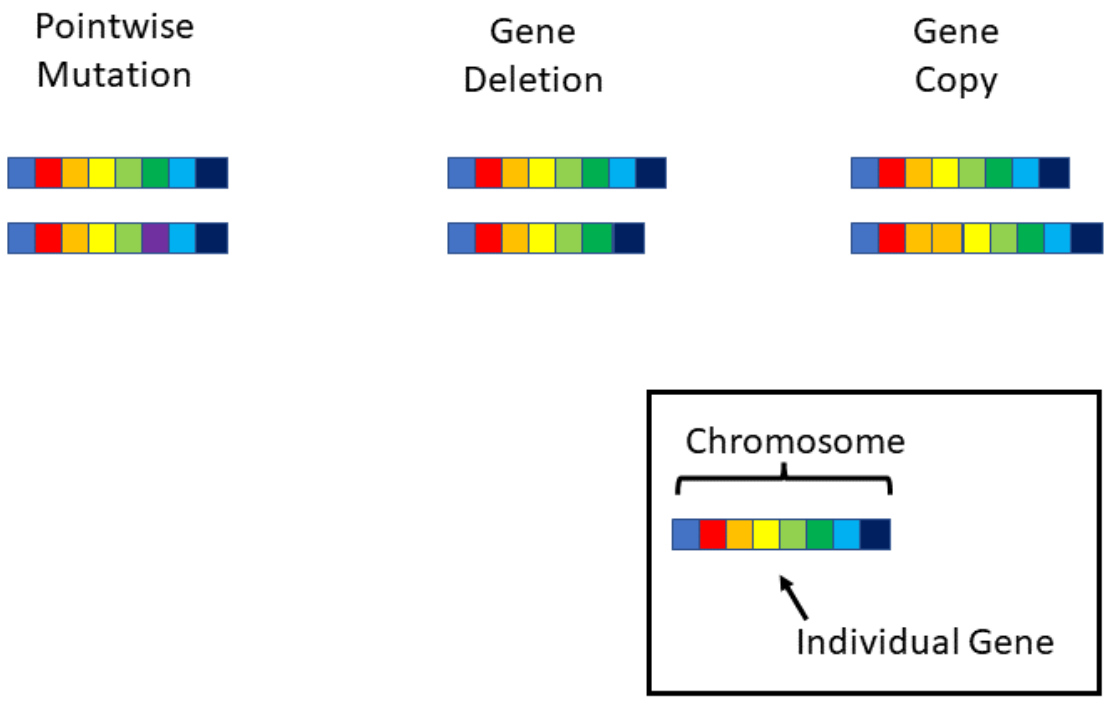

Figure 2-3: The VLC compatible mutation operator. Each gene within the chromosome has a chance to be perturbed, copied, or deleted.

these values. This warm-start effectively halves the number of functional evaluations in an optimization run.

However, before progressing to the next iteration, there are two checks. First, is a check for termination criteria. Termination criteria is based on the average crowding distance in the first front as determined by the non-dominated sorting. A relative percent change and a number of generations are both system parameters. When the average crowding distance has changed by less than the specified relative change for the specified number of generations, the algorithm is terminated. The crowding distance will perturb as the individuals on the first front move around, either from the extremes of the front expanding, individuals spreading across the front, or a higher proportion of the population entering the first front. This termination criteria is loosely based on the one MATLAB uses in its implementation of NSGA-II [3]. An example of the evolution of the crowding distance used as termination criteria is presented in figure 2-4.

If termination is not declared, then a reduction operator is possibly applied. The

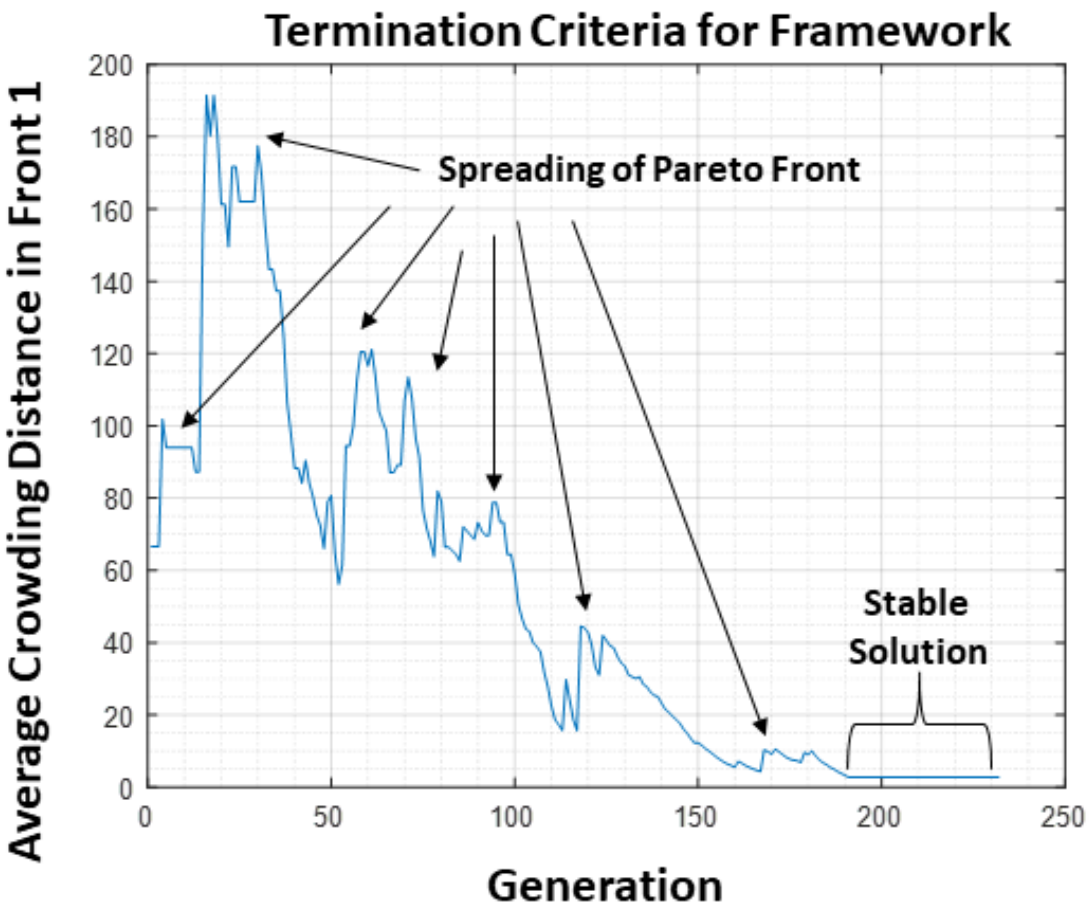

Figure 2-4: This is an example of the metric used for termination criteria as it evolves over an optimization run used in the first case study. The value of the metric is the average crowding distance for those solutions in the first front. This metric will change as more individuals join the front, the front expands, or the front becomes more evenly sampled. Convergence of the algorithm results in a flat-lining of this metric.

number of generations before a reduction operator is applied is a system parameter. If reduction is due, the operator uses the problem-specific gene equality operator to search for identical genes. The equality operator should only return true if identical genes will not have an effect on the system (i.e. when the extra copies will be ignored). The non-active genes will be removed from the chromosome. Reduction can be disabled by simply providing an extremely large number of generations as the period, although certain sets of mutation parameters may cause large amounts of growth in the length of chromosomes, Such long chromosomes can slow down functional evalu-

ation of the fitness function significantly. The overall outline of the algorithm used in this optimization framework is detailed in figure 2-5.

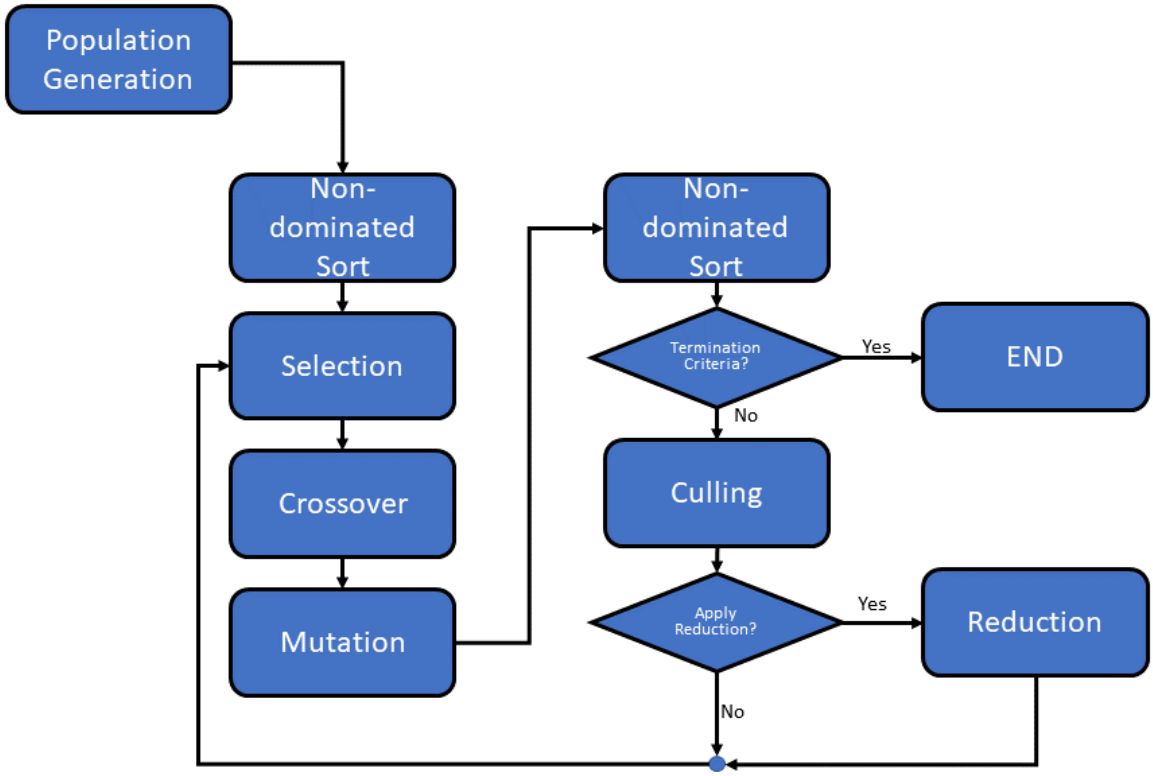

Figure 2-5: The high-level process flow of the framework. The high-level algorithm continually breeds new generations until a termination criteria based on NSGA-II's non-dominated sort is met. Periodically, a reduction operator is applied to the chromosomes to prevent unimpeded growth of inactive genes.

2.4 Specialization to a Problem

To summarize what needs to be done to apply the framework to a specific problem domain, the designer must:

- Formulate a "gene" data structure which describes a single maneuver and can be read by the fitness function. In Julia, this type should be declared as a subtype of the VLCGA.Gene type.
- Define a less-than operator for the gene type. The framework will keep the

Table 2.1: Optimization Framework High-Level System Parameters

Parameter	**Description**
Chance of Gene Mutation	Chance a single gene will perturb in a chromosome during mutation.
Chance of Gene Deletion	Chance a given gene will be deleted during mutation.
Chance of Gene Copy	Chance a single gene will be copied during mutation.
Reduction Generation	Number of generations between applications of the reduction operator.
Crowding Distance Tolerance	The minimum percent change in average crowding distance on the Pareto front to be considered different (for purposes of termination).
Maximum Generations w/o Change	The maximum number of generations where the average crowding distance on the Pareto front is less than the Crowding Distance Tolerance before termination is declared.
Maximum Generations	Maximum generations framework will run before terminating.

genes within a chromosome according to this order. In this thesis, I also found it possible design ordering which allowed efficient parsing by the fitness function.

- Define an equality operator for the gene type. This will be used by the reduction operator.

- Define a problem-specific mutation operator applicable to the gene type.

- Provide a fitness function that takes in a single chromosome as an argument. This is where the main numerical simulation will occur. This function should be written to be as computationally performant as possible, and compatible with distributed computing. It is obviously useful to be able to pass scenario data and parameters to a simulation function. Luckily, Julia provides a very simple method for wrapping those extra parameters in a generic function with the simulation function.

- Define a "FitnessScore" type which conforms to the structure expected by the non-dominated sorting function (which is simply an array called "objectives"

which contains the performance on the multiple objectives). The fitness function should return one instance of this type.

Examples of these specializations can be seen in the code listings in Appendix A.

2.5 Code Efficiency and Simulation Performance

The use of any genetic algorithm based approach will require significantly more functional evaluations than more traditional methods. It was quite clear from the start, that in order to keep the framework useful, a significant effort would be required to keep the simulations in both case studies, as well as the overall optimization framework, as performant as possible, lest the optimization become computationally intractable. The effort to optimize the framework and simulations can be divided into two areas. First, the code on a single-threaded level was developed to be as performant as possible. Second, the use of distributed computing techniques was employed in a massively parallel manner. I hope that the discussion here can be valuable to readers attempting to create their own optimization and simulation code.

2.5.1 Code Optimization

The very first choice made when developing the framework was which language to use. Computer programming languages are not just different ways to express the same set of instructions to a computer; their very structure and design choices greatly affect the final runtime of the deployed code. On the other hand, this desire for performance must be balanced by ease-of-development. There are surely lower-level approaches that could be taken which will squeeze out more performance than I have obtained, but they probably would have taken more time than was available to me, and would not be very accessible to others, which would greatly reduce any potential contribution the framework could make.

The obvious choice of language and development platform is MATLAB [9], as it is a higher-level language with a large collection of scientific libraries, and in my

personal experience, very widely used in the Aerospace industry and academic circles. The second choice to consider was Python [20], another higher-level language with good library support and widespread use. Also, Python comes with the advantage of being free, and thus inherently more accessible than a commercial product like MATLAB.

Both of these choices are dynamically-typed languages, and that comes with significant performance hits. Unlike statically-typed languages such as C, the types of variables are not known until runtime, and this requires the addition of much boiler-plate processing to check the types and sizes of variables at numerous points in code. Compare this to a statically-typed compiled language, where the types and sizes of structures are known at compile-time, enabling many compiler optimizations developed over decades to be applied. There is also the added quibble of how these languages are translated to machine code. MATLAB currently uses just-in-time (JIT) compilation, producing machine code (specific to the CPU platform it is running on) to be run directly by the processor [10], but Python is "compiled interpreted," meaning it is translated to an intermediate byte code which is then interpreted by a virtual machine [7], an approach that generally is slower.

So, is there a language that combines the speed of a statically-typed compiled language like C with the high-level expressiveness and scientific support libraries common to MATLAB and Python? Julia is a language that attempts to merge those two items. Julia, is high-level, dynamically typed, but still compiled, language which aims for the speed of C, while having an ecosystem of scientific support libraries available. In fact, it was the aim of the original developers to include these characteristics from Julia's inception [22].

Julia provides a set of benchmarks comparing its performance to various other scientific computing languages, most notably Python and MATLAB, and it indeed does appear to significantly outperform those two [8]. Figure 2-6 compares Julia's performance to other common languages used in scientific computing.

The full discussion of how that performance is obtained is beyond the scope of this thesis, so I will provide a quick summary of the designer's rationale [25]:

Figure 2-6: Benchmarks of Julia versus other common languages for a variety of tasks. Adopted from [8] under the MIT License. Copyright (c) 2011-2019: Jeff Bezanson, Stefan Karpinski, Viral B. Shah, and other contributors: https://github.com/JuliaLang/www.julialang.org/contributors

- While dynamically typed, a large amount of typing information is provided to the compiler via Julia's use of multiple dispatch. Instead of placing boiler-plate code to check the types of variables at runtime, Julia simply compiles different machine code instantiations of a method for each set of input (and output) types it encounters at function call. It then simply calls the corresponding function.

- Julia will aggressively specialize code against types of variables found at runtime, highly optimizing machine code for specific sets of types. This works in synergy with first item.

- JIT compilation using Low Level Virtual Machine (LLVM) [56]. LLVM is an intermediate code representation and compiler toolchain used in compiling several languages, including Ada, C, and C++. Once translated to LLVM intermediate code, Julia programs can leverage all the development and optimization

research that has been incorporated into LLVM over past 16 years.

The main performance regret is that calling a function with a novel set of types will require a new run of the compiler. Thus, if one function is called many times with different user-defined types every single call, performance will be quite degraded. However, that use case is rare, and basically non-existent in the framework presented in this thesis. In the course of an optimization run, functions will be called with almost exclusively one set of variable types, meaning most code will be compiled exactly once, and thus the framework benefits from the aggressive machine code optimization without incurring the penalties.

Of course, some effort is required on the part of the developer to best exploit these advantages. The code in this framework and the simulations associated with the two case studies was written to follow the performance guidelines outlined in the Julia manual [13]. Several profiling studies were undertaken to identify inefficient code segments and re-code them according to best practices. The most significant performance increases came from:

- Global variables were avoided. Instead such values were passed as parameters from function to function. This provides a performance gain as the memory location of globals cannot be hard-coded into functions, and thus require an indirect look-up when used.

- All fields in user-defined structures were concrete rather than abstract fields. Concrete fields have known memory sizes. For example, a double precision float is concrete as it is known to take 4 consecutive bytes to store. Examples of abstract types would be a superclass such as "Real". The compiler cannot know the size of the field at compile-time as the "Real" class contains single precision, double precision, and integer types of various lengths. Knowing the memory length of the structures at compile time enables many optimizations and possible placement on the stack rather than the heap.

- Static arrays were used rather than "Vector" and "Matrix" types. Like MATLAB, vectors and matrices in Julia can be different sizes, and can grow or

shrink. Using an explicitly sized array type provides explicit memory-length information at compile time, enabling optimization.

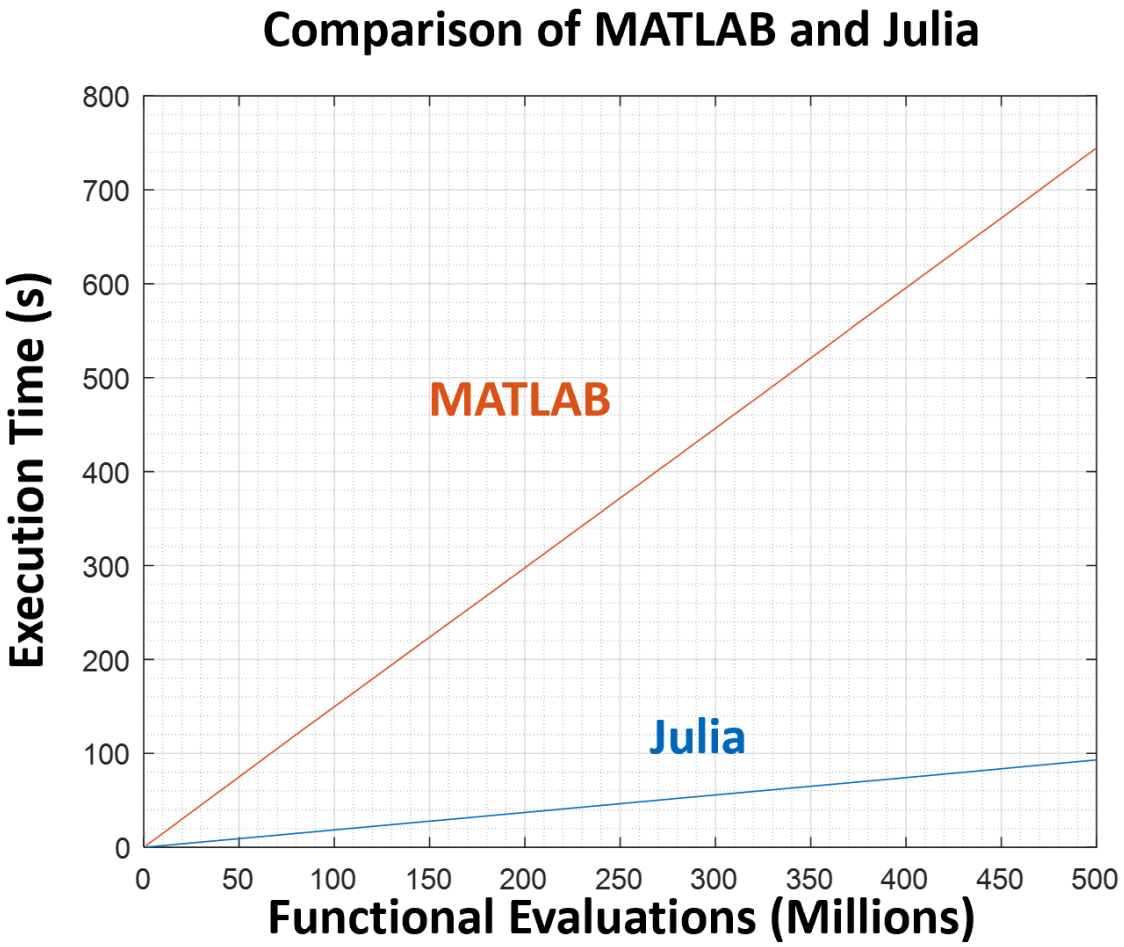

Figure 2-7: Julia generally outperforms MATLAB in terms of computational efficiency. Here is shown the performance of an algorithm for determining if a point is inside a closed polygon, which will be a core part of the simulation in case study 2. The algorithm is implemented as close as possible between MATLAB 2021a and Julia 1.6.1. Julia performs about 7 times as better on this particular function. Evaluation was done on a single Xeon Platinum core on the LLSC grid.

The magnitude of improvement that can come from these somewhat simple optimizations is quite significant. As an example, a function to determine if a point is contained within a given polygon was written for the second case study. Simply changing out the use of vectors holding 2D points for fixed-sized array equivalents improved simulation run time by 18 times. That improvement enabled much quicker

iteration in code development and optimization runs. Well-written Julia code generally significantly out-performs MATLAB code. Figure 2-7 shows a comparison between MATLAB and Julia on this polygon algorithm used in the second case study. That is only one specific algorithm, but the result holds true in general.

Beyond the performance benefits of Julia, the language also affords some benefits to development time. Notably, the Julia standard library has been built from its inception to support both traditional multi-threading as well as distributed cluster computing [11]. This streamlined the use of supercomputing resources to enable massive parallelism in the framework.

2.5.2 SLURM Cluster Computing

GAs are a great candidate for massively parallel computing. In a generation, the fitness function evaluations for every individual can be evaluated simultaneously. Thus, GA runs with large populations can greatly exploit the availability of additional compute resources. However, the size of the problem generally is large enough that simple multi-threading on a single shared memory computer is insufficient. A large distributed computing solution with many more Central Processing Unit (CPU) cores is needed. Spreading computation over discrete memory domains complicates the problem, as now data must be sent and collected between multiple nodes. Such a data transfer scheme adds both development time and latency.

The supercomputing resources available to me were provided by the Lincoln Laboratory Supercomputing Center (LLSC) thanks to my employment at Lincoln as a member of the technical staff. This computing cluster is very similar to the MIT Supercloud [6], very much being the same thing but with greater numbers of CPU cores available for an individual user. Both clusters use the open-source SLURM scheduling system [16] to launch and manage parallel jobs. Thus, the code developed here for the LLSC, can be used with minimal reconfiguration on the MIT Supercloud.

The use of SLURM on the computing cluster combines well with the choice of Julia as the implementation language. Julia has built-in support in its standard library for distributed computing solutions, including SLURM [12]. With Julia al-

ready providing a framework for distributed computing (which leverages its included serialization and efficient network message passing) it was fairly simply to adapt the VLC GA framework to a distributed environment. Contrast this with the solution for distributed computing on MATLAB, the pMATLAB library [14], which is not built into the base MATLAB language or its standard library, and uses the file system for message passing (much less efficient than pure network communication).

The LLSC provided me with concurrent access to over 3000 Intel Xeon Platinum 8260 cores for use during optimization runs. I used 2048 in practice, as the population size which is a power of two works elegantly with multiple rounds of tournament selection. The architecture for distributed computing is shown in Figure 2-8.

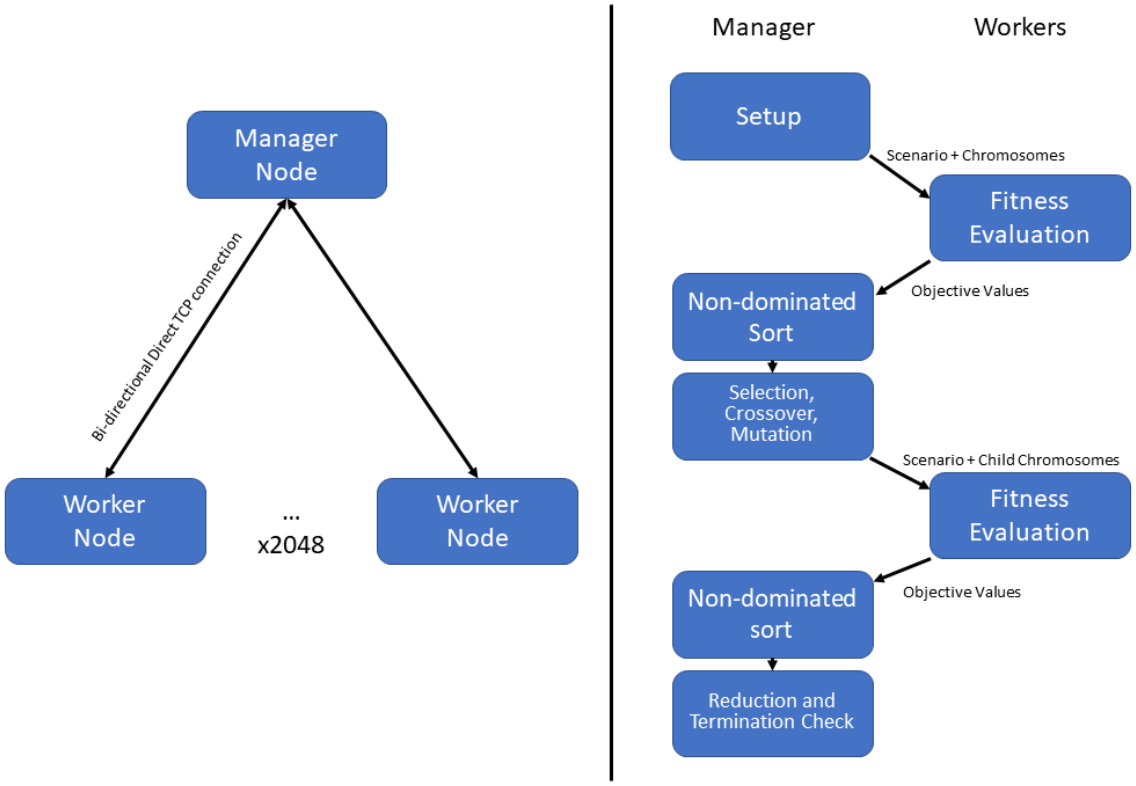

Figure 2-8: The distributed computing architecture utilized by the optimization framework. [LEFT] The communication network is very simple, each worker maintains a bi-directional TCP connection with the manager node. [RIGHT] The transfer of data between the manager and workers for the first generation of an optimization run. Workers are reserved for functional evaluations of the fitness function on a massively parallel scale.

A manager node is responsible for sorting, selection, crossover, and mutation,

Table 2.2: This is measured timing data of functional evaluations for some of the case study fitness functions used in this thesis. The time for a single functional evaluation was averaged over 10 samples run on a single core, identical to that provided in the SLURM cluster used for the optimization runs. A theoretical single core time for a 2048 individual population (what was used for these studies) and a termination of 500 generations (fairly representative) is provided as context to the necessity of distributed cluster computing.

Scenario	**Single Functional Evaluation**	**Theoretical 500 Generation Time**
ReCon 2 Day Global Scenario	14.1s	168 days
ReCon 14 Day Regional Scenario	12.8s	152 days
PEARL Atlantic Transit	5.1s	60 days

while functional evaluations of the fitness function are distributed among the 2048 worker nodes. The vast majority of the computational effort in both case studies was the functional evaluation of the fitness function, so this scheme best exploits the computing resources. Devising distributed sorting and breeding algorithms would require a fair bit of complexity, and may not even reduce total run time, as network communication latency might possible be larger than the gains from parallelism. Serialization and message passing was opaque to me, as I relied entirely on the built-in distributed computing API from Julia to handle those areas.

The benefits of this massive parallelism were significant in the course of the optimization runs. Although each individual functional evaluation was on the order of seconds, the large population size and the required number of generations meant that each run would take weeks or months to converge on a single-core! An estimate of such times can be seen in table 2.2. In practice, these runs were individually on the order of hours. Here the speedup arising the choice of Julia show its worth, as hours versus days is still quite significant at the lived human scale.

2.6 The Rain Catcher Toy Problem

In developing the framework, I postulated a simplified one dimensional problem to aid in coding, debugging, and validation of the framework, as well as to to act as an instructive worked example. There were a number of key components I thought this toy problem should have to match as general a vehicle maneuver problem as possible:

- The scenario should contain multiple agents (i.e. maneuverable vehicles). This means the assignment problem is not trivial and there is possible synergy between different vehicles.

- There should be multiple objectives. The framework is meant to solve multiple objective problems, and not just single objective situations. Multiple objectives exercise the NSGA-II component of the framework and its ability to trace out an approximated Pareto front.

- At least one of the performance metrics should be based on the maneuvering of the vehicle. Obviously, there is no point in maneuvering the vehicles if there position/state has no effect on the final disposition of the objectives. There should be multiple collection "sites" diverse in both time and position, in a way that necessitates maneuvering the vehicles to actually collect on different sites.

Taking into account these elements, I formulated a simple problem that I am calling the "Rain-Catcher" problem. In this problem, a number of maneuverable rain catchers with finite length exist inside a one dimensional space. Over the time of the scenario, rain drops fall at randomly chosen locations across this one dimensional space. A rain drop is considered "caught" if at the time of its fall, its position is within the interval of at least one of the rain catcher platforms. The objectives are to maximize the number of caught drops while minimizing the total cumulative movement of the platforms. With this setup, we have multiple maneuverable vehicles (the rain catcher platforms) attempting to collect on multiple "sites" (the rain drops) diverse in time and position. Assuming the widths of the platform are small compared

to the width of the total problem domain, we have a situation where the platforms must maneuver in a non-trivial way to increase the number of collected drops.

2.6.1 Formal Definition

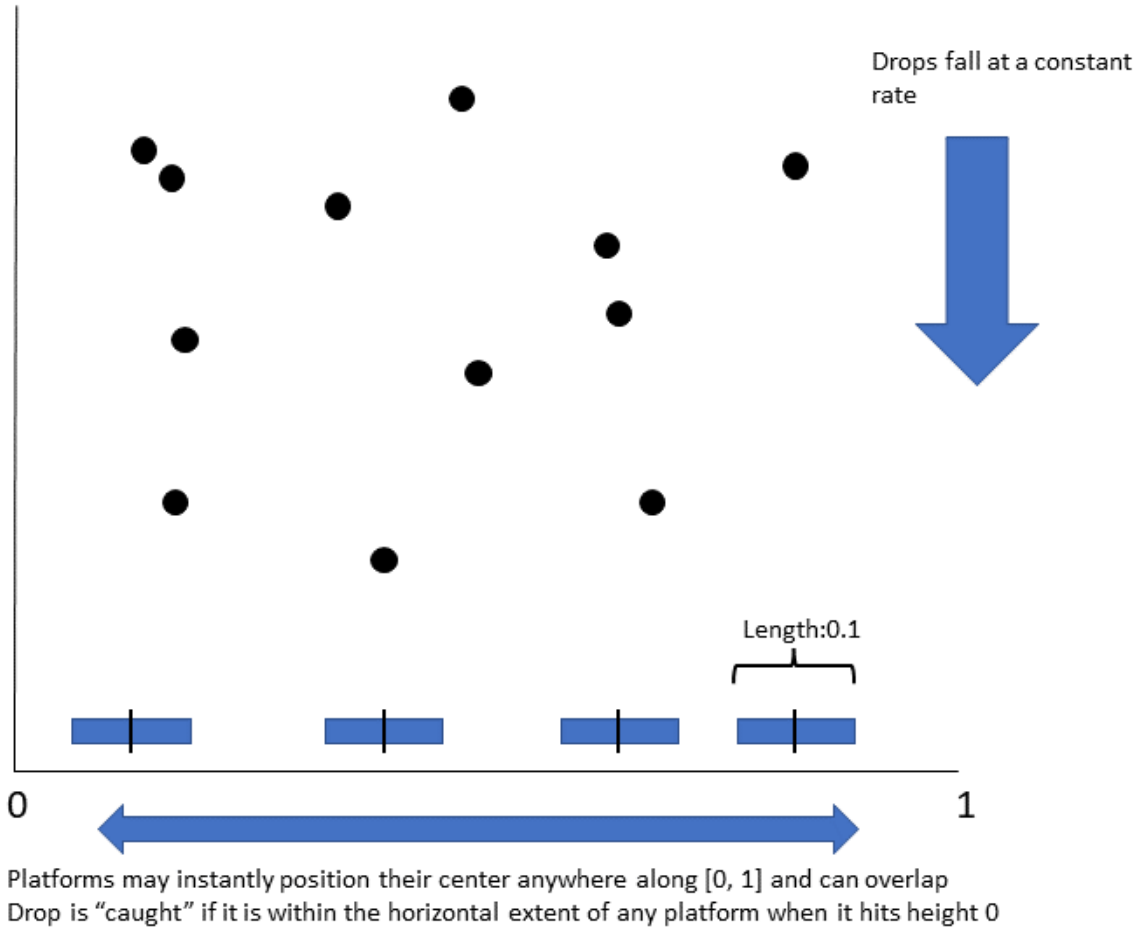

Figure 2-9: The 1D vehicle maneuver problem of catching rain. Four platforms of width 0.1 are allowed to re-position along the [0,1] interval where rain is falling. The objectives are to catch as much rain as possible while minimizing the total distance travelled by the platforms.

The specific setup of the modelled scenario is as follows and is shown in figure 2-9. Four maneuverable platforms of width 0.1 exist along the interval of [0,1]. Each platform can be re-positioned such that the center of the platform can be placed anywhere on the the [0,1] interval. The transit of the platforms is instantaneous. All four platforms start centered in the interval.

Table 2.3: Rain Catcher Gene Formulation

Field	**Description**	**Limits**	**Data Type**
Time	The time of a maneuver	[0,100]	Float64
Coordinate	The location to place the center of the platform at	[0,1]	Float64
Platform	Identifies which platform to move	[0,3]	Int64

A program of 50 rain drops was randomly drawn, with locations drawn from a uniform distribution on the [0,1] interval. The times of drops were chosen from a uniform distribution of on the interval [0,100]. The amount of drops is significant. This is large enough that four platforms face a challenge collecting all the drops, but not immense enough that the law of large numbers becomes overwhelming. In which case the trivial solution of simply spreading the platforms the minimal distance required to not overlap is the best solution, and can be reasoned straight from intuition.

2.6.2 Formulation of Genes and Scoring

Each gene in the chromosome contained 3 data values: (1) A platform number; a categorical variable digitally represented as an integer of 0 to 3, (2) a time of maneuver, represented as a positive double-precision float, and (3) a position represented as a double precision float. In simulating the scenario, the indicated platform is instantaneously moved at the time of maneuver to the location specified.

A sorting order was defined for the genes. Genes within a chromosome are simply sorted ascending by time, and for maneuvers at the same exact time, then sorted ascending by platform number. There is no sorting beyond that, so this is not a stable sort. If there happen to be multiple genes with the same platform number and time of maneuver, sorting position would be determined by original placement position in the chromosome vector. In practice, this situation has basically zero probability to occur, as the mutation operator perturbs all values of a single gene at once. A gene might be an exact copy of another and thus inactive, but as soon as mutation causes the position to change, the time is also perturbed.

The mutation operator was simple. All three values of a gene (platform number, maneuver time, and maneuver position) are randomly re-drawn from their valid distributions. This had the advantage that genes would never go infeasible. I use this strategy for mutation (where the maneuver is kept feasible, and/or infeasible maneuvers are simply ignored in simulation) in both case studies. It avoids the requirement of having a human-crafted penalty function, which would then necessitate a weighting of different types of constraints.

The reduction operator was similarly trivial. Any genes past the final rain-drop time are removed from the chromosome. Any set of identical maneuvers (in platform, time, and position) were culled such that only one instance remained. This reduction operator thus simply removes all genes that are inactive. There are significant memory and computational time benefits to removing these inactive genes, as many begin to pile up over subsequent generations. Not only does this obviously require more memory to store, but processing also becomes significantly slower. The simulation in the fitness function still must parse through inactive genes, and memory transfers of larger sizes significantly slow as they exceed CPU cache sizes.

Finally, a simple operator was defined for determining solution domination. Less total distance travelled by the platforms is considered superior to more distance, and more captured rain drops is considered superior to less. For example, a solution will dominate another if it catches more drops without the 4 platforms travelling more cumulative distance.

2.6.3 Simulation and Results

Since rain drops instantaneously fall, and rain catcher platforms instantaneously maneuver, the simulation of the problem can be done with infinite temporal precision. The simulation is initialized by placing all the platforms at the center of the interval, and then advanced to either the next rain-drop time, or the next maneuver time, whichever comes first. The genes of a chromosome being sorted by maneuver time is computationally efficient, as we can find the next maneuver (or rain drop time, as that vector is also time sorted) in constant time, by simply storing two indices of

the last drop and maneuver effectuated. Even without these indices, the time sorting would enable a binary search for the next valid event in logarithmic time, as opposed to searching an entire unsorted vector.

If the next event was a maneuver, the simulation simply updated the position of the indicated platform, and added the distance said platform moved to a bookkeeping variable. If the event was a raindrop, the position of the drop was checked against the position and extent of all four platforms. If any of the platforms contained the drop position, a different bookkeeping variable was incremented.

After the last rain drop falls, the simulation returns the total distance travelled cumulative among all four platforms, and the total number of rain drops collected as objective values. For reasons that will be explained in 2.6.4, any chromosome that was empty, i.e. no maneuvers, was artificially penalized in these values so as not to be Pareto optimal.

The rain "program" of drop times and positions was pre-generated beforehand. The framework found solutions specific to this rain program. It would be possible to create fitness functions that score the chromosomes against various different Monte Carlo rain programs, but at that point, the problem begins to approach the law of large numbers limit for a random uniform distribution, with the previously mentioned trivial solution.

The initial population is a set of 10000 single maneuver individuals. The mutation operator makes this initial population easy to generate. One simply creates a gene with all values zero and applies the operator once. The simplicity of this simulation removed computational efficiency concerns, and this toy problem was simply run on a single core on a desktop computer, rather than the LLSC grid.

The Pareto front generated by the framework is shown in Figure 2-10. From this front, it can be seen that the optimization framework properly converged to a convex set of solutions, all of which are non-dominated relative to each other. As one would expect, the number of drops increases as more distance is travelled by the platforms. Since there was no constraint on the total distance the platforms could move, the Pareto front extends from the static case to the case where all fifty drops

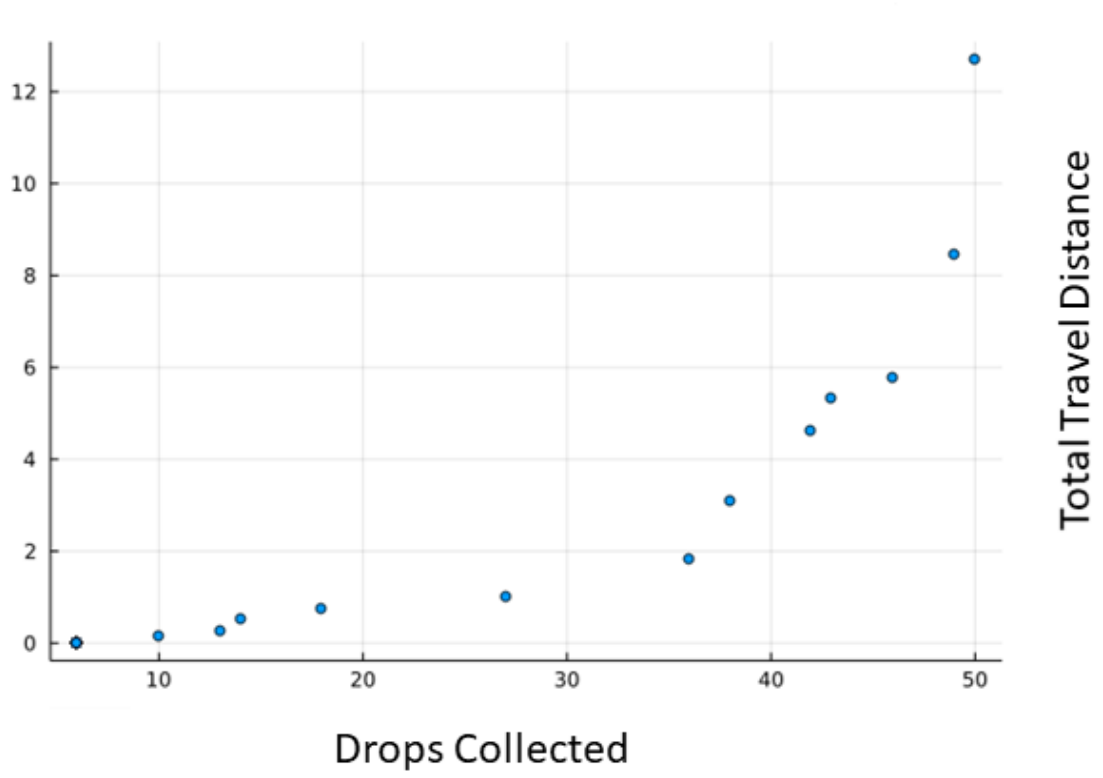

Figure 2-10: The Pareto front as approximated by the optimization framework. As expected, as there was no total constraint on platform movement, it is possible to generate a solution which collects all 50 drops, and the Pareto front extends to that extreme.

are collected.

2.6.4 Lessons Learned

I found a number of useful lessons learned during the development of this toy problem, and the framework went through a fair number of iterations to incorporate them. I will now briefly describe the most significant improvements.

Penalty Application to the Empty Chromosome Case

As mentioned in the discussion on simulation, the static, no-maneuver case was artificially penalized so that it would not be Pareto optimal. I found that when this modification was not present, the static case very often began to dominate the population, especially early on. This degraded convergence and the quality of the spread

of solutions over the front. Figure 2-11 shows an example of this effect.

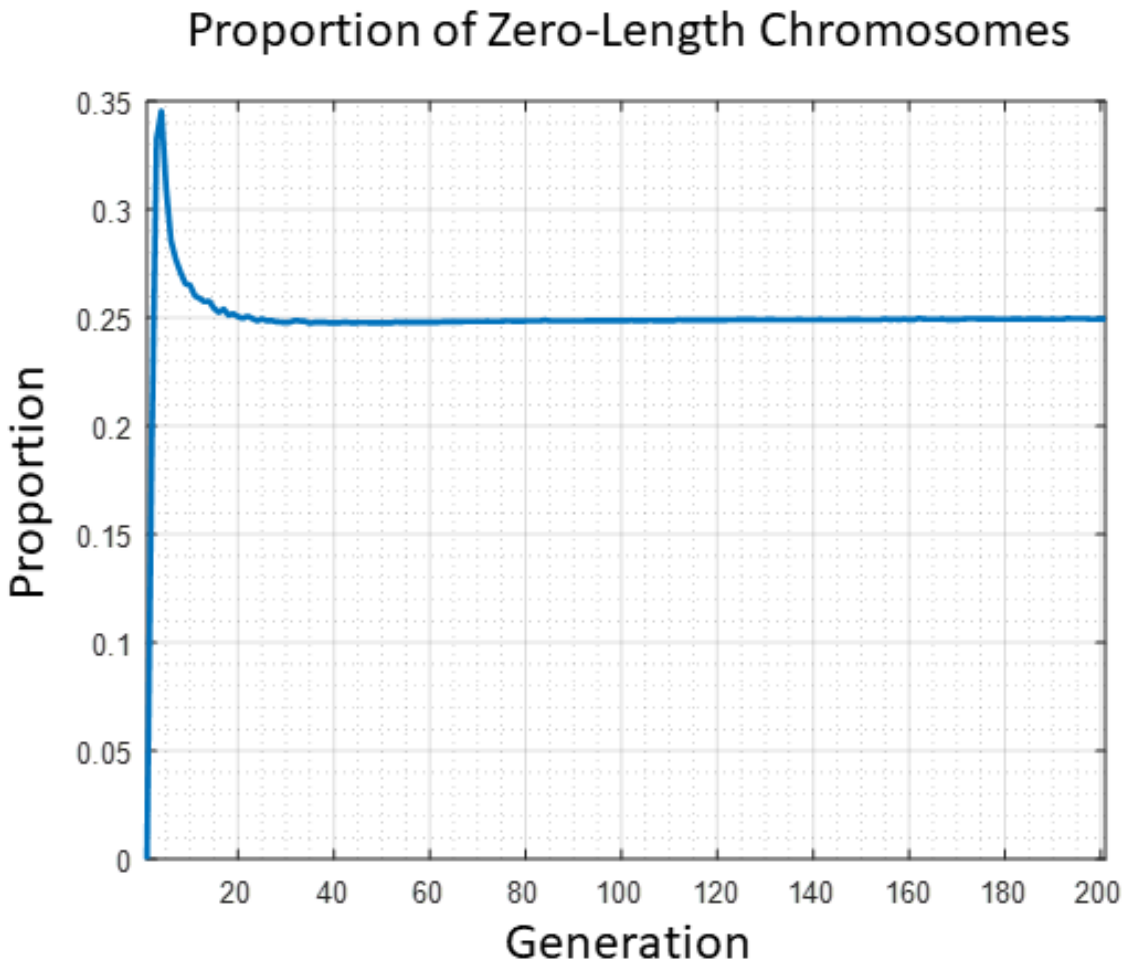

Figure 2-11: The proportion of individuals in the population with length-zero chromosomes traced out for a run of the Rain-Catcher problem. In this case, and many similar cases where a fuel or fuel-analogue is an objective, the non-maneuver case is Pareto optimal and dominates a large proportion of the population.

The static case is clearly Pareto optimal, because in this scenario there is no possibility to move "negative" distance. Thus, the static case always exists in the first front from the NSGA-II non-dominated sort. In addition, one instance of the static case would always be the extrema of the approximated Pareto front (as it is in fact the extrema of the actual Pareto front), and would be absolutely guaranteed to advance to the next generation. Any static-case individual in the population would be preferred in tournament selection to any individual not in the first front, and in the early generations, a large proportion of the other individuals are not in the first

front.

The mechanics of mutation and crossover also contribute to the domination of the static case. An empty maneuver case cannot mutate. There are no chromosomes to perturb, and no chromosomes to copy. On the other hand, single-maneuver chromosomes do have a chance to delete their last remaining gene and become static cases. As the initial population consists of one maneuver individuals, the chance of this occurring is elevated in early generations. Crossover also has an elevated chance of producing zero-length chromosomes early on. When one parent is a zero-length chromosome and the other is a single maneuver individual, then one of the children is guaranteed to be a zero-length individual, as dividing a zero-length chromosome in two generates two zero-length gene fragments, and dividing a single-maneuver chromosome generates one zero-length fragment.

There are multiple ways to address this problem, mostly by adding logic into the mutation and crossover operators to prevent the creation of zero-length individuals, but it is quite simple to just penalize the static case such that is not scored as Pareto optimal. It is then trivial effort to add back the static case into the final Pareto front traced out by the framework.

Stochastic vs Spectral Maneuvers

In the presented form of the problem, each gene contains the absolute time of the maneuver. Originally, I had this value indicate a relative time from the previous maneuver (with this setup, genes were not sorted by time within a chromosome). This was not optimal for convergence of the GA. For a fairly well-developed series of maneuvers, optimal for some later part of the scenario, but missing some earlier drops, it became very unlikely that the series could be combined with additional maneuvers which do catch those early drops. The addition of any earlier maneuvers would change the timing of all the later maneuvers, almost guaranteeing they become non-optimal. As the individuals grew in number of maneuvers this became more problematic, as any addition of a gene or mutation in a gene would affect all later maneuvers.

The problem was solved by formulating the genes to use absolute time rather than relative time as design variables. (No change to the other variables was necessary, as both platform number and position were already formulated in an absolute manner.) From this experience, I extrapolate a general principle: it is best to formulate maneuver genes in a way such that a change in one gene, or the addition/deletion of a gene, minimally affect the utility of other genes. To borrow terms from mathematics, I would place all gene formulation options on a continuum from "spectral," where any change will affect the utility of all other maneuvers, to "stochastic," where only the immediately following maneuver (or possibly no other maneuvers) are affected.

Earlier in section 1.4.1, I briefly discussed the building-block hypothesis (BBH), a possible explanation for the ability of GAs to find optimal solutions. This difference in performance between a "spectral" maneuver formulation and a "stochastic" formulation can be contextualized with the BBH. The BBH only works if one can combine high-utility building blocks together without destroying their utility. Clearly, a spectral formulation has a significant chance of making such combination difficult.

Sorting Order on Genes

Originally, there was no ordering enforced on the genes inside a chromosome. Some sorting happened within the simulation code, but no ordering was guaranteed entering crossover. I found this to be detrimental to convergence. It was far better to enforce some kind of ordering among genes within a chromosome than let them lie as is. I chose specifically to order by time and then platform number, as that simplified code in the fitness function simulation, but I find no reason to think a primary sort of platform followed by time would not perform equally as well. Sorting a gene probably results in better convergence for reasons similar to the "spectral" versus "stochastic" maneuver formulation issue. Because the framework's crossover operator makes a cut in chromosomes by array index, any "building block" which describes a set of optimal maneuvers for some portion of the scenario needs to be coalesced into adjacent elements in the chromosome vector. For example, without a sorting, it would be difficult to combine an individual well-optimized for collecting in the first

half of the scenario with an individual well-optimized for the second. The probability of either of those building blocks existing in a contiguous run of array entries which is likely to be taken by the crossover operator in full is exceedingly small without enforced sorting.

Utility of Reduction Operator

When viewed strictly in the light of mathematical formulation, the reduction operator may seem superfluous. Inactive genes will simply be inactive. Since the chance of gene mutation is corrected by the total length of a chromosome, a reduction operator would not even affect the chance of a mutation occurring, just where in the genome it occurs.

In practice though, the reduction operator has significant effect on the computational performance of the simulation. Without the operator, the chromosomes grow quite long, with many copies of genes, most of which are inactive. By the end of a simulation, the active part of a chromosome can be less than a tenth the total length. Fitness functions need to search and parse all of these genes, even the inactive ones, and the high-level framework must manipulate and pass the gene data around. Both areas suffer from such large amount of extra data. The reduction operator keeps the chromosomes in a manageable state, without changing the fitness of any individual. Figure 2-12 shows a comparison between a run of the Rain Catcher problem with and without the application of a reduction operator.

The regret of the reduction operator is that it introduces another hyperparameter that requires tuning by the human designer. How often should the reduction be applied? On one extreme, the reduction operator could never be run, resulting in the performance degradation arising from the large amount of inactive genes. The other extreme, running the reduction operator every generation, stymies the exploration of the search space. To successfully grow a chromosome and find a more performant individual, a gene must both be copied and then mutated. A copied gene might be inactive until a mutation occurs, and so must be allowed to exist long enough to provide it a chance to mutate. Thus, the number of generations between applications of the

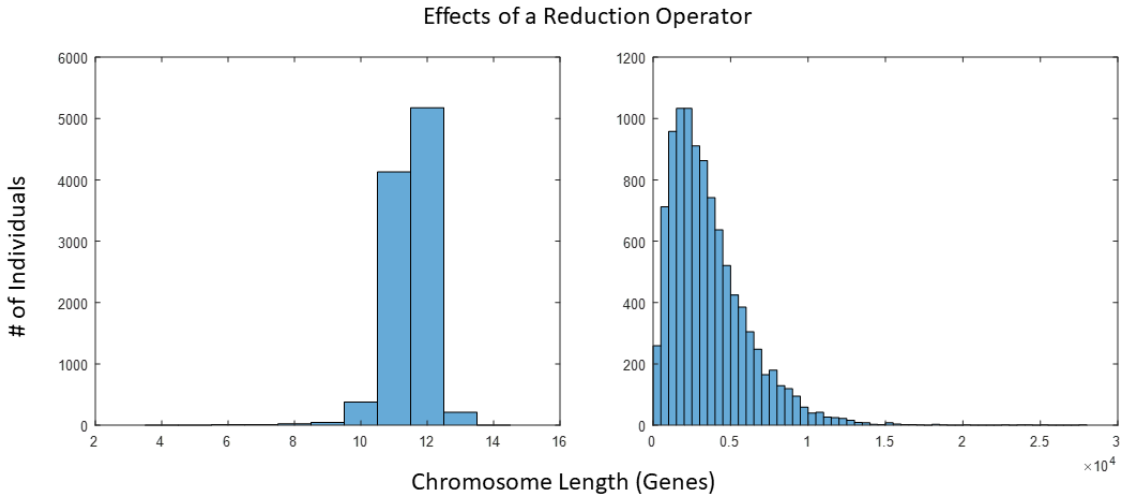

Figure 2-12: The lengths of individuals' chromosomes before and after the application of a reduction operator to an optimized population for the Rain Catcher problem. [RIGHT] The lengths of chromosomes before the reduction. [LEFT] The lengths of chromosomes after reduction. The actual active genes pre- and post- operator are identical.

reduction operator must be tuned between the extremes, taking into considerations the problem-specifics of the mutation operator and chances of mutation.

Applicability to Case Studies

The improvements from these lessons were applied to the two case studies described in Chapters 3 and 4. It was fairly simple to penalize the zero-length chromosome case and impose ordering on the genes for the maneuvers in the two cases (as both had an obvious sorting). The preference to formulate the maneuvers in as close to a stochastic way as possible had significant influence on design decisions. This aspect would probably be the most difficult part of extending the framework to a new problem domain, as it is required not to just frame the design variables in absolute terms rather than relative terms, but to also devise the form of the maneuvering and associated design variables to limit dependence on previous state.

Chapter 3

Case Study 1: ReCon Maneuvering

In this chapter I will present the formulation and specialization of the framework towards the problem of satellite maneuver planning in a ReCon. This scenario differs from the previous works described in section 1.2. In contrast it presents the constellation with a large number of collection targets simultaneously, as well as a more complex performance metric which based on the geometry of multiple passes and models the combination of differing modalities of sensor measurements. This type of scenario was the original impetus for the research in this thesis. In my work in Lincoln's Space System Analysis group, I have often had to model large constellations of space-based sensors taking measurements of sites on the surface of the Earth, but with more complicated modalities then a simple image.

For this scenario, the framework outputs Pareto optimal sets of maneuvers for a hypothetical constellation of imaging LEO satellites. The constellation is tasked with imaging a large number of collection sites placed on the surface of the Earth, diverse in location and time. The number of collection sites is large enough that measurement occurs on multiple sites simultaneously rather than sequentially. The first optimization objective is to minimize the amount of delta-V consumed by the satellites from maneuvers. The second objective is to minimize the position uncertainty of a hypothetical feature at each target site. This is a complex performance metric which will be discussed in section 3.1.

3.1 Photogrammetric Uncertainty as an Objective

While the delta-V objective is quite straightforward, the data collection objective is complex, intentionally so, to showcase the GA's ability to handle such complexity in an objective. The objective is relevant to the field of photogrammetry, the 3D measurement/reconstruction of an object from 2D imagery. The algorithms and approaches for fusing multiple images into a 3D reconstruction are beyond the discussion of this thesis. Instead of modelling uncertainty measurements based on any specific set of photogrammetric algorithms, I use an information-theory based bound. Specifically, I use the Cramer-Rao lower bound (CRLB), which quantifies the best performance an optimal estimator could hope to achieve. In this way, the scenario stays agnostic of the choice of specific photogrammetry algorithms, or any future algorithms which may be devised.

3.1.1 The Cramer-Rao Lower Bound

A lower bound on the variance of unbiased estimators was first described by Rao in 1945 [71]. Any unbiased estimator which achieves this variance is called *efficient* and no other unbiased estimator can obtain less mean squared error. This bound can be applied to estimators which combine noisy measurements (specifically those that behave as random variables). The CRLB has forms which can handle biased estimators, but I assume a notional unbiased estimator for use in this case study.

Consider an unbiased estimator case, whose input is n independent measurements of a scalar quantity x, which are a function of the parameter θ we are attempting to estimate. If the measurements follow a probability distribution $f(x;\theta)$ the CRLB is then:

$$\mathrm{var}(\hat{\theta}) \geq \frac{1}{I(\theta)} \tag{3.1}$$

Where $\hat{\theta}$ is the estimator of θ. $I(\theta)$ is the Fisher information of the n measurements

combined and is defined as

$$I(\theta) = nE_\theta \left[\left(\frac{\partial l(\boldsymbol{X};\theta)}{\partial \theta} \right)^2 \right] \tag{3.2}$$

where $E_\theta[.]$ is the expectation value with respect to $f(x;\theta)$ and $l(x;\theta)$ is the natural log of the likelihood function of a single measurement. It is useful to note that the Fisher information is additive. The Fisher information for any group of independent measurements in combination is simply the sum of the Fisher information for the individual measurements. This is true even if the measurements do not follow identical probability distributions.

For a multi-variate case where $\boldsymbol{\theta}$ is a vector of parameters, the CRLB is

$$\mathrm{cov}_\theta(\boldsymbol{T}(\boldsymbol{X})) \geq \frac{\partial \psi(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} [I(\theta)]^{-1} \left(\frac{\partial \psi(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right)^T \tag{3.3}$$

where $\boldsymbol{T}(\boldsymbol{X})$ is an estimator of any vector of functions based on the measurements $\boldsymbol{X}$. Its expectation value is denoted by $\psi(\boldsymbol{\theta})$. The Fisher information for multiple parameters takes a matrix form:

$$[I(\boldsymbol{\theta})]_{i,j} = E\left[\left(\frac{\partial}{\partial \theta_i} \log f(\boldsymbol{X};\boldsymbol{\theta}) \right) \left(\frac{\partial}{\partial \theta_j} \log f(\boldsymbol{X};\boldsymbol{\theta}) \right) | \boldsymbol{\theta} \right] \tag{3.4}$$

In the multi-variate case where the measurements follow a multi-variate normal distribution, $\boldsymbol{X} \approx N(\mu(\boldsymbol{\theta}), \Sigma(\boldsymbol{\theta}))$. The Fisher information matrix is

$$I_{m,n} = \frac{\partial \mu^T}{\partial \theta_m} \Sigma^{-1} \frac{\partial \mu}{\partial \theta_n} + \frac{1}{2} \mathrm{tr} \left(\Sigma^{-1} \frac{\partial \Sigma}{\partial \theta_m} \Sigma^{-1} \frac{\partial \Sigma}{\partial \theta_n} \right) \tag{3.5}$$

and in the case where Σ has no dependence on $\boldsymbol{\theta}$ the right-hand term disappears.

3.1.2 Fusion of Multiple Satellite Measurements

In this case study, the photogrammetric objective is the cumulative sum of position uncertainty over a hypothetical set of features to characterize, one at each collection sites. To quantify the uncertainty, I calculate a CRLB for each site generated from all

measurements events of that location. The specifics of when an imaging event takes place are discussed in section 3.3.3.

One of the benefits of using the CRLB in this way is that it is fairly simple to combine the utility of differing measurement modalities into a single quantity. This case study contains two such measurement modalities. For each collection site, all the measurement events ReCon satellites taken over the scenario duration are combined into a CRLB measurement of position uncertainty. Both modalities assume normally distributed covariance for a single measurement, simplifying the calculation and resulting in a normally distributed CRLB covariance for each site. Then, for each site CRLB estimate Principle Component Analysis (PCA) is used to find the 1-sigma characteristic extent of this covariance (i.e. the largest diameter of the covariance ellipsoid). Leveraging normality again, this is simply the largest eigenvalue of the covariance matrix. These characteristic extents of uncertainty are summed over all sites to arrive at the final scalar objective value. If there happens to be a site that has escaped measurement by any satellite, it is assumed to have a characteristic extent of uncertainty of 10 kilometers.

The process for a measurement event between a satellite and a collection site is summarized as:

1. For each modality of measurement, assume normally-distributed errors and generate a covariance in the Earth-Centered Inertial (ECI) frame. This covariance is based on the geometry between the satellite and collection site as well as assumed properties of the satellite imaging capabilities. The exact function depends on the measurement modality.

2. Convert the ECI covariance to a covariance in the Earth-Centered Earth-Fixed (ECEF) coordinate frame. The difference between ECI and ECEF and is just a simple rotation, so if $\boldsymbol{R}$ is the ECI-to-ECEF rotation matrix, then we have $\Sigma_{ECEF} = \boldsymbol{R}\Sigma_{ECI}\boldsymbol{R}^T$

3. Convert this ECEF covariance to a Fisher information matrix. In this case, the measurement quantities and the parameters we are estimating are both ECEF

position, so our Fisher information matrix is simply the inverse of Σ_{ECEF}. The second term of equation 3.5 is approximated as 0, as at the distances between a satellite and a collection site, Σ has almost zero derivative with respect to a change in position. If the satellite was significantly closer, there would be significant dependence.

4. Add this Fisher information matrix to the running sum of Fisher information for the collection site in question. An example of how the orientation and shape of the measurement uncertainty affect the final CRLB is shown in Figure 3-1.

In the follow two sections (3.1.3 and 3.1.4) I discuss in detail the two measurement modalities and how their ECI measurement errors are determined.

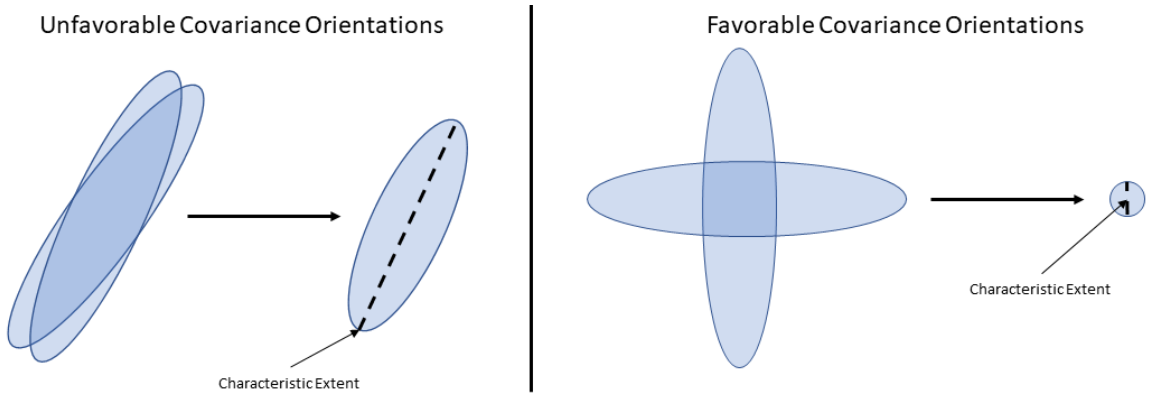

Figure 3-1: An example of fusing the covariance of two measurements to find the CRLB. [LEFT] The long dimensions of the covariances are almost aligned, the characteristic length of the combined covariance is only reduced by approximately $\sqrt{2}$. [RIGHT] The favorable orientation of these two covariance envelopes results in a much smaller uncertainty in their CRLB.

3.1.3 Measurement Modality 1: Direct Imagery

The first measurement type models using the 2D pixel location of a feature in satellite imagery to locate that feature in the 3D world. Generally, two or more such measurements are fused together to exploit stereoscopy. The camera parameters (which will be known by the satellite operator) can be used to convert a pixel location to the angular direction a feature lies from the a satellite, however, there is no direct

distance information from this measurement. We assume that the feature of interest is near the surface of the Earth. As such, the modeled uncertainty for this type of measurement is a long skinny ellipsoid. The angular resolution of the satellite imager is a system parameter provided by the user. This can be converted to a GSD specific to the distance from the satellite to the target site:

$$\text{GSD} = \Delta\Omega \cdot d \tag{3.6}$$

where $\Delta\Omega$ is the angular uncertainty parameter in radians and d is the distance between the satellite and the target site. The covariance in the short dimensions of the ellipsoid is set to a circle of the size of the GSD.

In the orthogonal, long, dimension, the ellipsoid is assumed to have a covariance of 10 kilometers. This number is arbitrary, but chosen to be much larger than the GSD used in my simulations. The assumption is the feature of interest is ground based, and should be within 10 km of the surface location the camera points at.

These guidelines are used to create a measurement covariance in the satellite frame of reference, with the x-axis pointing from the satellite towards the target site, and the other axes orthogonal to x. (The exact orientation of y and z being irrelevant, because the covariance in the y-z plane is a circle.) This covariance is then rotated to the ECI frame, which is sufficient for use in the measurement fusion process described in section 3.1.2. Figure 3-2 diagrams the measurement uncertainty of this type of direct measurement from 2D pixel locations.

3.1.4 Measurement Modality 2: Shadow Length Measurement

The second modality models measuring the length of a shadow to infer height information for a feature. This modality uses both the position of the satellite and the position of the sun relative to the target site.

The basis for this measurement is the simple trigonometric relationship between the length of a shadow, the elevation angle of the sun, and the height of an object above the surface of the Earth. For this calculation, a flat Earth is assumed, as the

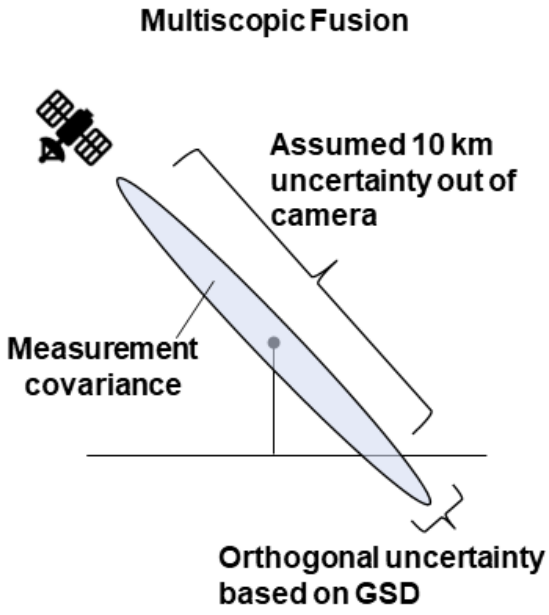

Figure 3-2: The covariance for a single direct measurement of a feature's location from satellite imagery is based on the GSD of the satellite optics. The uncertainty in depth is assumed to be 10 km. Orthogonal to the depth vector, the uncertainty is a direct function of GSD corrected for off-nadir pointing. In practice, two or more of these looks from diverse angular positions can be used to shrink the large covariance in the range dimension.

length of shadows of most man-made structures or other features that would be of interest for LEO satellite imagery are far smaller than the scale of the curvature of the Earth. It is assumed that the local elevation grading around the site is well enough to correct for those effects in the final calculated height. For a known length of shadow and solar elevation angle we have

$$h = l \cdot \tan^{-1}(\theta_{sun}) \tag{3.7}$$

where h is the calculated height of the feature, θ_{sun} is the elevation angle of the sun from the feature's location on Earth, and l is the measured horizontal length of the shadow.

An uncertainty in the length of the shadow as measured by the overhead satellite is approximated by the same GSD used in section 3.1.3. We can convert this into an

uncertainty in height by using the partial derivative of the equation 3.7.

$$\sigma_h = \text{GSD} \cdot \tan^{-1}(\theta_{sun}) \tag{3.8}$$

We then convert this scalar uncertainty in height to a 3D error ellipsoid in ECI space using the Jacobian of the transform from a length in height to a vector length in ECI.

$$\Sigma_{ECI} = \boldsymbol{J}\text{GSD}^2\boldsymbol{J}^T \tag{3.9}$$

The Jacobian $\boldsymbol{J}$ is simply the unit vector pointing in height direction (expressed in the ECI coordinate frame). Now the process can follow section 3.1.2. Figure 3-3 details the uncertainty from a a single shadow measurement.

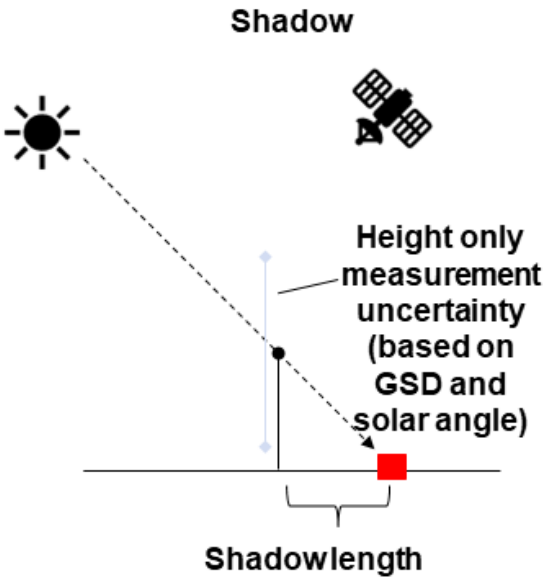

Figure 3-3: The uncertainty from a measurement of a shadow length for photogrammetric purposes. The uncertainty is based both on satellite GSD and the current solar angle. This provides information in only a single dimension.

The astute reader will wonder about singularities in the covariance and Fisher information transforms. A covariance of a 1D height measurement transformed into ECI covariance will indeed be singular. To avoid complications from this singularity, for this modality the ECI Fisher matrix is formed directly, rather than by inverting a covariance matrix. However, that matrix is still singular (after all, how could one get

3D uncertainty from information only along a 1D vector?), and it would seem this has just moved the problem from the domain of covariance to the domain of information. Recall though, this Fisher matrix will be summed with the Fisher matrices of other measurements, and the first modality described in section 3.1.3 produces a full 3D uncertainty. The combination of these 1D height measurements with at least one measurement from the first modality will not be singular, and the imaging logic discussed section 3.3.3 guarantees there will never be a case of a target site with only a shadow-based height measurement.

3.2 Variable Length Chromosome Formulation

The chromosomes of this scenario are collections of orbits specific satellites are directed to maneuver into at specific times. While it would be possible use the entire domain of valid Earth-centered orbits, I limit the space to circular orbits without any plane changes, like many of the works discussed in section 1.2. This is done as both a simplification to the development effort, and also a reflection of the very high energetic cost for plane changes. Exploration of more general orbits is left to future works. However, this is still more general than some previous works which limited options to RGT orbits. In this formulation, satellites are free to take non-RGT orbits and maneuvers may be scheduled at any time.

Recall from the lessons learned in section 2.6.4 that it is preferable to find a formulation of maneuver that is "stochastic" as possible. It is thus not sufficient to simply specify the altitude or semi-major axis of a target orbit, because maneuvering from one altitude to another will induce a phase shift in true anomaly. Thus, if an individual has a set of high-performing maneuvers, the addition of an earlier maneuver may result in a large degradation in performance. If for example, said early maneuver introduced a 180 degree phase shift in true anomaly, the satellite would be be on the opposite side of the Earth during the previously high-performing later portion of the simulation.

To avoid this problem, maneuver specification in the chromosome includes not only

Table 3.1: The Gene for a maneuver in this case study consists of these fields. The limits reflect the specific system parameters chosen in this study.

Field	**Description**	**Limits**	**Data Type**
Satellite Number	Identifies which satellite to maneuver	[0,22]	Int64
Phase	The true anomaly to target with a maneuver	[0,360]	Float64
Semi-major Axis	The semi-major axis of the circular target orbit	[250, 750] (in terms of altitude km)	Float64
Time	Starting time of the maneuver	[0,Scenario_Length] days	Float64

an altitude specification, but also a target true anomaly. With the specification of an absolute true anomaly to target, later maneuvers are more insulated from previous maneuvers.

Table 3.1 shows the parameters inside a Gene for this ReCon simulation. Included are a target semi-major axis, target phasing, a time of maneuver start, and a satellite identifier.

The ordering imposed on the genes is a simple sort first by time and then by satellite number. The equality operator compared all the fields of the gene. Equality is only declared if all fields matched. The mutation operator draws new values for all the fields from uniform random distributions. The distributions were limited to the values allowed by the constraints from the system parameters describing altitude regime and the number of satellites in the constellation. Phase was limited to the 0-360 degree range.

This scheme keeps any single maneuver feasible, but there are considerations regarding interactions between maneuvers which must be addressed. Certain combinations of maneuvers may be incompatible. For one, a maneuver should not start before the previous one has finished. In this case study, constraint violations like this are explicitly checked in the fitness simulation at the time of maneuver validity. With maneuver timing, a small amount of state tracks the progress of the currently

executing orbit transfer, and any subsequent maneuver which occurs before the platform idles is simply ignored. The gene becomes inactive. This approach can easily be extended to include some additional amount of turnaround time after physical maneuvering has ended. Additional checks are made on the remaining amount of delta-V in the platform, and energetically infeasible maneuvers are also cancelled. A maximum magnitude of burn can also be enforced in this manner (but wasn't here).

Not all potential constraints can be implemented by ignoring genes. This case study does not check for proximity between platforms and other satellites but that type of violation would need to be enforced with a penalty to the objective values. Any approach that guarantees feasible solutions end with superior rankings in the non-dominated sort is sufficient, such as modifying objective values by some large number. If the amount of violation can be quantified and stored as a new objective value, then even the infeasible solutions will be ranked according to how infeasible each is.

The details of simulating and implementing the maneuvers specified in a gene type are discussed in section 3.3.

3.3 Satellite Orbital Propagation and Maneuver Modeling

It is necessary to simulate the trajectory of each satellite in the constellation at every time-step in the scenario. The following subsections will discuss the details of orbit propagation, the exact form the maneuvers specified by genes take, and how the navigational burns were simulated.

3.3.1 Orbit Propagation

An explicit Runge-Kutta Fourth-Order (RK4) integration scheme was chosen to integrate the motion of satellites. As state, each satellite has a 3D position (x, y, z) and a 3D velocity $(v_x, v_y v_z)$. At each time-step, the local acceleration due to gravity

$(g_x, g_y g_z)$ is calculated such that the derivatives of state are

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ g_x \\ g_y \\ g_z \end{bmatrix} \tag{3.10}$$

Models of gravity are often presented as series of spherical harmonics fit to large amounts of experimental data. For this simulation, I use the data from EGM2008 [70]. For computational efficiency, I only use the first level of spherical harmonic terms. This is the tidal term that models the major oblateness of the Earth's shape, and is sufficient to account for most of the nodal precession used by ReCon satellites entering RGT orbits. The gravitational potential at a point was calculated as

$$V = \frac{GM}{r}\left(1 + \frac{a}{r}^2 C_{20} P_{20}\right) \tag{3.11}$$

where r is the distance of the point to the center of the Earth, a is the equatorial radius of the Earth, and

$$C_{20} = -0.484165143790815 \cdot 10^{-3}$$

$$P_{20} = \frac{1}{2}\left(3\cos(\theta)^2 - 1\right)$$

$$\theta = \frac{\pi}{2} - \tan^- 1(\phi)$$

where ϕ is geocentric co-latitude. Gravitational acceleration is then

$$g = \frac{V}{r} \tag{3.12}$$

The simulation was time-stepped at 5 seconds for the total time of the scenario. 5 seconds was chosen as a compromise between accuracy and computational perfor-

mance.

3.3.2 Double Hohmann-Transfer and Impulsive Burns

Maneuvering a satellite from its current orbit to a target semi-major axis and phasing is implemented using a double Hohmann transfer. A single Hohmann transfer is the a maneuver from one circular orbit to another. A first burn is executed to raise or lower the altitude of the orbit opposite the vehicle to the new value. Then 180 degrees later, where the vehicle's transfer orbit intersects with the new orbit, a second burn corrects orbital velocity such that the vehicle is now in the target orbit. The Hohmann transfer is very efficient, and requires only two burns, which are both parallel or anti-parallel to velocity.

Unfortunately, a single Hohmann transfer is not sufficient to control the phasing of the vehicle in the new orbit. To target a specific true anomaly in the target orbit, a more sophisticated program of burns is required. The double Hohmann transfer has been used for rendezvous between the space shuttle and orbiting platforms [48]. In the double Hohmann transfer, the first transfer places the vehicle in a phasing orbit. The vehicle idles here, and the phasing between the vehicle and the final target phasing is adjusted. Then the vehicle begins the second Hohmann transfer at a precise moment in time such that the phase change induced by this second transfer places the vehicle at the correct true anomaly upon termination. This is an attractive option for this case study, as the full phase authority provided satisfies the formulation of the VLCs in section 3.2.

As given in [48], the semi-major axis of the phasing orbit can be solved for from the following equation:

$$0 = \left[\left(\frac{\mu}{a_t^3}\right)^{1/2} - \left(\frac{\mu}{a_p^3}\right)^{1/2}\right] T - \phi - 2\pi - \frac{\pi}{\sqrt{8}}\left[\left(\frac{a_p + a_c}{a_p}\right)^{3/2} + \left(\frac{a_p + a_t}{a_p}\right)^{3/2}\right] \quad (3.13)$$

with μ as the Earth's gravitational constant, a_t the target semimajor axis, a_c the current semimajor axis, a_p the phasing orbit semimajor axis, ϕ the initial phase angle between the satellites current location and where a correctly phased satellite would

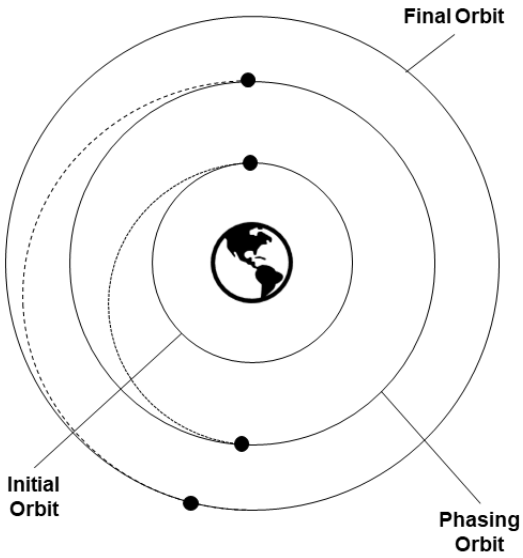

Figure 3-4: The double Hohmann transfer which is the maneuver form for the satellites in this ReCon case study. The first Hohmann transfer places the satellite in a phasing orbit. The second Hohmann transfer places the satellite in the final orbit, and in combination with the time spent in the phasing orbit, at the true anomaly specified by the Gene.

exist in the target orbit, and T the total duration of the maneuver. In this simulation, T is a system parameter provided by the user, and is set to 1 day for my optimization runs.

The equation must be solved iteratively for a_p, the semimajor axis of the phasing orbit. I solve this using an implementation of Newton's method. The gradient for Newton's method is calculated using the complex step method [60]. This method is similar to a traditional finite difference method, but instead takes a step in the imaginary direction. It provides superior truncation error and is simple to implement.

With the correct semimajor axis for the phasing orbit determined, the timing of the 4 burns can be calculated. The first burn occurs immediately, and the last burn occurs T time later. From those two set points, the second and third burns can be scheduled with knowledge of the periods of the 2 transfer orbits, which are calculated as

$$T_{transfer} = 2\pi\sqrt{\left(\frac{a_{initial} + a_{final}}{2}\right)^3 / \mu} \tag{3.14}$$

with $a_{initial}$ and a_{final} the beginning and ending semimajor axes for a given transfer. It is possible that a gene specifies the next maneuver before the current one has finished. When this occurs, the second maneuver is ignored. A maneuver will only be considered valid if at the validity time of the maneuver, the satellite in question is idle, and has completed all 4 burns from any previous maneuver.

The burns required for the transfer are modelled as impulsive burns. When the time for a burn falls within the next timestep, the vis-viva equation

$$v^2 = GM\left(\frac{2}{r} - \frac{1}{a}\right) \tag{3.15}$$

can solve for v, the required linear velocity of the satellite for an orbit of semimajor axis a. The difference between the required velocity and the current velocity is added to a counter recording the satellite's total expended delta-V, and the v_x, v_y, v_z values for the satellite are updated. The direction of the burn is either exactly parallel or anti-parallel to the current velocity.

3.3.3 On-Board Imaging Logic

To model limited space and energy for capturing images of collection sites onboard the satellites, measurements are not taken at every possible time the site is in view. Instead, measurement is only commanded at the point of closest approach (POCA) for a satellite-site pair. POCA is declared only when the relative velocity of a satellite with respect to a given target site will transition from negative to positive in the next time step. In addition there are requirements on elevation and solar illumination. The full criteria is:

- A satellite will only image a target site when POCA will be reached within the next timestep.

- The satellite must be above a minimum elevation from the point of view of the target site. This is given as a system parameter, and I have set it to 10 degrees in my simulations.

- The target site must be sunlit. Similar to the minimum elevation required for the satellite, there is also a minimum solar elevation angle for the site. This is also a system parameter which I have set to 10 degrees for my simulations.

- Each target site has a time period during the simulation which it is available to be imaged. This is meant to stand in for the temporal requirements potential ReCon tasking may impose. The target site cannot be measured if it is outside of this validity period.

If those criteria are met, measurements are made following the process in section 3.1.

3.4 N2 Diagram

The N2 diagram for the fitness function simulation is shown in figure 3-5.

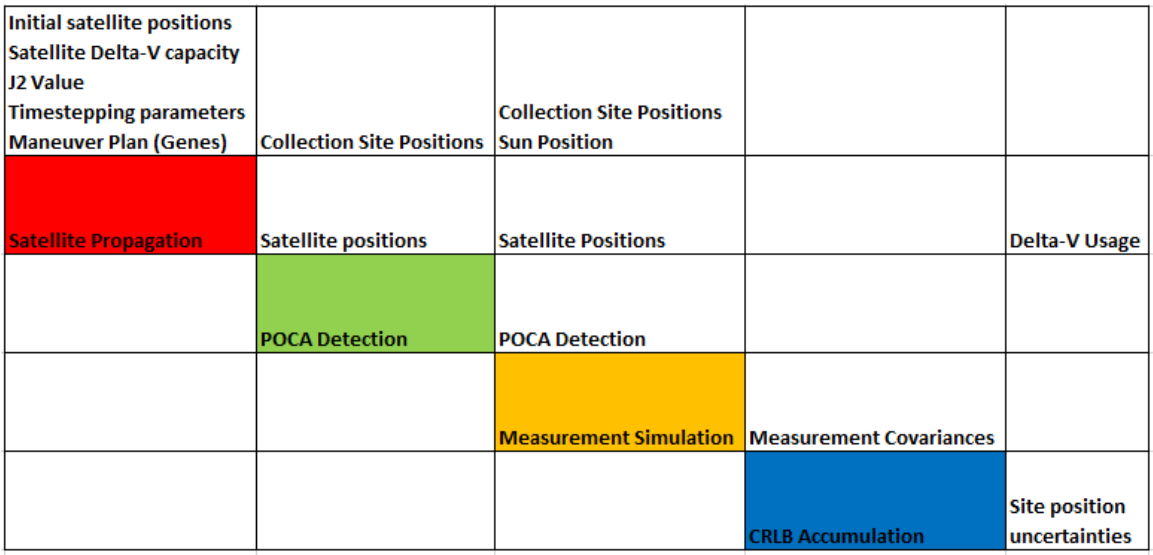

Figure 3-5: The N2 diagram outlining the simulation used within the fitness function.

The simulation is designed in an entirely feed-forward manner, avoiding feedback. As the satellites propagate and maneuver in an entirely open-loop manner, the trajectories of satellites can be entirely computed before running interactions with the ground sites.

3.5 Results

I analyzed two sub-scenarios in this case study. The first is a global scenario with collection sites scattered across the entire globe with the campaign constrained to a short time. The second is a regional case where all the collection sites lie in a geographically constrained area, but the duration of the campaign in longer. The validity period for sites was varied over several different time lengths. This thesis focuses on optimizing maneuvers, and not the design of a constellation or its constituent satellites. As such, a ReCon was chosen from previous study. The constellation evaluated in these runs was taken from Legge's work [58]. Legge co-optimized the design of satellite platforms and orbits for various classes of constellations considering cost and an imaging performance metric. I chose to use a non-dominated solution he identified in his section on symmetric constellations. The chosen solution comes from the "knee-in-the-curve" of the approximated Pareto front (Figure 3-6). This is the point where the returns from investing more money begin to significantly diminish.

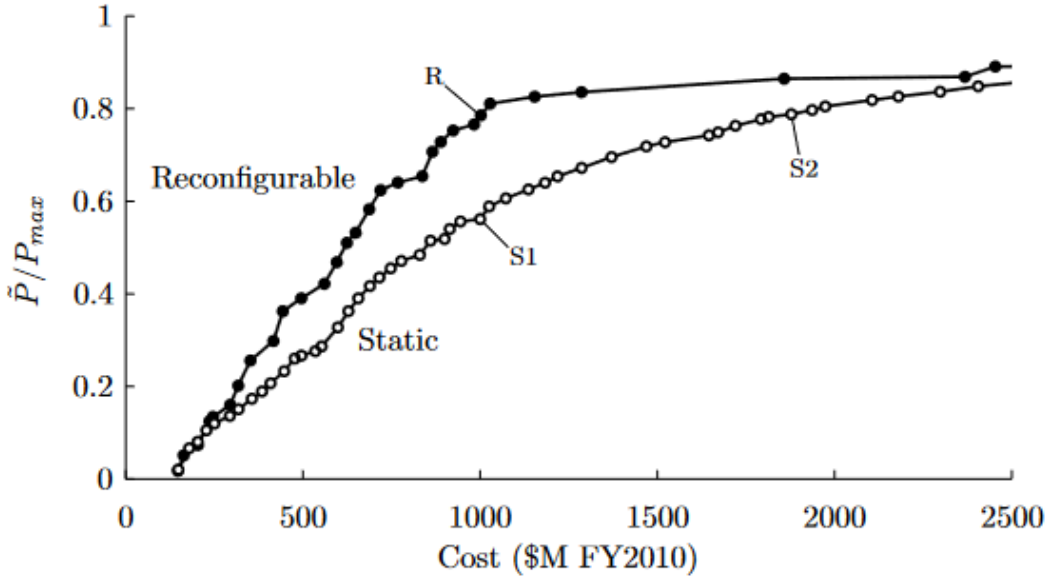

Figure 3-6: The performance of reconfigurable symmetric constellations versus static symmetric constellations. The non-dominated solution marked with the R was used as the constellation under test for this case study. Adopted from [58].

The hardware and orbit configuration of the selected constellation is shown in

Table 3.2: The details of the test constellation chosen from Legge. This is a symmetric Walker constellation. Each plane contains a single satellite.

Parameter	Value
Number of Orbital Planes	23
Inclination	57.73 deg
Altitude (at start)	484.9 km
Available ΔV on Satellite	917 m/s

table 3.2.

3.5.1 Global Observation Scenario

I generated a collection of randomly chosen collection sites on the Earth for the global scenario. The sites are limited to being on the land areas of the Earth, between -60 and +60 degrees of latitude. The latitude limits were chosen to match the inclination of the constellation under test. Latitudes beyond 60 would not observable by the constellation. Figure 3-7 shows the location of chosen sites.

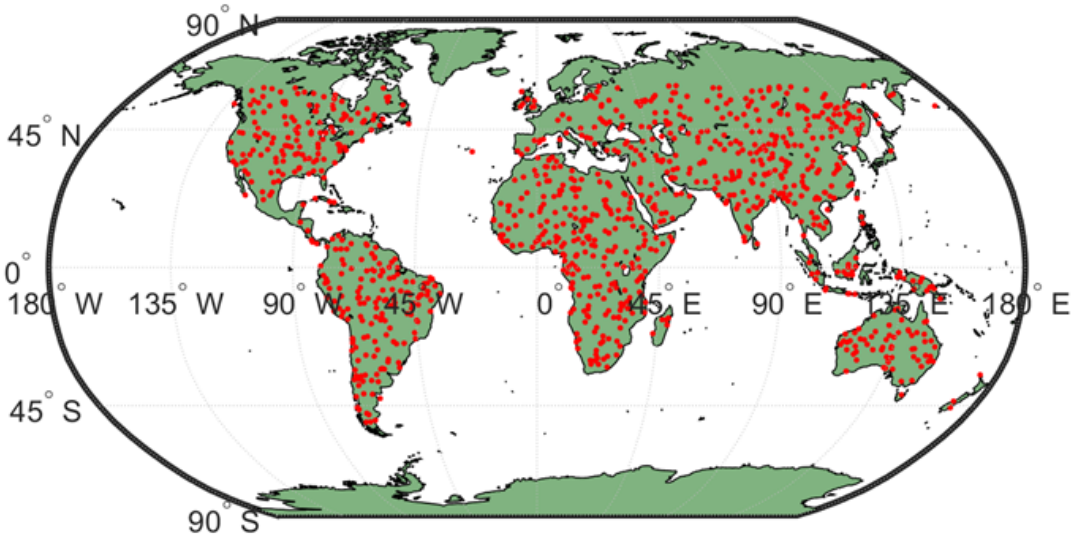

Figure 3-7: The geographic locations of target sites for the global ReCon scenario. The sites are uniformly distributed across the land area of Earth. 1000 sites were drawn.

Each target site was given a randomly assigned location on the Earth, as well as

Table 3.3: The system parameters used in the global scenario.

Parameter	**Value**
Population	2048
Chance of Gene Mutation	0.03
Chance of Gene Copy	0.01
Chance of Gene Delete	0.01
Rel. Tolerance for Termination Criteria	0.01
Termination Max Number of Unchanged Generations	40
Solar Elevation Limit	10 deg
Satellite Elevation Limit	10 deg
Satellite Angular Resolution	12 μrad
Scenario Duration	3 Days
Number of Collection Sites	100, 1000
Validity Time of Sites	6 Hr, 12 Hr, 1 Day, 3 Day, 6 Day

a validity period. Inside a single run, the validity periods are of uniform length, but for each site, the center of the period was drawn uniformly across the whole scenario. Validity periods were allowed to begin before the beginning of the simulation or after the end.

The surface location was drawn using the Metropolis-Hastings algorithm [46], which provided a simple way to limit the ground locations to actual land areas on the Earth. In addition, I added an additional third dimension beyond latitude and longitude to the Metropolis-Hastings algorithm. This dimension corrected for the effects of the Mercator projection; higher latitude sites were less likely to account for the latitude dependence on the length of a degree of longitude.

In total, a 1000 sites were drawn across the globe. Not all optimization runs used the totality of sites.

A series of optimization runs were undertaken with different validity time lengths for the target sites. The system parameters for the global scenario are shown in table 3.3. This series was run with the first 100 collection sites and then the full 1000. The Pareto fronts for the 100 site cases are shown in figure 3-8.

As one would expect, the scenarios with longer site validity times had much lower

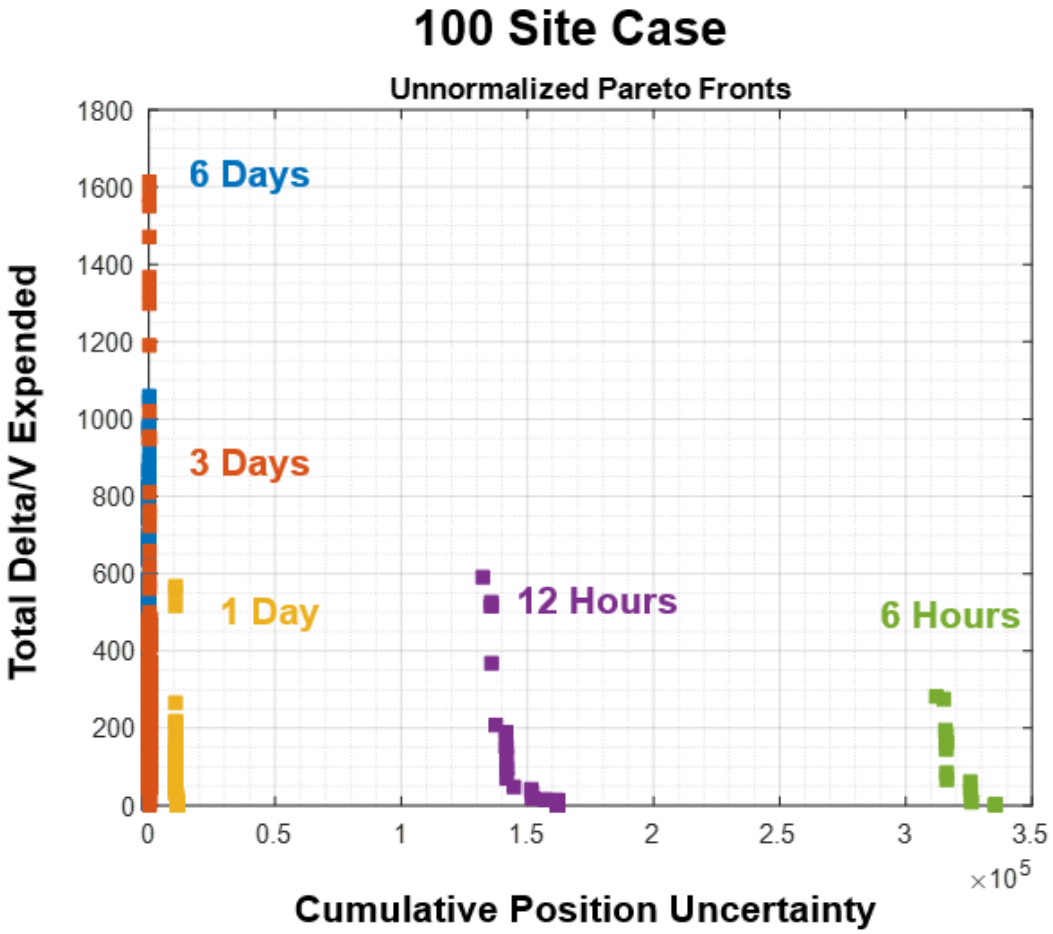

Figure 3-8: The Pareto front traced out by the optimization framework for 100 global sites. The different traces correspond to runs with different lifetimes for the target sites.

absolute position uncertainty than the those with shorter. There appears to be a large uptick in the position uncertainty as the validity length falls under a day. Possibly, this is an interaction with the requirements for the sites to be sunlit. When the validity period is less than one day, it is possible for a site to exist in darkness for the majority of time, or even all of the entire validity interval. Another trend is that the runs with longer validity times found solutions with more total delta-V expanded. One might expect that shorter validity times might result in more delta-V spent, as satellites are forced to take fuel-intensive routes to collect on different collection sites before they disappear. Instead, it seems that at some point, the schedule becomes

so challenging that the physics of the allowable orbits simply do not permit zipping between sites. To make a travel analogy, if you have in-person meetings in New York and California one hour apart no flight can even get you to both so you choose one meeting. If alternatively, you only needed to visit both cities in an 6 hour period, you certainly could spend all your funding on chartering a private plane.

To test if these trends would hold up with more sites (under the idea that if enough sites existed, even if short-lived, a vehicle might still gain utility taking an expensive maneuver) this same suite of validity times was run with all 1000 sites. Results can be shown in figure 3-9.

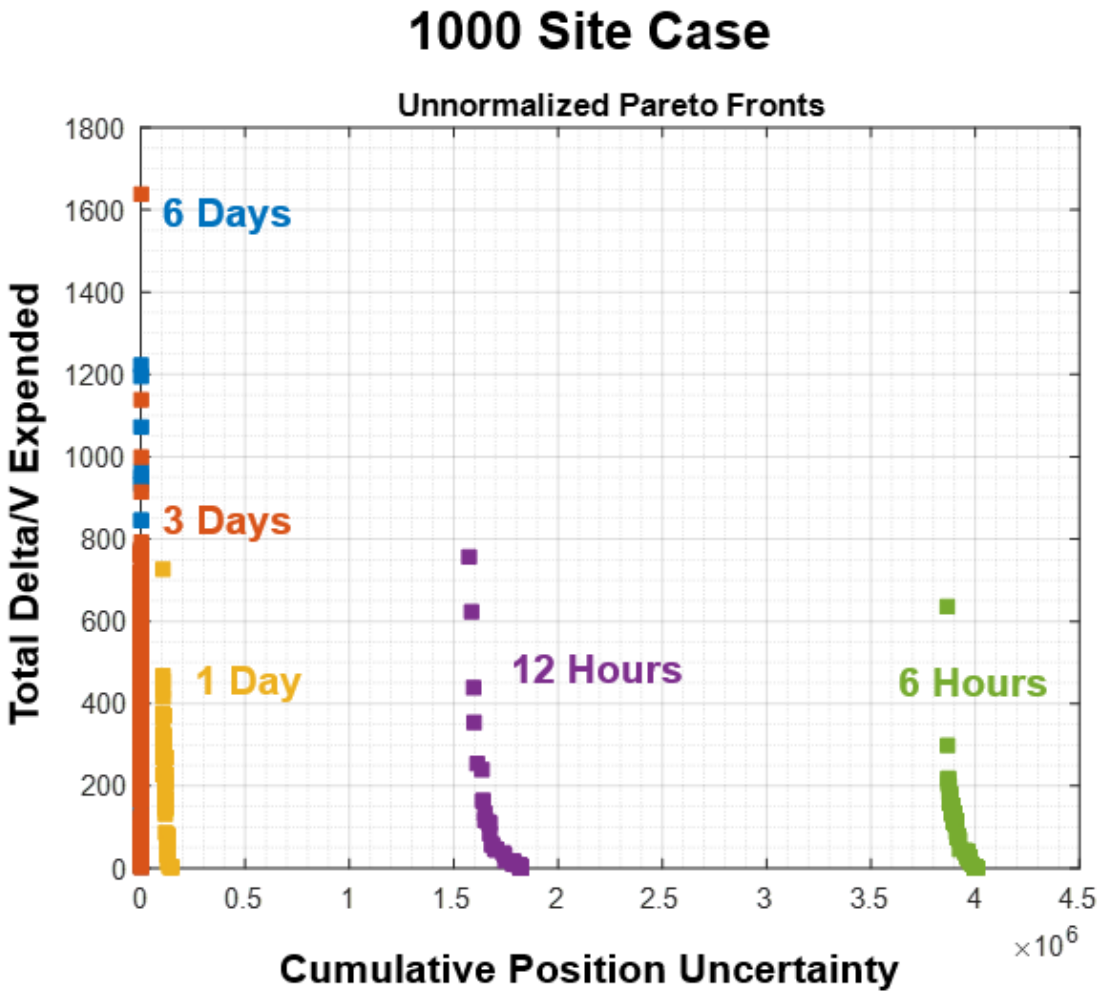

Figure 3-9: The Pareto front traced out by the optimization framework for 1000 global sites. The different traces correspond to runs with different lifetimes for the target sites.

It is clear from figure 3-9, that the discussed trends hold up in both cases. The position uncertainty varies by about one order of magnitude, which would be explained by the increase in sites from 100 to 1000. I posit that for an expensive maneuver to be profitable in utility, a vehicle can't sacrifice significant numbers of target sites at either the origin orbit or the destination orbit. Simply, either the vehicle zips away and misses significant utility from its original orbit, or the vehicle stays in its original orbit and misses utility from the what would have been the new orbit. The only difference being, the second option saves delta-V. For scenarios with long-lived collection sites, there is time to collect on both ends, but this is not true for the short-lived sites.

It is useful to also compare these runs in terms of relative performance increase (compared to the static case) as opposed to just absolute objective numbers. Figures 3-10, and 3-11 show the Pareto fronts normalized to the photogrammetric performance of the static (non-maneuvering) case. The static case was added in separately, as following the discussion in section 2.6.4, it was beneficial to penalize the zero-maneuver case to stop it from dominating the early population.

In the normalized Pareto fronts, a direct trade-off between delta-V and performance enhancement can be seen. These Pareto curves can be thought of illustrating the value of reconfigurability versus the added delta-V cost. In general, it seems that the scenarios with longer site validity times have more opportunity for maneuvers to improve performance, reaching approximately a 25 percent decrease in position uncertainty. There appears to be more variance in the performance between 3 days and 6 hours validity time with respect to the number of sites. Indeed, the 1 day validity time scenario had a large increase in relative performance with the 1000 target site case. It is difficult to say with certainty what is happening there, but the validity time is matched to the maneuver time in the 1 day case, and perhaps this makes such a scenario particularly sensitive to the draw of sites.

In addition to the overall performance, it is possible to analyze how the maneuvering plans gain utility over the static case. It is insightful to take the highest maneuvering case and case and compare the photogrammetric performance across

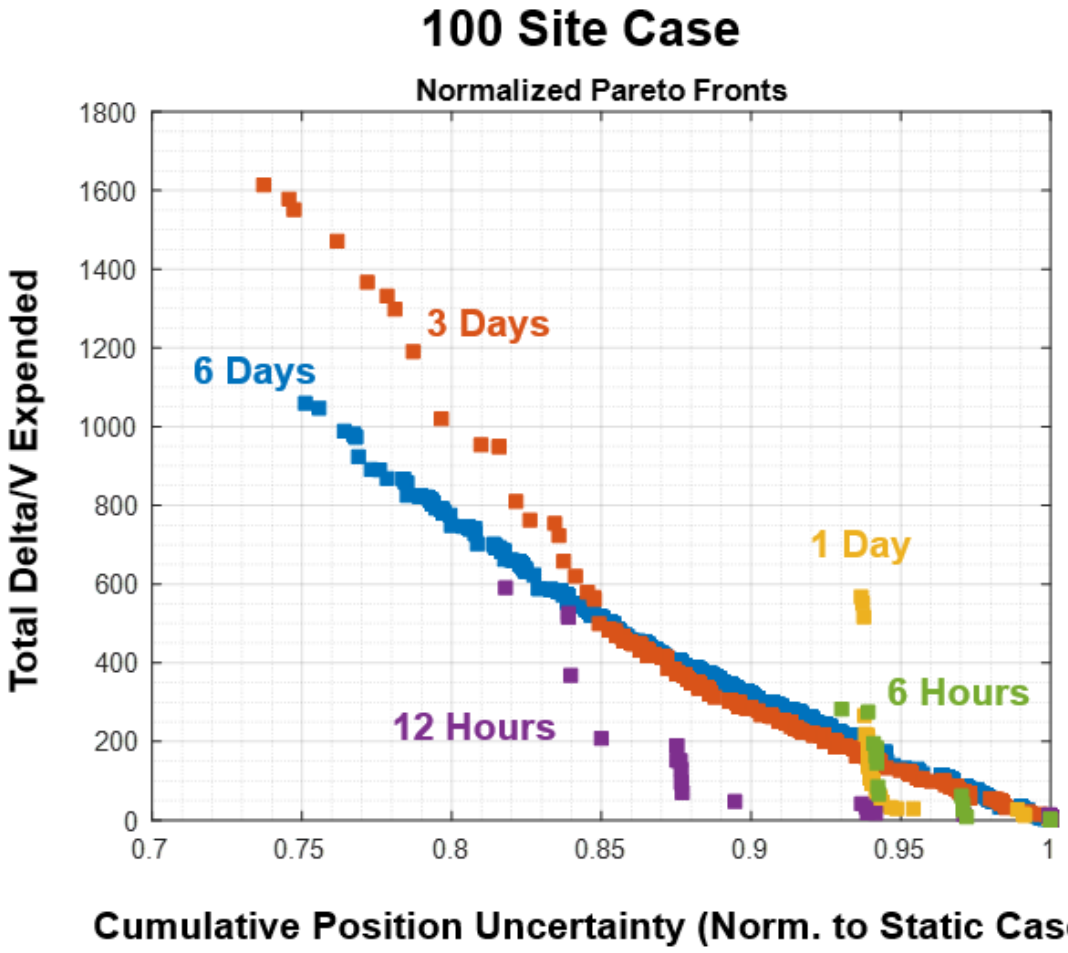

Figure 3-10: The Pareto front traced out by the optimization framework for 100 global sites, but relative to the performance in the static case.

the target sites. Figures 3-12 and 3-13 show such a comparison for the 6 day validity time run and the 6 hour validity time run, both on the 1000 site case.

One might first posit that for long-lived target sites, there is less utility to be gained from maneuvering, as even a non-maneuvering satellite will cover a large swatch of the Earth's surface over the entire scenario time. With long-lived sites, satellites get a large amount of measurement opportunities simply by existing for long period of time, i.e. a vehicle only has to be in the right place eventually, and not at the right time. That intuition doesn't pan out though. It is clear from figure 3-12 and 3-13, that it is the long-lived collection site case where maneuvering brings significant gains. Here, the vast majority of sites improved in utility with maneuvering, and by a larger magnitude than other sites degraded. On the other hand, the short-lived case had

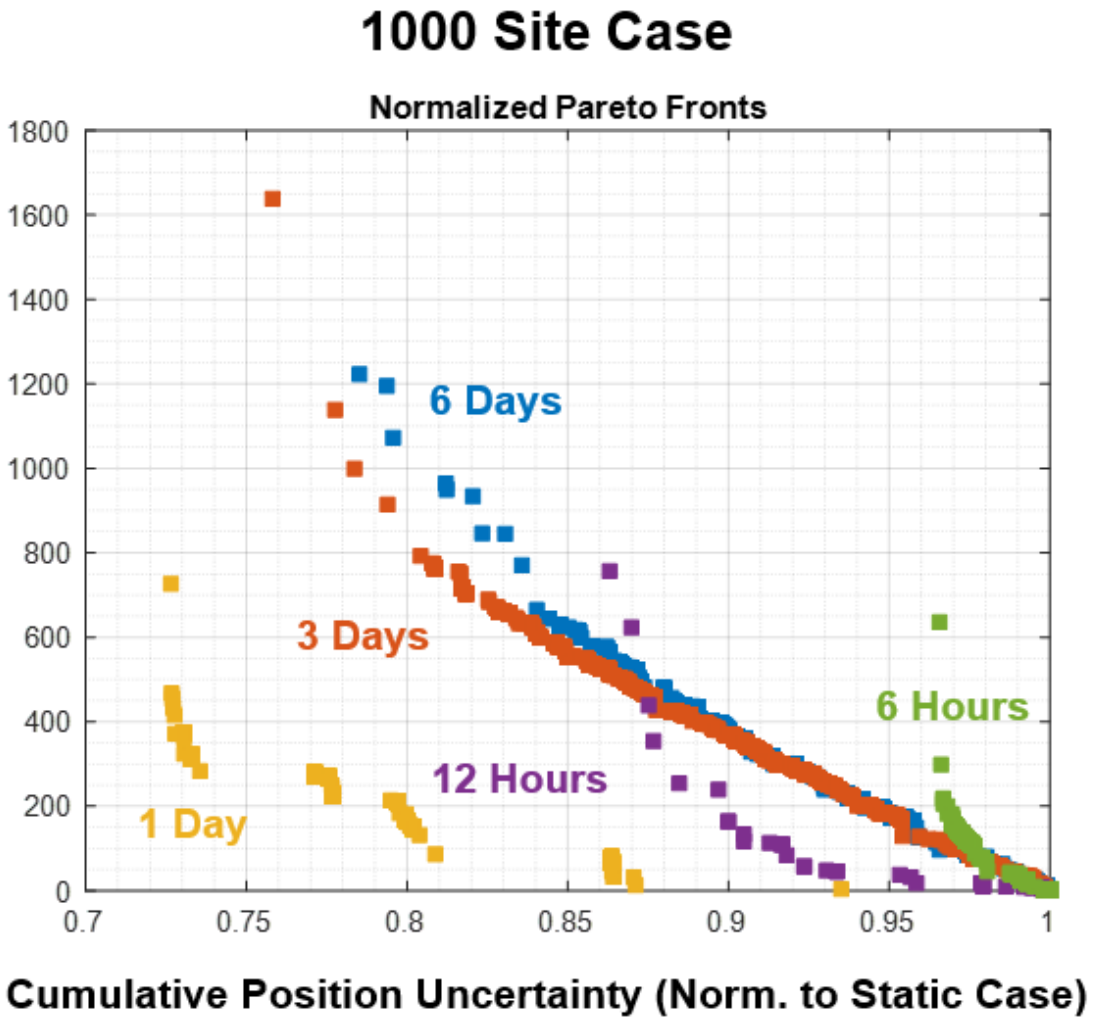

Figure 3-11: The Pareto front traced out by the optimization framework for 1000 global sites but normalized to the static case

a much more even equal split between sites improved and sites improved, and such improvement was only a slight bit higher than the degradation.

The conclusion then is that large number of sites with long periods of temporal overlap have the most potential benefit from reconfiguration. Despite being valid for relatively large amounts of time, for at least this photogrammetric metric, a static constellation is quite un-optimal, and there exist maneuvers which can improve performance broadly among sites. On the other hand, for short-lived target sites, it just not seems possible for a maneuver to increase performance in one site without degrading others. Even if a maneuver causes a net gain in utility, it is quite small in

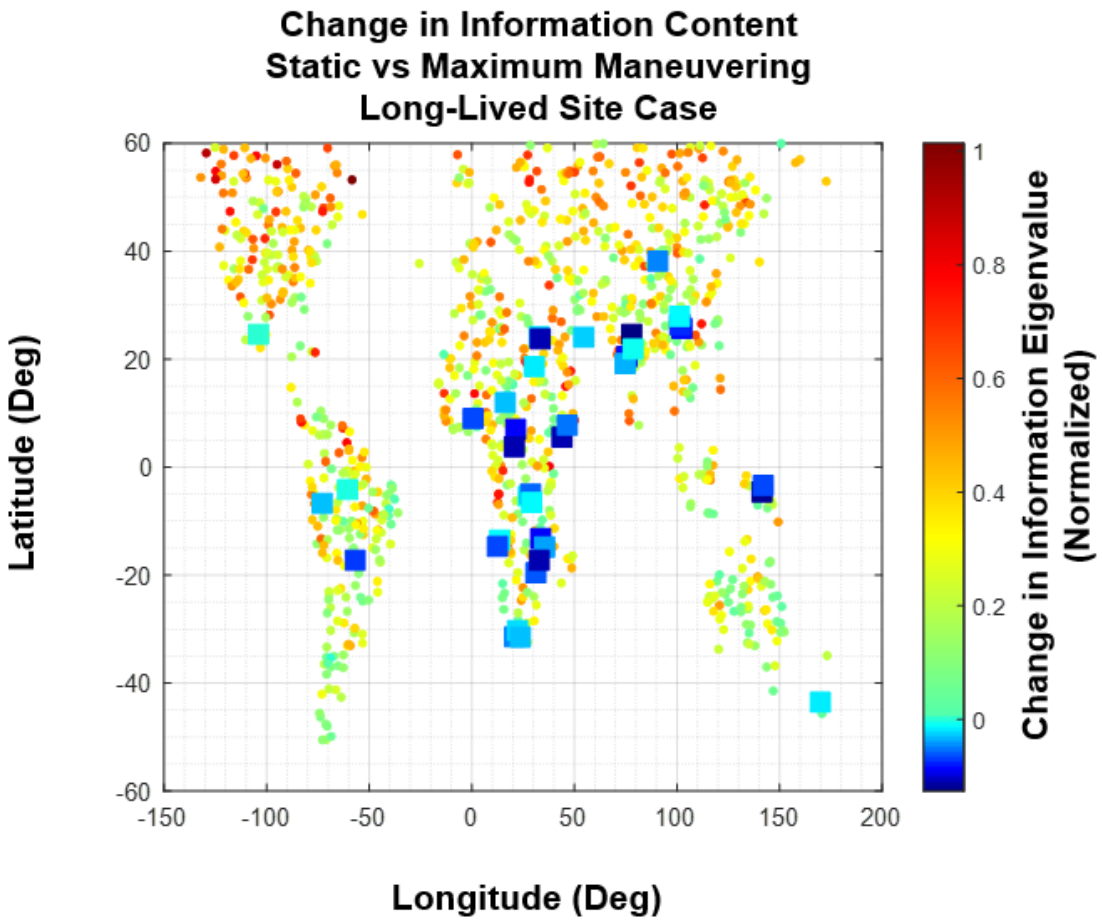

Figure 3-12: This map plots how much additional information content (in terms of Fisher information) was added to the component of greatest uncertainty at each site in the max maneuvering case compared to the static case. Dots show sites which are improvements and squares show sites which worsened. This was for the case with 3 day lifetimes for the target sites. Most sites improved, and by magnitudes significantly greater than the other sites degraded.

comparison.

3.5.2 Regional Observation Scenario: Ukraine

The previous section evaluated maneuver optimization on a global scale, distributing the target sites over the entire world (within the inclination range of the constellation orbits). However, a real life application may necessitate a period of elevated interest in one regional part of the world, small enough to be clearly different from a global

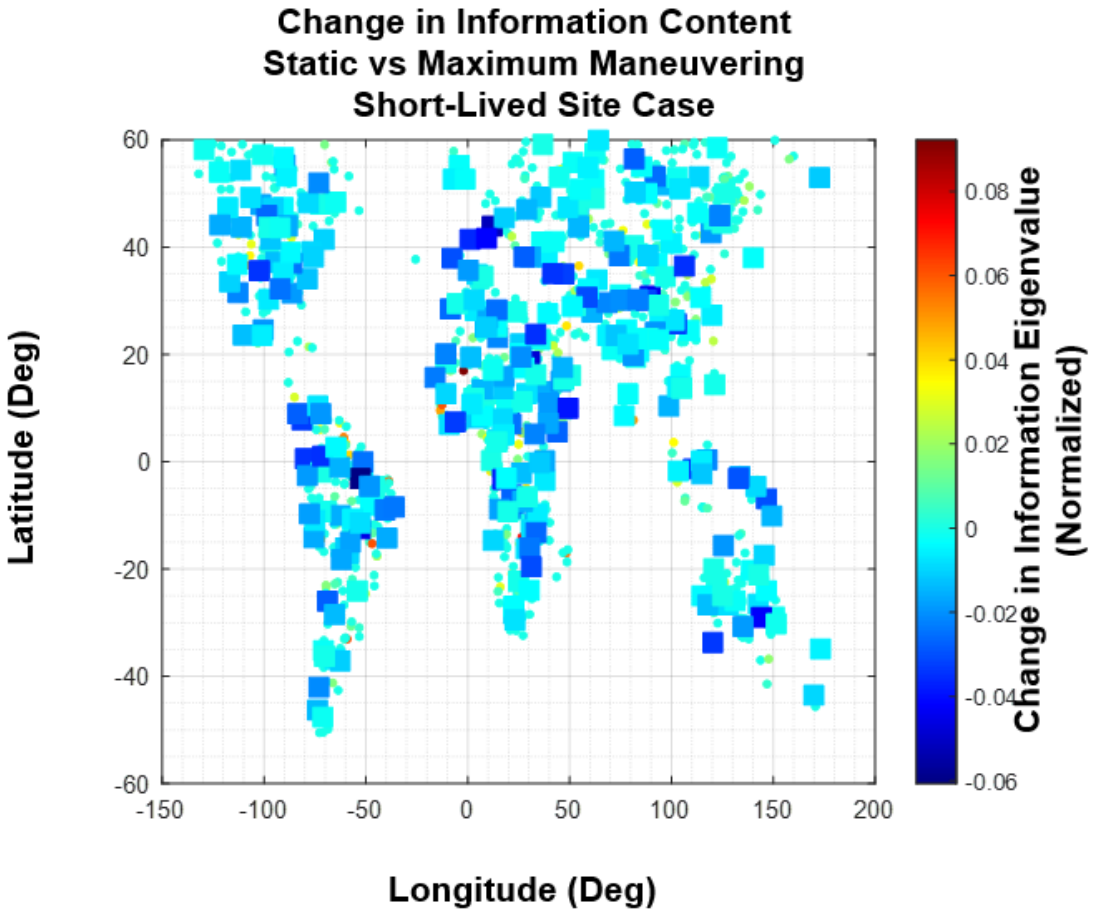

Figure 3-13: This map plots how much additional information content (in terms of Fisher information) was added to the component of greatest uncertainty at each site in the max maneuvering case compared to the static case. Dots show sites which are improvements and squares show sites which worsened. This is for the case with target site life times of 6 hours. Unlike the 3 day case, the proportion of improved sites to degraded sites is much lower (almost equal).

scope, but large enough that a small amount of traditional ReCon RGT orbits are not sufficient for coverage. Thus, I ran a regional simulation to evaluate the performance of the optimization framework for this set of circumstances.

Scenario: Ukraine

Real-life geopolitics currently offers an actual example of such a circumstance. It is apparent that the current conflict in Ukraine has brought heightened demand for

satellite products from both government and civilian entities. Open reporting highlights the use of satellite imagery for tracking military elements [54], the collection of evidence regarding war crimes [29], and monitoring humanitarian issues [76]. Recognizing the key influence Earth imaging satellites are imposing on the current conflict, I formulated an optimization run with collection sites limited to Ukraine.

Figure 3-14: The region of interest considered for the regional observation scenario. This region includes the legally recognized borders of Ukraine and areas of the Black Sea, an area which should include the majority of sites of interest with respect to the current conflict.

Figure 3-14 shows the region considered in this scenario. It is digitally represented as a set of latitude/longitude vertices which describe a closed polygon where target sites are placed. This region of interest includes the current legally recognized borders of Ukraine and parts of the Black Sea. A 1000 sites were generated for this scenario, in the same Metropolis-Hastings method as the global scenario (obviously with the addition that the sites fall within the region of interest). The final draw of collection

sites is shown in Figure 3-15

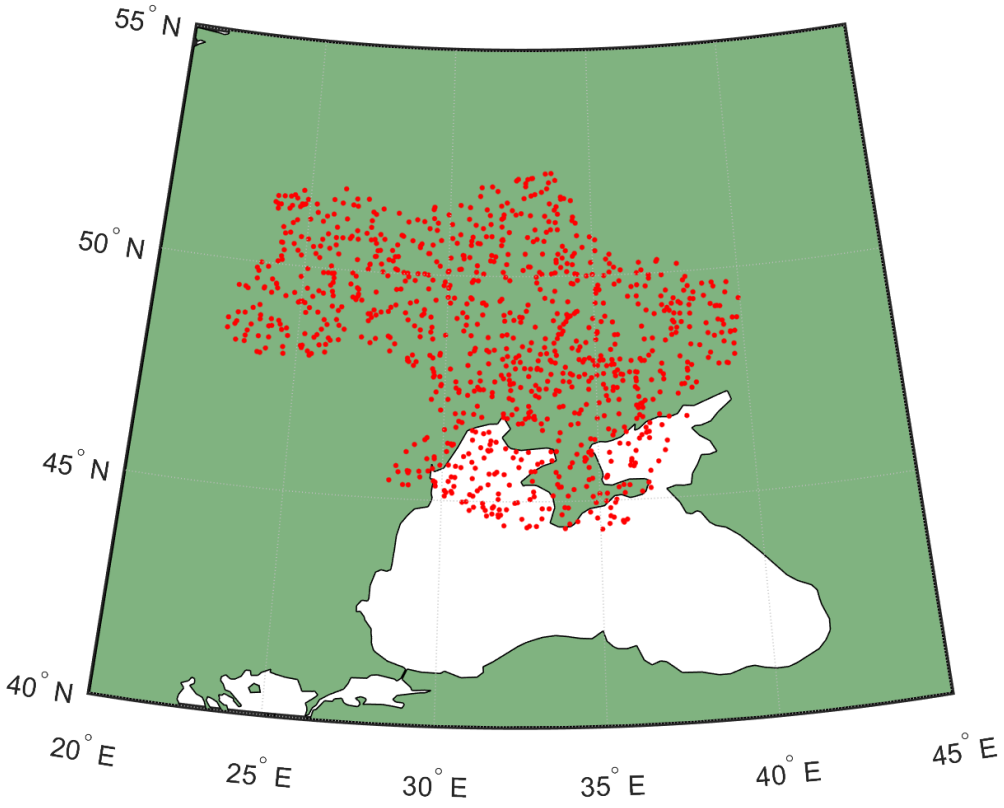

Figure 3-15: The collection sites as drawn for the regional collection scenario. 1000 sites were drawn in total.

Temporally, the total simulated time of the collection campaign was extended from 3 days to 14 days. This increase in scenario time length is meant to be more representative of the planning horizon a regional campaign might use. The validity times for the sites were varied along the same parameter sampling as the global case, and the optimizer ran separately for these cases. Also like the global case, runs were made for both the full 1000 site campaign as well as a 100 site subset. This regional scenario uses the same constellation as modelled in the global scenario (the Pareto optimal design from Legge). The system parameters for the regional scenario are summarized in table 3.4.

The raw (not normalized to the performance of the static case) Pareto fronts are shown in figures 3-16 and 3-17. The same overall trends that were observed in

Table 3.4: The system parameters used in the regional scenario.

Parameter	**Value**
Population	2048
Chance of Gene Mutation	0.03
Chance of Gene Copy	0.01
Chance of Gene Delete	0.01
Rel. Tolerance for Termination Criteria	0.01
Termination Max Number of Unchanged Generations	40
Solar Elevation Limit	10 deg
Satellite Elevation Limit	10 deg
Satellite Angular Resolution	12 μrad
Scenario Duration	14 Days
Number of Collection Sites	100, 1000
Validity Time of Sites	6 Hr, 12 Hr, 1 Day, 3 Day, 6 Day

the global case are seen here as well. The relative performance between the runs is still heavily dependent on the lifetime of the collection sites, with a large jump in performance occurring when jumping from a 12 hour site lifetime to a 24 hour site lifetime. In addition, the longer the site lifetime, the higher the delta-V ceiling of the Pareto curve; that is the marginal utility of expending delta-V diminished more slowly in the long lifetime cases than the short.

The Pareto fronts normalized to the uncertainty performance of the static case are shown in figures 3-18 and 3-19. Overall, we can see the same trends as in the global case. The runs with longer validity times still generally can achieve a greater relative reduction in cumulative position uncertainty than those runs with shorter site lifetimes. As in the global case, there was more volatility between the 1000 site run and the 100 site run for the shorter validity times. In the global case there was a large change in the performance curve of the 1 day run, while here there is a large change in the 12 hour run. Of note, in this regional scenario some of the runs achieved a 35 percent decrease in position uncertainty compared to the static case. This is better than the approximately 25 percent improvement in the global case. Intuitively, one might expect better performance from this case, as tighter geographical constraints

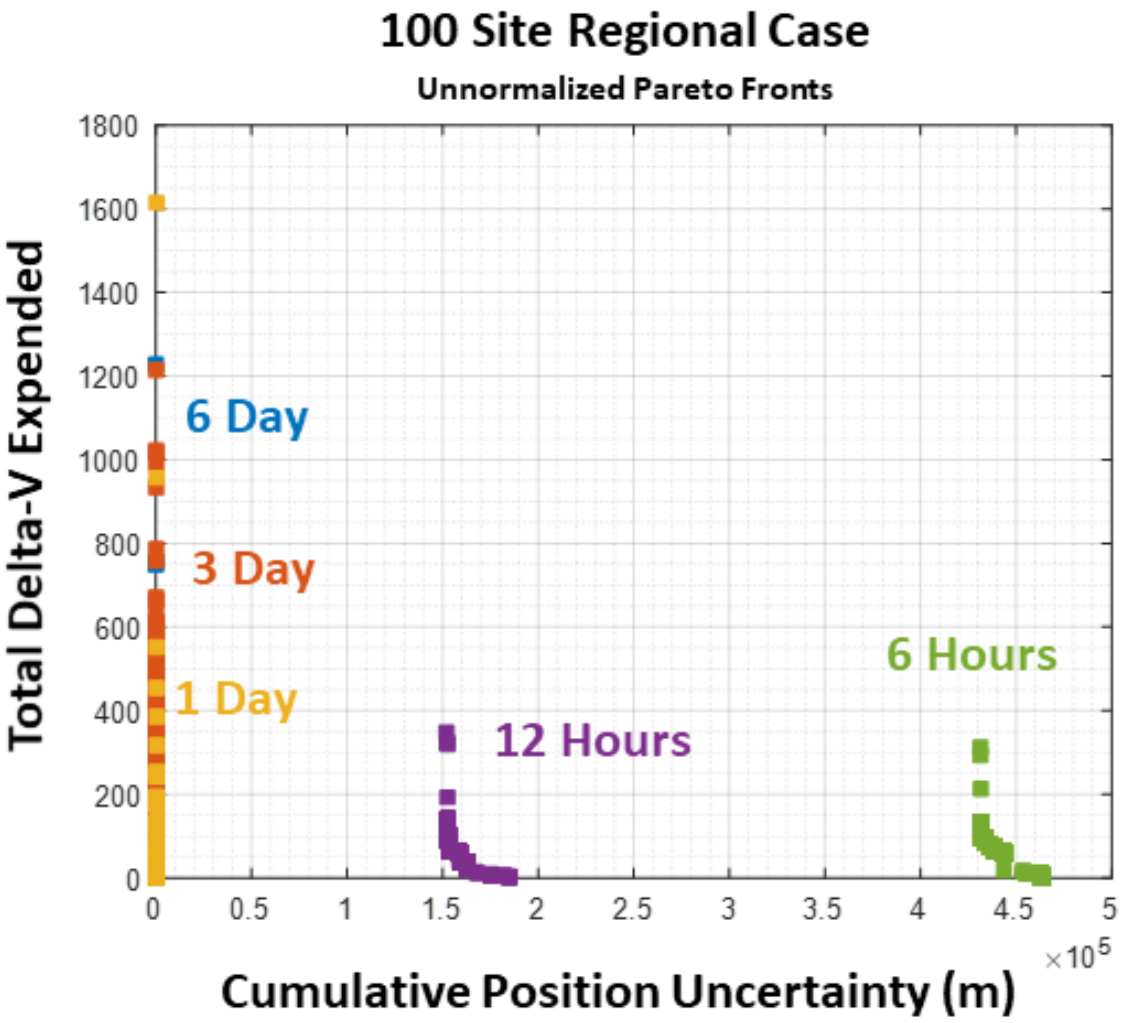

Figure 3-16: Pareto fronts for the maneuver plans generated for the regional scenario. These are the runs for the 100 site case at different collection site lifetimes.

should give more "bang for the delta-V" when evaluating a geometry based objective function; much easier to optimize for coverage over a constrained area rather than most of the globe, but it is unclear if that conclusion would generalize to all scenarios. As the area of interest becomes smaller and smaller, most of a satellite's trajectory becomes irrelevant. There could be a situation where it is more fuel efficient to make small gains across the entirety of a satellite's trajectory, rather than making a large gain in one critical area.

Maps which illustrate the improvement in Fisher information for each site are presented in figures 3-20 and 3-21. The trends once again follow the global case.

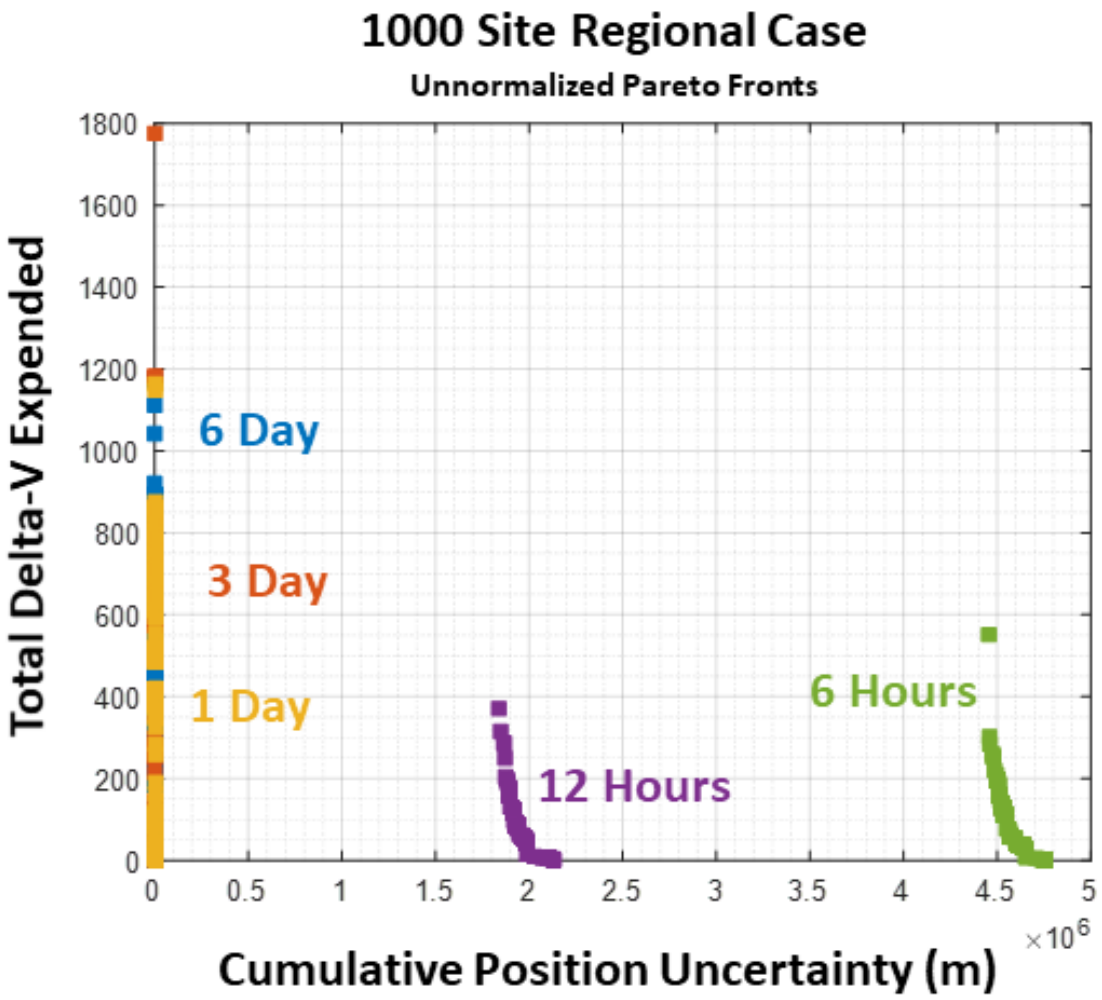

Figure 3-17: Pareto fronts for the maneuver plans generated for the regional scenario. These are the runs for the 1000 site case at different collection site lifetimes.

These maps again compare the difference in performance between the static case and the most delta-V intensive case for two 1000 case runs: the longest collection site lifetime and the shortest. Here, the trend in the longest lifetime can be seen as the same trend for the global case taken to the extreme. Recall in the global case, the vast majority, but not all, sites improved, and by a larger magnitude than those sites which degraded. In this regional case, every single site improved. There are variations geographically in the improvement. A corridor from the northwest to the southeast of the country seems to benefit the most from reconfiguration. The shortest collection lifetime case follows the trend from the global scenario as well. Here, we

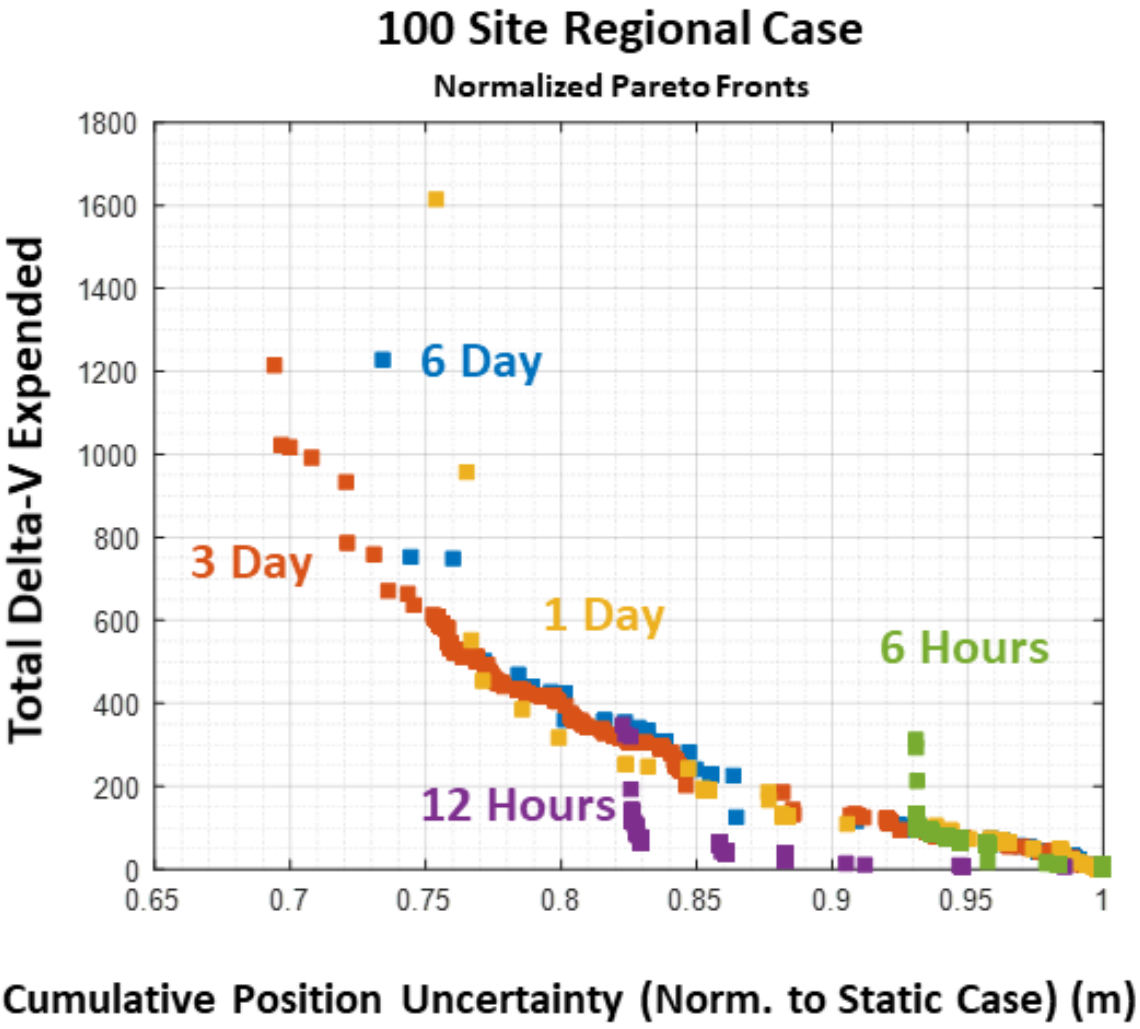

Figure 3-18: Pareto fronts for the maneuver plans generated for the regional scenario. These are the runs for the 100 site case at different collection site lifetimes. Uncertainty performance is normalized to the static case.

still see a much more mixed bag of change in uncertainty. A much more even amount of sites improved versus worsened. Several key sites drove most of the improvement. A difference from the global case are the magnitudes of the relative changes. In the global scenario, the short site lifetime case had relative changes on the order of percents, but here they are on the orders of tens of percents.

The fact of the regional case displaying the same overall trends as the global case is not immediately apparent from intuition, so the similarity has some significance here. In this regional case, a vast proportion of the trajectory of a satellite is away from any

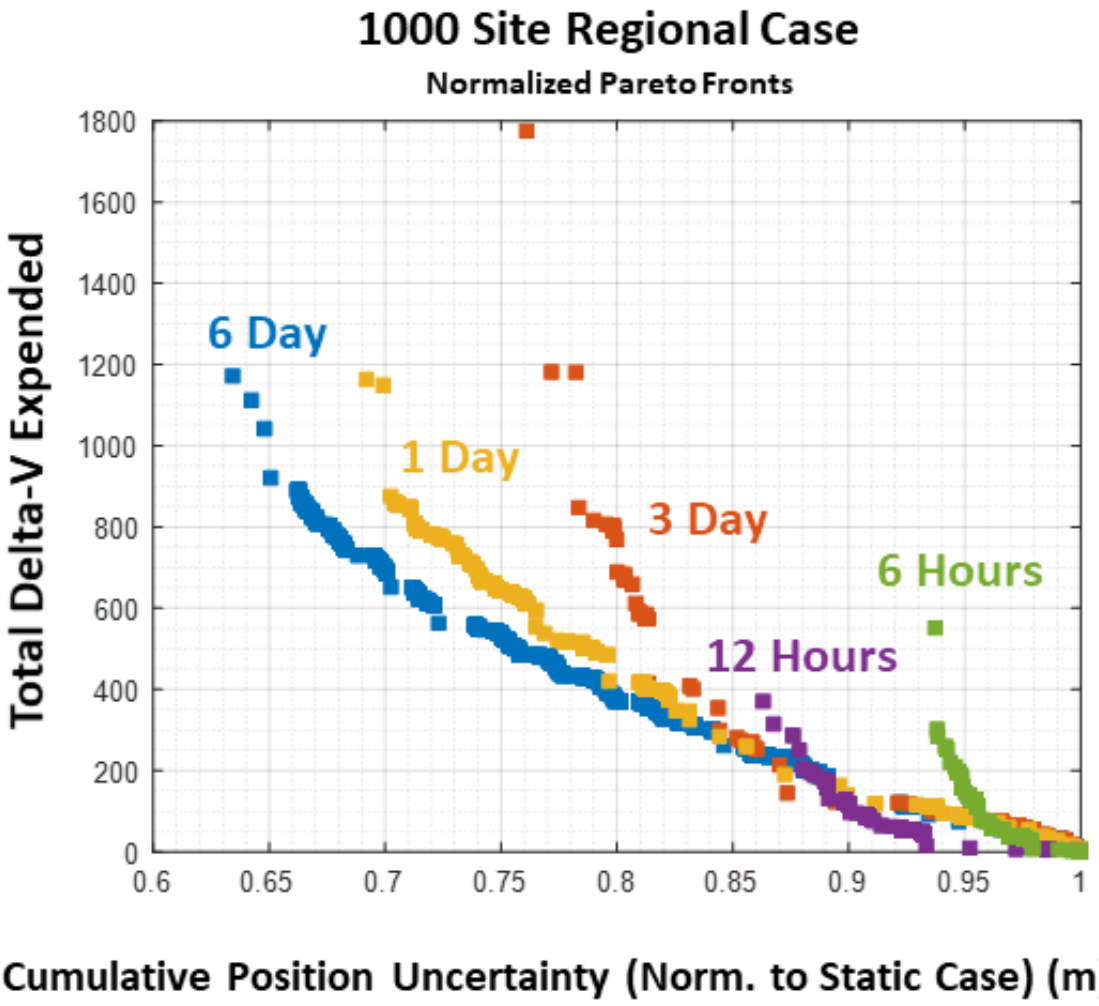

Figure 3-19: Pareto fronts for the maneuver plans generated for the regional scenario. These are the runs for the 1000 site case at different collection site lifetimes. Uncertainty performance is normalized to the static case.

site, whereas in the global case a satellite may image multiple sites across continents in a single orbit. It could have been the case that spreading the imaging opportunities across a broader proportion of the satellite trajectory versus concentrating the imaging along a smaller portion might lead to different behaviors. The comparison between the global scenario and the regional scenario here seem to show that is not the

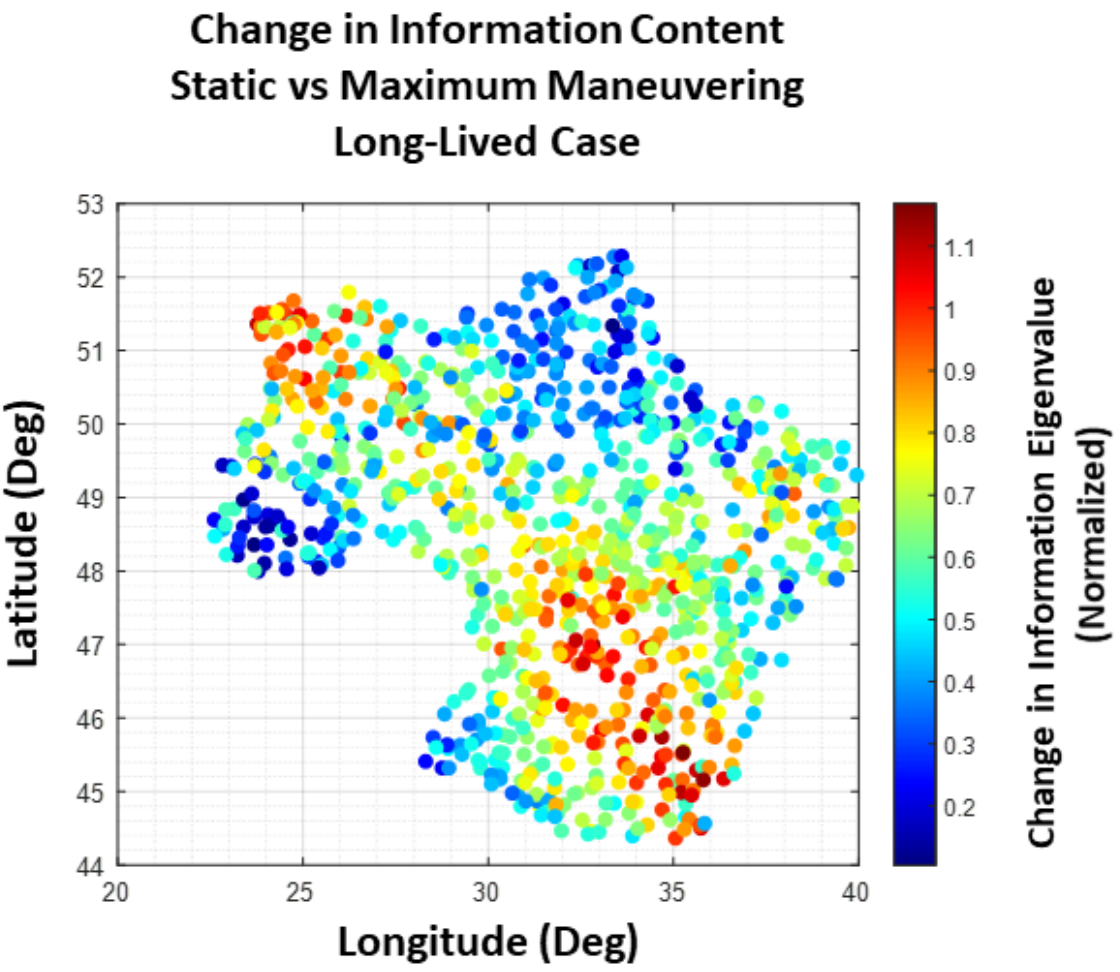

Figure 3-20: This map plots how much additional information content (in terms of Fisher information) was added to the component of greatest uncertainty at each site in the max maneuvering case compared to the static case for the regional scenario. Dots show sites which are improvements and squares show sites which worsened. This was for the case with 3 day lifetimes for the collection sites. Most sites improved, and by magnitudes significantly greater than the other sites degraded.

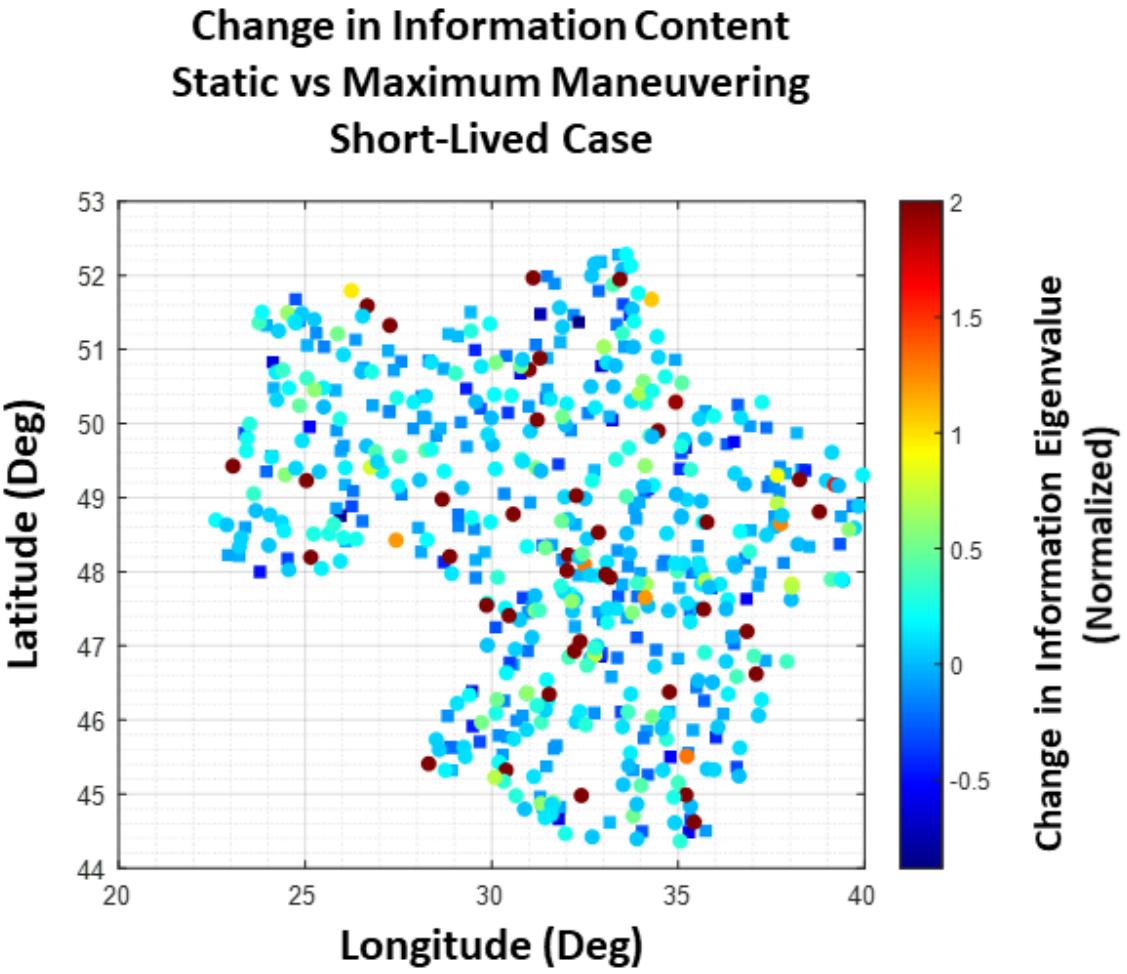

Figure 3-21: This map plots how much additional information content (in terms of Fisher information) was added to the component of greatest uncertainty at each site in the max maneuvering case compared to the static case for the regional scenario. Dots show sites which are improvements and squares show sites which worsened. This was for the case with 6 hour lifetimes for the collection sites. Most sites improved, and by magnitudes significantly greater than the other sites degraded.

Chapter 4

Case Study 2: PEARL

This chapter analyzes a case study involving the PEARL platform described in section 1.3.1. There are already proposals in MIT's Engineering Systems Laboratory to send a PEARL platform on a cross-Atlantic journey from Boston to the Azores. This case study simulates such an oceanic journey over a period of 3 months, and attempts to optimize the path PEARL takes to maximize the utility it can provide to a set of LEO satellites were it to host a SPARC-like set of mirrors for vicarious calibration.

Scenarios of this type are similar to problems commercial calibration networks will face with mobile calibration sites. Given some set of customer satellites requiring calibration over some campaign period, what is the optimal set of maneuvers a mobile calibration should take? How can it be done while beginning and ending at specified facilities? This case study tackles a specific instantiation of both those problems. The PEARL platform is simultaneously required to begin at Boston, end in the Azores on a given date, all while attempting to place itself in favorable calibration positions. Like the previous case study in Chapter 3, there are a large number of collection sites (in this case collection satellites), and as such this problem is a combination of prioritizing the list of potential satellites and optimizing to those priorities.

In a sense, this study with PEARL is an inverse situation to the ReCon case; ReCon has a series of space-based vehicles maneuvering to interact with target sites on the Earth, while this PEARL case has a mobile Earth vehicle maneuvering to interact with non-maneuvering target spacecraft. This duality (along with my familiarity and

proximity to the PEARL project team) led to the selection of PEARL for case study 2.

The multi-objective problem is to (1) minimize the total amount of distance travelled by PEARL (a fuel-analogue) while (2) maximizing the calibration potential the platform presents to a set of overhead LEO satellites, all while planning a valid travel path which leaves Boston and arrives at the Azores 90 days later. The system parameters include the maximum speed of PEARL (relative to the surface ocean currents), the satellites and their trajectories, the elevation constraints for successful calibration, and the simulation date (used for positioning of the sun). Constraints include regions of operation, and the requirement to end the journey in the Azores. The chromosomes specify the path PEARL is to take on this journey and the fitness function is a time-stepping simulation of PEARL along this path, tallying opportunities for satellite calibration.

4.1 Calibration Potential as an Objective

The modelled PEARL platform hosts a set of steerable mirrors to engage in vicarious calibration akin to those described in section 1.3.2. This instrument reflects the sun as a calibration source towards overhead LEO satellites and their optical instruments. The scenario posits a set of LEO satellites in need of calibration, a set of potential customer satellites which will I will refer to as *target satellites*. There are 50 target satellites, a number large enough and with orbits diverse enough that the numerous target satellites present competing alternatives for PEARL's path. Thus, this situation, like the previous ReCon case study, presents a prioritization problem. On the other hand, because there is only one PEARL platform in this case study, the assignment problem is trivial.

The actual metric used as an objective value is the total number of *satellite-seconds* of potential calibration. Every timestep, each satellite which could potentially use the PEARL platform for vicarious calibration credits Δt to this quantity, where Δt is the length of the simulation timestep. As this calibration method is based on reflecting the

sun towards the satellites, requirements are placed on the viewing geometry of both the sun and the target satellite in question. Minimum elevation limits are specified as system parameters for both the sun and the satellites from the location of PEARL, and both must be met for a target satellite to be credited as a potential calibration event. Weather is out of scope for this case study; clear skies are assumed. In addition, the satellites are assumed to have enough pointing authority that the optics can always be pointed towards PEARL (at least within the specified elevation limits).

4.2 Variable Length Chromosome Formulation

The chromosome for an individual in this case study describes a set of times and waypoints along the surface of the Earth in latitude and longitude. Each gene specifies a single one of these waypoints. Specifically, a gene contains fields for platform identifier, waypoint latitude, waypoint longitude, and waypoint validity time. A platform identifier is included because, despite the fact that this simulation includes only a single PEARL platform, the simulation as coded can easily be configured to co-optimize the paths of multiple platform. Table 4.1 summarizes the formulation.

Because of lessons learned from the Rain Catcher toy problem (described in section 2.6.4), an ordering was defined for the genes. The ordering is simply a comparison first by time, and then by platform number. An equality comparison was also defined, and equality is declared if two genes are equal in time, platform identifier, and waypoint lat/lon.

The gene mutation operator follows similarly to that used in the ReCon case study. When mutated, a gene has new values drawn from uniform distributions for latitude, longitude and waypoint time. Time is drawn from a uniform interval between 0 and 90 days. Latitude and longitude are drawn from a uniform (in the Mercator sense) distribution across a square within the Northern Atlantic. For reasons described in section 4.4.2, any waypoint location on Earth is feasible, but land or no-go considerations will affect the routing of PEARL towards that waypoint.

Table 4.1: The Gene for PEARL describes a waypoint along its path. The limits reflect the specific system parameters chosen in this study.

Field	**Description**	**Limits**	**Data Type**
Platform Number	Identifies Platform this Gene affects (N/A this study)	[1,1]	Int64
Latitude	Waypoint Latitude (deg)	[0, 70]	Float64
Longitude	Waypoint Longitude (deg)	[-50, 0]	Float64
Time	Validity time of waypoint (sec)	[0,90] days (expressed in seconds)	Float64

4.3 Satellite Orbit Selection and Propagation

The set of target satellites consisted of 50 LEO space platforms. Orbits in the LEO regime were chosen specifically to be best representative of the types of satellites which are ideal customers of vicarious calibration; small and relatively inexpensive Earth-imaging satellites which do not carry their own calibration instruments, and otherwise may have to use a "Big Satellite" method. The 50 satellites are based on real-life LEO satellites, as categorized by the US Space Surveillance Network and published on Space-Track [17]. Space-Track provides the orbits of unclassified satellites as tracked and determined by the US government in the form of two-line element sets (TLEs).

Space-Track allows the download of a subset of the catalog defined by orbital regime. For the purposes of this case study, the first 50 satellites from the LEO subset were chosen as target satellites. Propagation of TLEs was done using the Standardized Astrodynamic Algorithm Librarys (SAALs) [51] [75] [18]. This library was specifically chosen to keep the most accurate representation of the satellite trajectories at the chosen epoch. The fields in the TLEs are set not to best represent the quantities of physical reality at the epoch time, but instead are the Simplified General Perturbation Model 4 (SGP4) input which produces the best fitting trajectory to reality. The SAAL provides an authoritative implementation of SGP4. The developers of the SAAL specifically recommend the use of the SAAL for TLE propagation with this

model matching justification [21].

The portions of the SAAL which do not fall under export-control, which luckily includes SGP4 (the model for propagating TLEs), are provided free for public use. It comes in the form of compiled shared libraries, which can be called by any language (notably Julia) which can link to C code. The TLE propagation and some coordinate transform standards were utilized by the fitness function simulation for this case study.

The trajectories of the target satellites are constant; they will not change while evaluating different individuals in the population. Thus, there is a large computational savings gained by pre-computing their positions. As such, the 50 satellites were propagated and their positions recorded in the ECEF frame. This ephemeris contains data at a temporal resolution of 5 seconds; the Δt which is used in all the optimization runs discussed for this case study. SGP4 is a continuous model, so the choice of time step does not affect the position accuracy of the target satellites. Rather, it only affects the accuracy of the integration of PEARL's position and the temporal resolution of evaluating the calibration potential objective discussed in section 4.1. To save memory, this ephemeris was stored at single precision rather than double, however, at the altitude regimes of these satellites, this corresponds to an error of only 0.5 to 1 meter. That error is irrelevant to the elevation calculations used in the fitness function simulation.

4.4 Ocean Navigation Simulation

4.4.1 Current-Aware Navigation Simulation

The virtual PEARL platform navigates the ocean with the effects of ocean surface currents included. Information on ocean surface currents was extracted from NOAA's GlobalReal-Time Ocean Forecast System (RTOFS) [5]. The RTOFS is based on an Hybrid Coordinate Ocean Model by Chassignet et al. [31]. It provides many variables of data for use in weather and marine prediction, one of which is predicted ocean

surface currents (both direction and magnitude).

The RTOFS provides snapshots of its predictions for 1 to 2 days. Unfortunately, no large archive of historical snapshots could be located, and the current model used in this case study is based on a single day. Figure 4-1 shows this snapshot. Generalization to time-dependent and/or uncertainty in currents is left to future work.

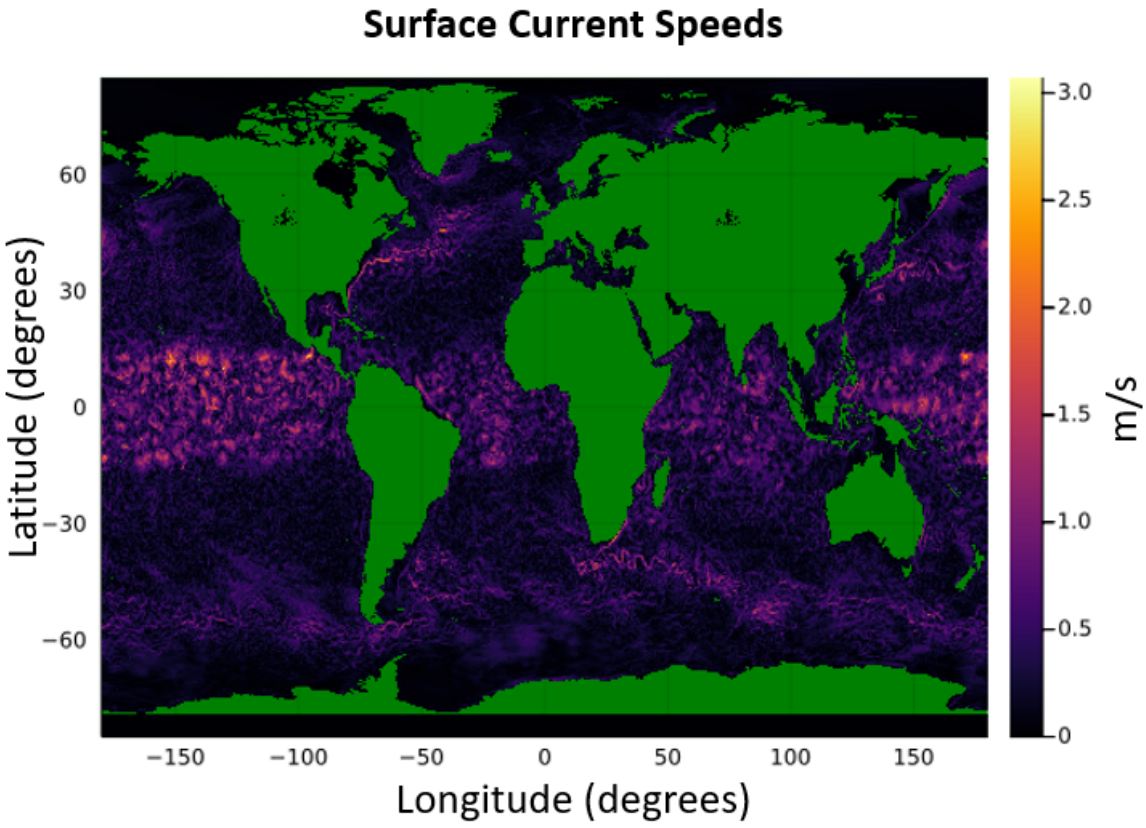

Figure 4-1: Magnitude of ocean surface currents as predicted for NOAA's RTOFS. This is for the day of Feb 22, 2022, and was used in the modelling for this case study.

PEARL's movement is integrated in a simple forward Euler matter at the timestep Δt of the scenario. Given the relatively small speeds of PEARL, this scheme remains stable for the timesteps required for reasonable resolution of calibration potential. The Δt chosen for the simulations in this chapter was 5 seconds, a tradeoff between the temporal resolution of calibration opportunities and computational tractability.

At any given timestep, a desired bearing for PEARL is chosen. This bearing is either (1) the great circle direction towards the gene-specified waypoint whose validity

time last occured in the past, (2) the origin point of PEARL if no waypoint has yet become valid or (3) the bearing along the most efficient route from PEARL's current position to the destination in the Azores. Choice 3 occurs when it is estimated PEARL must start steaming to the Azores to arrive in time. Once the bearing is decided, it is compared to the surface currents at PEARL's position. If PEARL has enough control authority, it will travel directly upon this bearing, i.e. PEARL steams in a direction such that the vector combination of the surface current and PEARL's movement results in the desired bearing. Otherwise, in the case that some perturbation to the bearing must be accepted, PEARL simply steams directly in the desired bearing and the currents are allowed to perturb the effective bearing. Figure 4-2 shows this scheme. When PEARL is travelling towards a gene-defined waypoint, the Haversine formula calculates the current distance from PEARL to said waypoint:

$$d = R_E \left(2 \tan^{-1} \left(\frac{\sqrt{a}}{\sqrt{1-a}} \right) \right) \tag{4.1}$$

with

$$a = \sin^2 \left(\frac{\Delta\phi}{2} \right) + \cos\phi_1 \cdot \cos\phi_2 \cdot \sin^2 \left(\frac{\Delta\lambda}{2} \right)$$

and ϕ is latitude, λ is longitude, and R_E is the radius of the Earth.

If PEARL is expected to reach the waypoint in the next timestep, its position is pinned to the waypoint. Otherwise, the latitude and longitude of PEARL are integrated with an explicit forward Euler step based on the vector combination of surface currents, PEARL's maximum speed, and the bearing it steams in.

Efficient look-up of surface current values is key to the computational performance and overall tractability of the fitness function. The RTOFS does not provide values over a uniform latitude/longitude grid. A uniform grid would allow the constant time $O(1)$ lookup of current values. The non-uniform coordinate grid necessitates the use of an data structure with efficient query performance. The R* Tree [24] is such a structure and can handle 2D data points. Based on the earlier R-Tree [42], R* Trees, divide multidimensional data into hierarchies of multi-dimensional rectangles until the data points (or shapes) are stored in leaf nodes. This allows an efficient query with

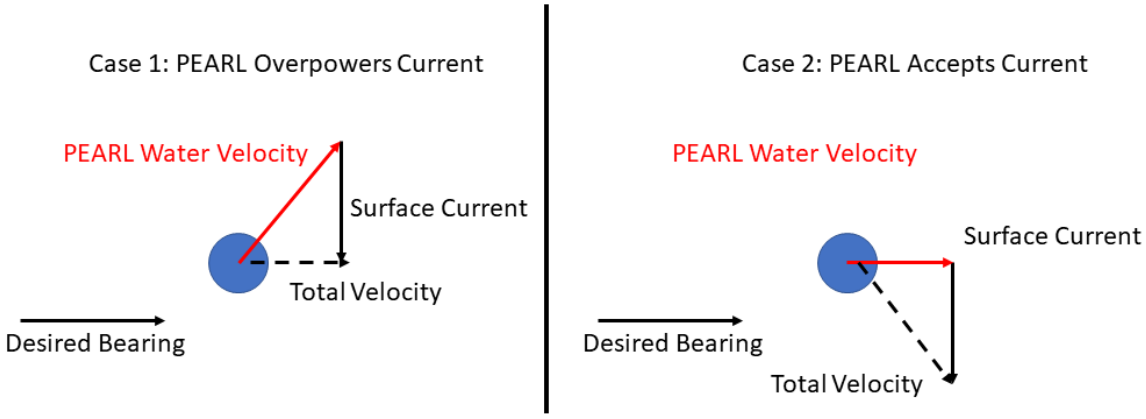

Figure 4-2: The two cases for PEARL's simulated navigation control. [LEFT] PEARL is has enough steaming ability to cancel out all orthogonal surface currents to the desired bearing. [RIGHT] The current is too powerful for PEARL to overpower; in this case, PEARL steams directly in the bearing desired and accepts all drift from the current.

logarithmic performance. A very simplified version of an R-Tree was implemented in Julia for storing the surface current information from the RTOFS snapshot. it was not necessary to implement all the heuristics and optimal balancing algorithms from [42] or [24] to achieve the needed performance; the $O(\log N)$ asymptotic time from the tree structure was sufficient.

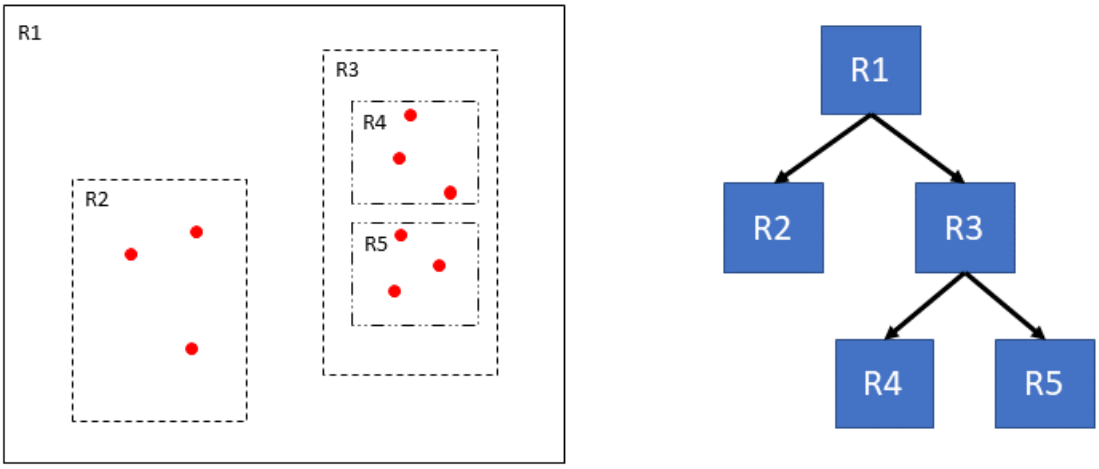

Figure 4-3: The structure of an R-Tree. The 2D domain is divided up into a hierarchy of rectangular subdomains. The data points are stored at the leaf rectangles. More advanced versions have heuristics and algorithms to keep the tree optimally balanced.

Because PEARL is required to end up in the Azores at the end of the scenario

period, Djikstra's algorithm [36] generated the optimal routes to the Azores from any ocean location in a current-aware manner. The RTOFS provides current information in a 2D array, just one that does not correspond to a uniform grid over the Earth. Taking the sample points of the RTOFS as the nodes of the graph, and considering nodes whose values are stored adjacent in the 2D array as connected, Djikstra's algorithm calculated the travel time from any such point to the Azores, as well as the bearing PEARL should travel in to take at any point. The implementation is summarized as follows:

1. Set the travel time for the node corresponding to the destination in the Azores to 0. Set the travel times of all other nodes to 0. Create a queue of nodes and add the destination node to it.

2. Take the node in the queue with the lowest travel time as the "current" node.

3. For each neighbor of the current node, calculate the time it would take PEARL to travel from the neighbor to the current node. Add this time to the travel time of the current node. If this time is less than the travel time currently listed on the neighbor, update the neighbor's travel time and travel bearing before adding the neighbor to the queue. Don't add the node to the queue if the neighbor is on land.

4. Repeat steps 2 and 3 until the queue is empty.

For efficient query performance, this optimal routing information was stored in the same tree data structure as the surface current information. The travel times and routes depend on the maximum speed of PEARL. As such, different versions of this tree structure were generated and saved for different velocity capabilities of PEARL. Julia contains built-in serialization capability, and this native ability recorded the tree structures onto the disk. Worker nodes used in the SLURM cluster loaded these files to make use of these pre-computed results.

The optimal routing information enables PEARL to determine at any moment in time when it needs to start steaming on the route to the Azores. These times are

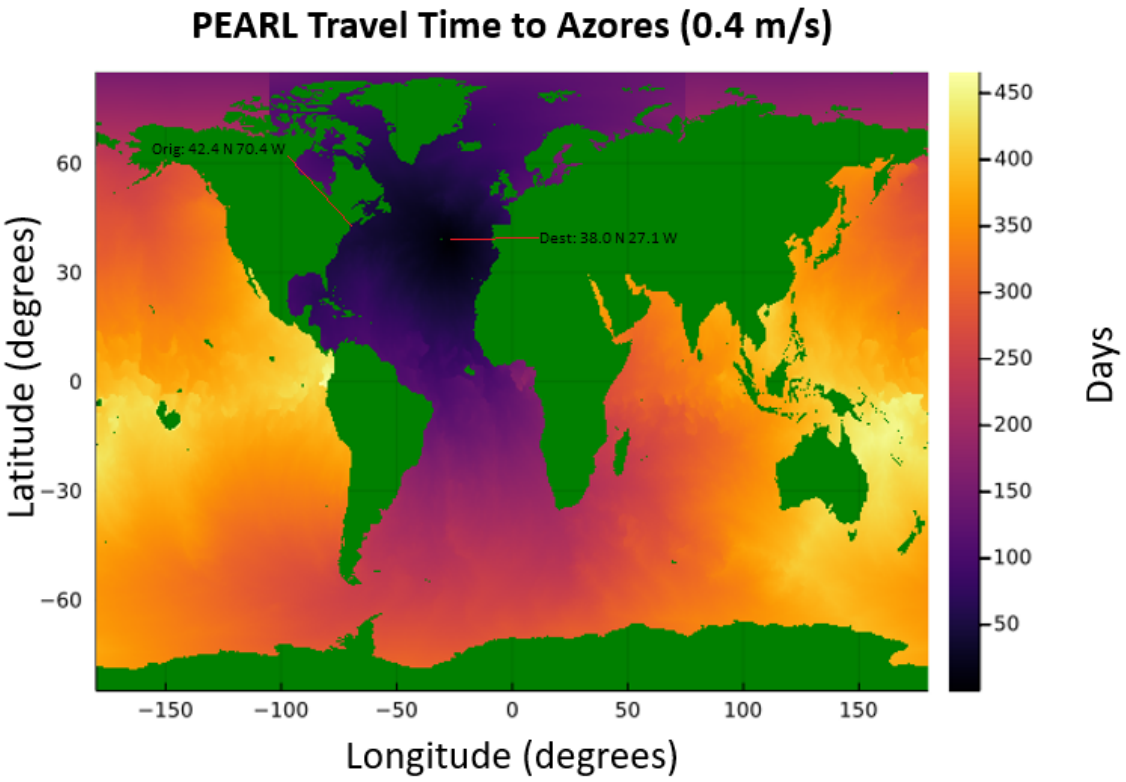

Figure 4-4: A map of the travel time from any point in the ocean to the Azores with a PEARL platform which can travel at 0.4 m/s. These times are the optimal routes using global knowledge of the surface currents. The Mediterranean is not shown as PEARL at this speed cannot overcome the currents in the Strait of Gibraltar. The origin near Boston (42.4 N, 70.4 W) and the destination in the Azores (38.0 N 27.1 W) are labelled. These points are the beginning and end of all PEARL transits in this case study.

shown in figures 4-4 and 4-5. The simulation code in the fitness function declares this "drop-dead" condition when the remaining time in the scenario is less than the reported Azores-travel-time at PEARL's location. Once this condition is declared, PEARL ignores any remaining gene-defined waypoints and steams along the optimal bearing.

4.4.2 Satisfying Navigational Constraints

PEARL is not free to travel anywhere across the globe. It obviously must stay in the ocean, but there all sorts of complex considerations such as shipping lanes and coastal

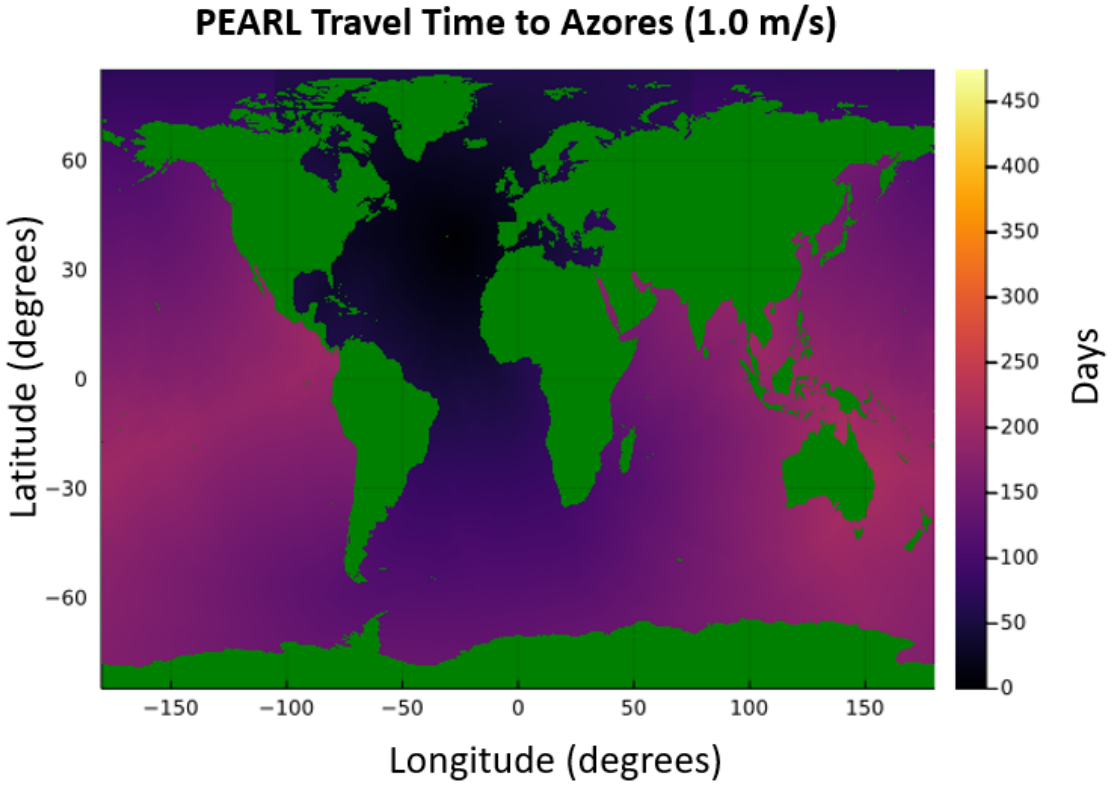

Figure 4-5: A map of the travel time from any point in the ocean to the Azores with a PEARL platform which can travel at 1.0 m/s. These times are the optimal routes using global knowledge of the surface currents. In comparison to the 0.4 m/s case, the Mediterranean is now available.

safety which in any real-life application need to be encoded. In addition, there is the aforementioned requirement that PEARL end its journey at a predictable spot in the Azores.

Implementation of Movement Constraints

A zone of valid travel for PEARL is encoded at as a complex polygon. Any non-overlapping polygon is valid; concavity and holes are allowed. The simulation code in the fitness function enforces this constraint, rather than, for example, a penalty-function in the optimization hierarchy. At any timestep, if the predicted location of PEARL in the next timestep is outside this polygon, navigation logic commands PEARL to hold at its current location. In effect, this means that any set of maneuvers as specified by an individual's chromosome is feasible. PEARL will simply stop at

the polygon boundary, and the exact location of this halt is still dependent on the waypoint, even if that location is far outside the boundary. This scheme sidesteps the issue of choosing penalty function weights. The algorithm for determining whether PEARL is in or out of the polygon is shown in figure 4-6 and the constraints used for this case study are shown in figure 4-7.

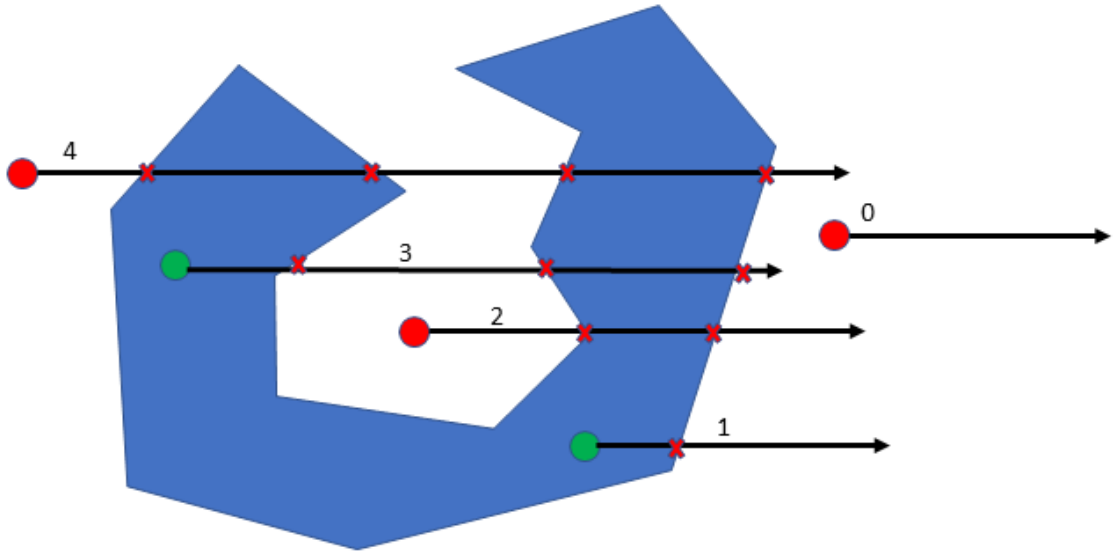

Figure 4-6: The algorithm for determining if PEARL is inside the constrained area. The blue polygon represents the allowable area for PEARL to travel in. A ray is cast due East (though any direction would work) and the number of interceptions with the line segments defining the polygon are counted. An odd number of crossings indicates PEARL is in the polygon. An even or zero crossings indicates PEARL is outside. The numbers quantify the number of crossings for each ray.

Terminus Constraints

Any PEARL path is constrained to end reasonably close to the Azores. The fitness function once again enforces this constraint. As described earlier in this section, the optimal routing information generated with Djikstra's algorithm is consulted at every timestep to determine when to ignore remaining waypoints and begin steaming towards the Azores. Like the polygon constraint, this scheme keeps any waypoints technically feasible and side-steps the choosing of weights for a penalty function.

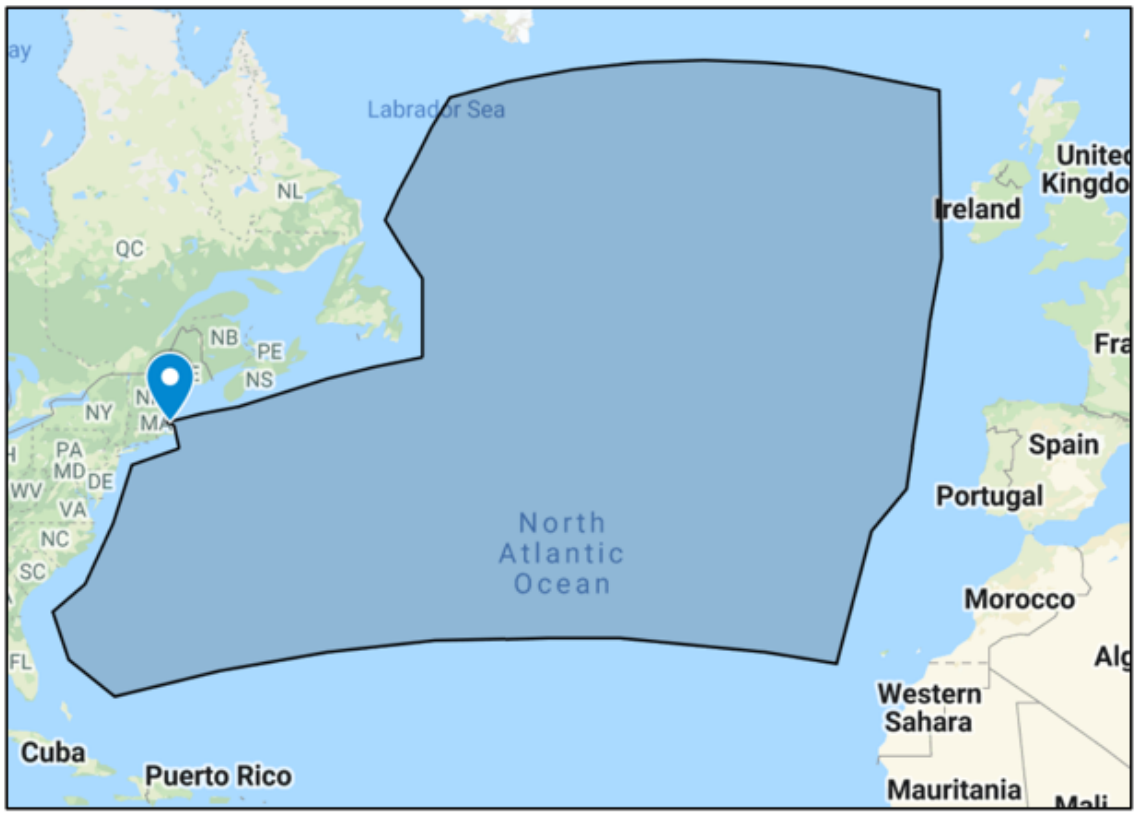

Figure 4-7: The constraints imposed on PEARL's location for this case study. For the most part, these were chosen to keep PEARL acceptably away from the coasts. The Southern boundary was arbitrary, but will act as an active constraint in more capable optimization runs.

4.5 N2 Diagram

The N2 diagram for the fitness function evaluation is shown in figure 4-8.

Feedback loops are avoided except for the dependence between PEARL's trajectory and its ability to steam in a direction. The use of pre-computing the positions of the satellites, the sun, and the optimal Djikstra routes to the Azores removed the need for several modules and came with considerable computational savings.

4.6 Results

The framework produced Pareto fronts for two sets of conditions, differing in the maximum velocity of the PEARL platform. The first case had PEARL with a max-

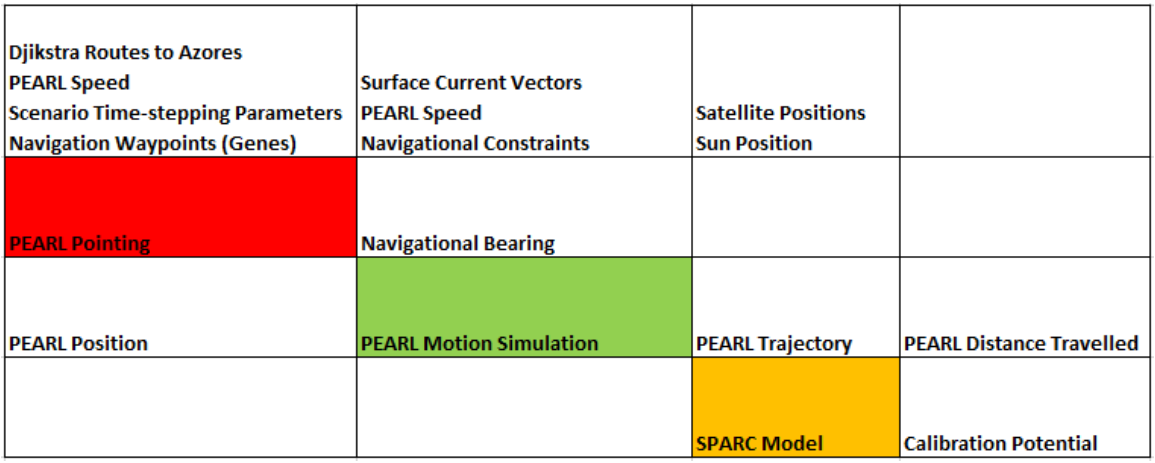

Figure 4-8: The N2 diagram for PEARL's fitness function/simulation.

Table 4.2: The system parameters used in PEARL optimization runs

Parameter	**Value**
Population	2048
Chance of Gene Mutation	0.03
Chance of Gene Copy	0.01
Chance of Gene Delete	0.01
Rel. Tolerance for Termination Criteria	0.01
Termination Max Number of Unchanged Generations	40
Solar Elevation Limit	45 deg
Satellite Elevation Limit	0 deg
Scenario Duration	90 Days
PEARL Maximum Speed	0.4 m/s, 1.0 m/s

imum water-speed of 0.4 m/s, which is the demonstrated effective speed of current prototypes. This accounts for inefficiencies from the motion controller loop. The second case uses a maximum water speed of 1.0 m/s. This is a hypothetical upgraded version of PEARL, and while possible, represents a rather high-performance model if using current technology. Table 4.2 shows the system parameters used in the PEARL optimization runs.

4.6.1 0.4 m/s Velocity Data

The Pareto front for the 0.4 m/s case is shown in figure 4-9. The non-dominated routes are plotted on the map in figure 4-10.

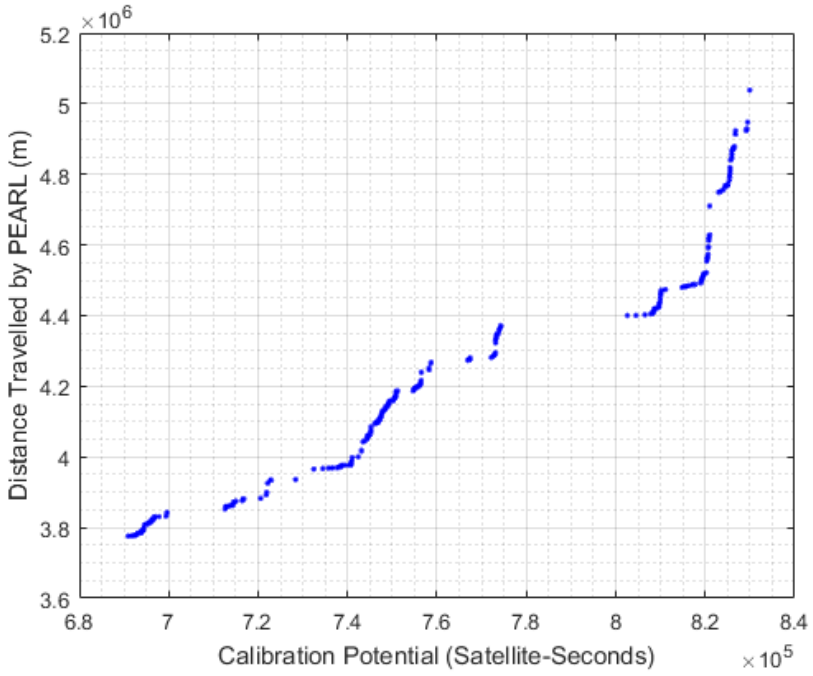

Figure 4-9: The approximated Pareto front for a PEARL transit from Boston to the Azores at 0.4 m/s.

4.6.2 1.0 m/s Velocity Data

The Pareto front for the 1.0 m/s case is shown in figure 4-11. The non-dominated routes are plotted on the map in figure 4-12.

4.6.3 Discussion

As one would expected, the more capable PEARL platform performs significantly better than the less-capable case. The Pareto fronts for both are overlayed in figure 4-13.

The difference in calibration potential increases as the routes become longer. This suggests that the larger utility in the 1.0 m/s case comes from access to areas not

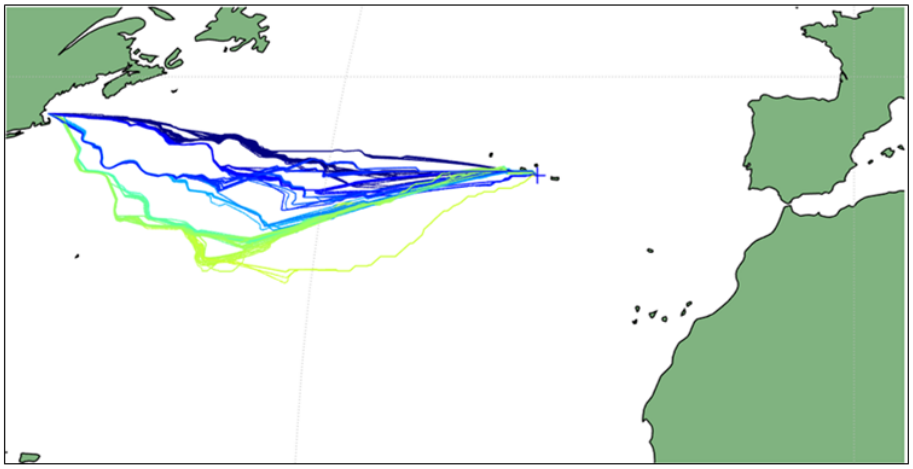

Figure 4-10: The non-dominated routes for PEARL in the 0.4 m/s velocity case. The colors represent the relative performance in the calibration potential metric.

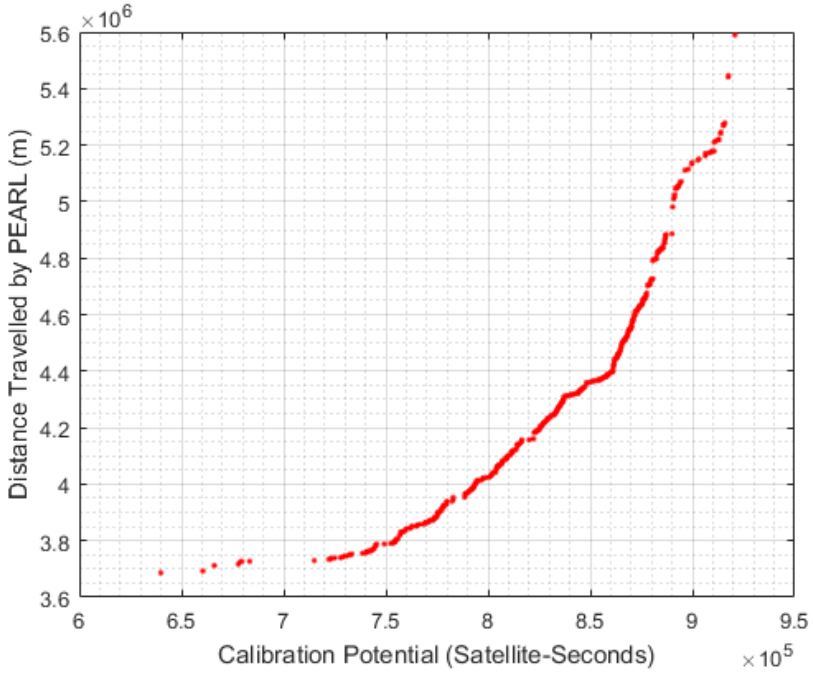

Figure 4-11: The approximated Pareto front for a PEARL transit from Boston to the Azores at 1.0 m/s.

available to the 0.4 m/s platform, rather than the ability to loiter longer in some favorable area available to both platforms. Indeed, the map of routes for the 1.0 m/s case show a significant dip towards the South, only stopped by the (somewhat arbitrary) movement constraint imposed.

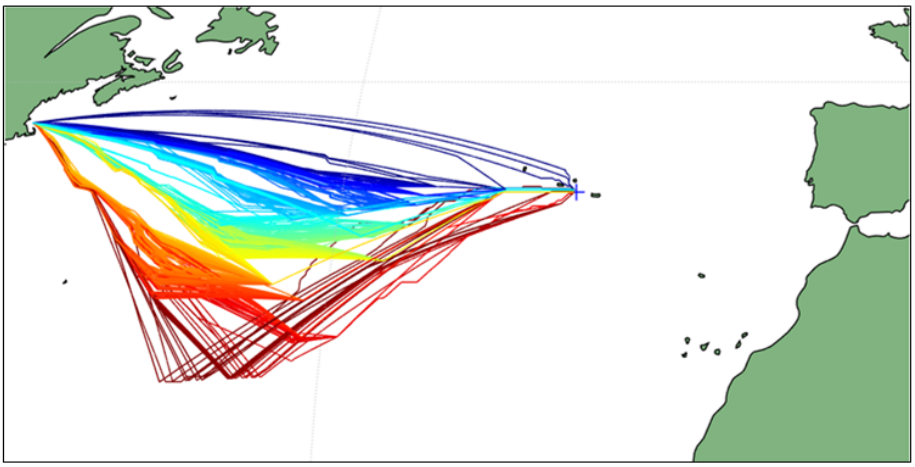

Figure 4-12: The non-dominated routes for PEARL in the 1.0 m/s velocity case. The colors represent the relative performance in the calibration potential metric. It is on the same scale as figure 4-10.

A possible explanation comes from the distribution of satellites with respect to latitude. The target satellites orbit in planes at a diverse set of inclinations. A given target satellite will only travel the latitudes less than or equal in magnitude. There are thus more satellites at lower latitudes than higher. This can be seen directly in the distribution of satellites over latitudes.

The overlay of ground traces in figure 4-14 show specific bands of heightened target satellite density. These bands probably correspond to the latitudes of rocket launch sites. It is most energetically favorable to launch due East from a launch site, and this results in an orbit with an inclination approximately the latitude of the launch site. This energetically favorably insertion strategy privileges specific inclinations, resulting in the observed bands. Figure 4-15 shows a histogram of target satellite position latitudes over the entire scenario timeline. In particular, the histogram shows Boston, the origin of PEARL, is at a particularly disadvantageous latitude. The drift South is movement towards the closest peak of satellite density.

In addition, there is the added factor of sunlight. Since the location of PEARL is required to be sunlit, in fact, with a minimum solar elevation, lower latitudes are privileged simply by the fact that there are longer daylight hours. More daylight hours and more satellite density results in a significant boost in calibration potential.

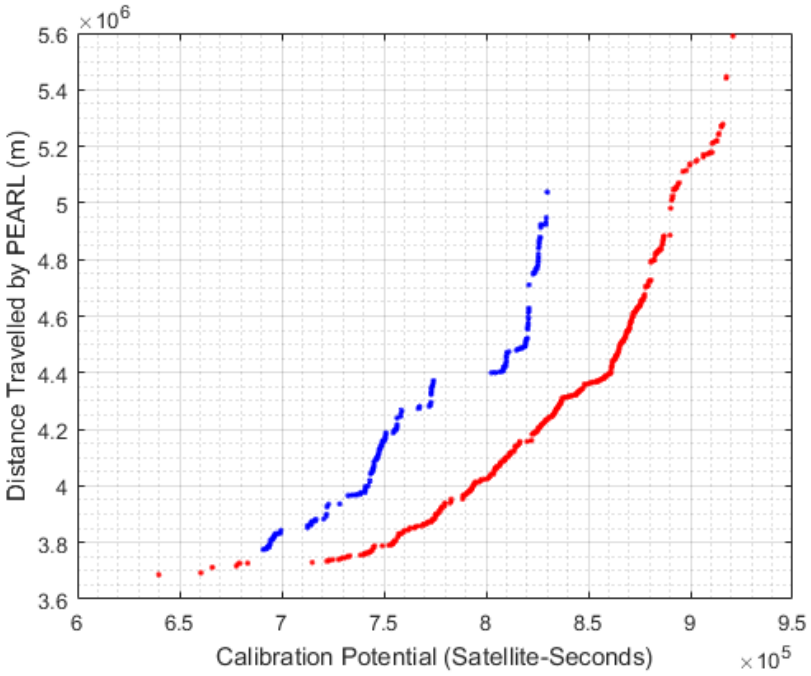

Figure 4-13: The approximated Pareto fronts for PEARL steaming at both 0.4 m/s (blue) and PEARL steaming at 1.0 m/s (red).

Another notable effect is the difference in the ability of the VLC GA framework to approximate a Pareto front between the two cases. Indeed, while the 1.0 m/s case has the entire population of individuals non-dominated, there are a large number of dominated solutions in the population of the 0.4 m/s case. In fact, this appears to be the only case of all the optimization runs done in this thesis where every individual did not join the Pareto front before termination. Figure 4-16 shows the dominated solutions overlayed on the non-dominated solutions for the 0.4 m/s case.

This seems to be arising from some interaction with the satellite elevation constraints. When optimizing such that PEARL is graded solely on how often it can see the sun (a scenario investigated only due to a previous bug in the code), both velocity cases end with all individuals on the Pareto front. Given the dependence on PEARL's speed and satellite viewing, a possible explanation is that even with 50 LEO satellites, there exists significant amounts of sunlit time with no satellite in view. In such a case, an individual may be perturbed a significant amount, making a significant change in distance travelled, but no change in the calibration objective at

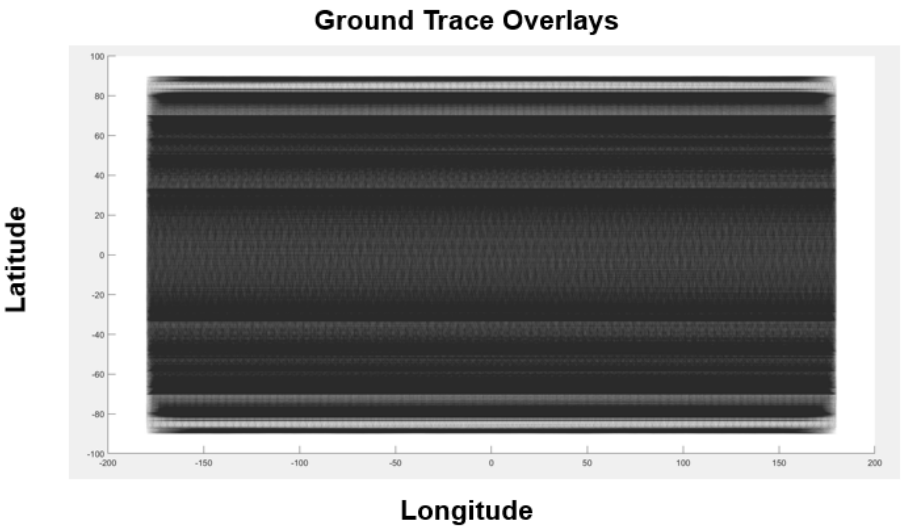

Figure 4-14: The ground traces for the 50 target satellites in the PEARL case study for the entire 90 day period overlayed. Bands can be seen corresponding to common inclinations.

all. Only when a satellite is in view for some of this perturbation will any effect on the calibration objective occur. This creates a type of binary step-like behavior. A step-like shape in the Pareto front is very sparse, only those solutions on the "corners" of said steps can remain non-dominated. A higher PEARL speed possibly negates this problem because the time under no satellites is simply easier to minimize when one can travel faster, so leaving the 0 gradient area of the fitness space is less difficult.

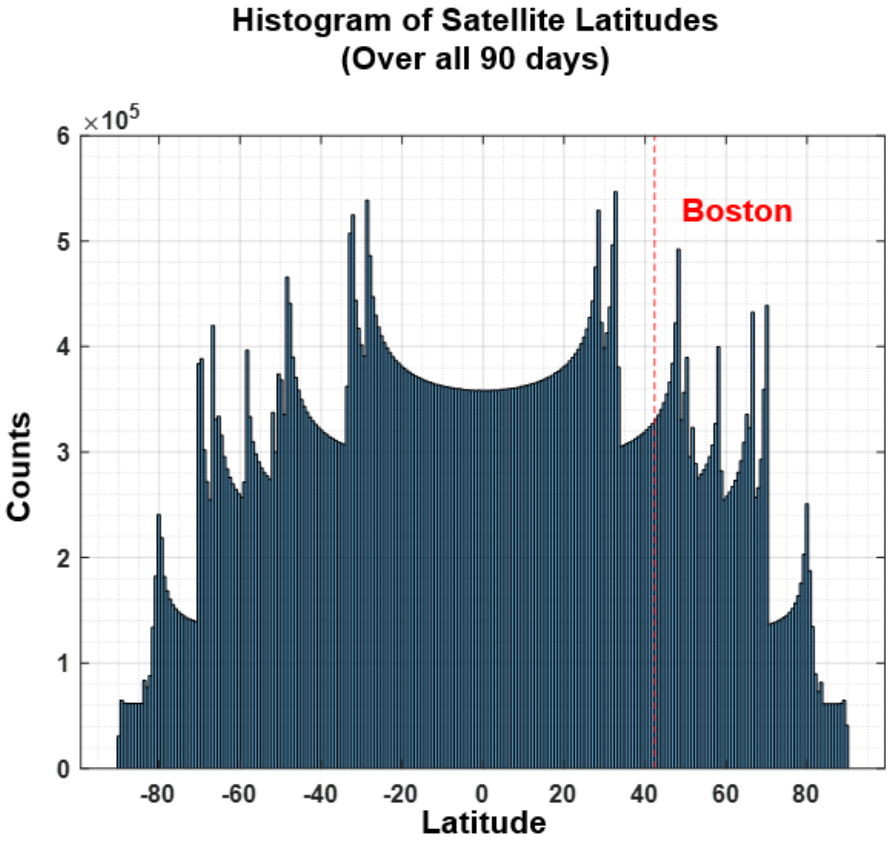

Figure 4-15: A histogram of the 50 target satellites' latitude over the entire 90 day scenario. Structure shows that Boston is particularly disadvantaged. This possibly explains why the trajectories with larger calibration potential drift South.

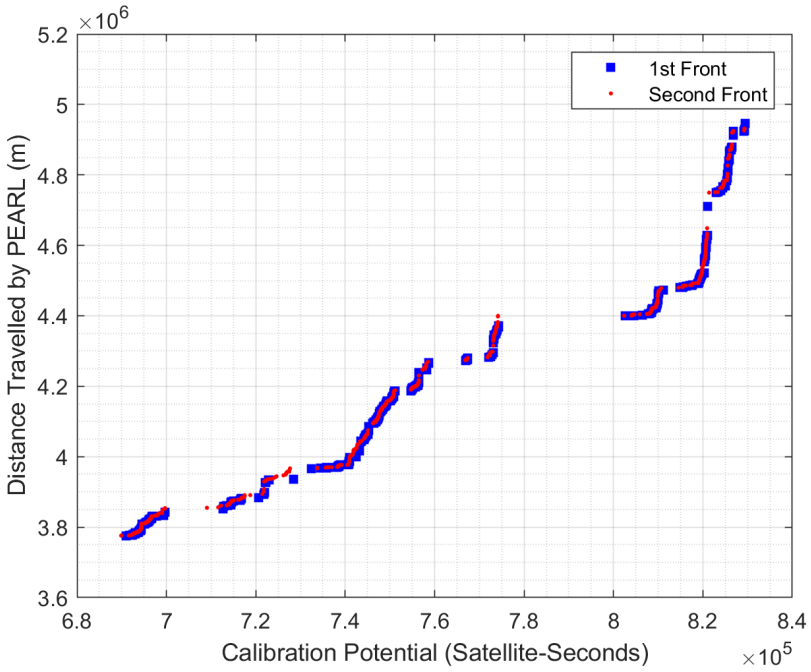

Figure 4-16: The 0.4 m/s case did not converge with all individuals as non-dominated solutions. Only about a fourth of the population is part of the approximated Pareto front, although the other solutions are quite close.

Chapter 5

Conclusions and Future Work

5.1 Summary

This thesis aimed to develop a framework for solving vehicle maneuver problems with a genetic algorithm approach combining multi-objective optimization based on NSGA-II with variable length chromosomes before validating its performance with case studies.

Chapter 1 outlined the motivation for this work, citing the value of reconfigurability and outlining some of the challenges associated with choosing optimal sets of maneuvers. Then, background was presented on both ReCons and PEARL, which are used as real-life case studies for the framework in chapters 3 and 4. Finally, a primer on genetic algorithms with specifics on NSGA-II and variable length chromosomes provided the necessary background concepts incorporated into the framework.

Chapter 2 provided a motivation for the choice of a GA based method to solve the vehicle maneuver problem, outlining how many of the characteristics of this family of problems made application of more traditional optimization methods problematic. A detailed description of the combination of VLCs into NSGA-II as the high-level framework was presented, followed by details on how a specific problem is implemented into the framework. Then, a discussion on computational performance outlined key design decisions and a strategy with distributed computing which combine to maximize computational time efficiency. Finally, a 1D toy problem was presented as a worked

example using the framework.

Chapter 3 presented the first case study, the optimization of a program of maneuvers for a ReCon tasked with collecting on a large number of target sites. This situation differed from previous described work which posed the problem as collecting on targets one at a time. In addition, a complex performance metric based on the Cramer-Rao lower bound for position uncertainty modelled the complex considerations of satellite imagery use for applications in photogrammetry. The formulation of the VLCs encoding vehicle maneuvers was provided as well as details on the simulation used as a fitness function. Results were presented for both a global case and regional case.

Chapter 4 presented the second case study with a PEARL platform travelling across the Atlantic. The simulation used a set of waypoints encoded by the VLCs to plot PEARL's path across the Atlantic and evaluated the platform's potential use as a vicarious calibration site to a set of overhead LEO platforms. A variant on an R-Tree was used in combination with Djikstra's algorithm for efficient incorporation of ocean current information and routing. Results were presented on two cases: the current version of PEARL and a hypothetical improved future version with a large water-speed.

There are three main contributions this work offers:

- A general formulation of a VLC GA framework for vehicle maneuvering. This framework provides a way forward for multi-platform collaborative mission design. This approach can handle a large variety of these problems, many of which contain fitness domains and constraints which are problematic for non-heuristic optimization algorithms.

- A computationally efficient Julia implementation of that framework, adapted for use on SLURM distributed computing clusters. There was significant effort spent on code efficiency and distributed parallelism. That focus exploits modern improvements in computing infrastructure, using literal thousands of cores simultaneously to extend numerical optimization to problems requiring a higher

level of computational complexity. This implementation is designed to be easily extensible to other problems beyond the case studies here.

- An analysis of two real-world situations which embed accurate governing physics-equations into their fitness functions. The insights from these two studies can feed into the evolution of both these concepts. In addition, the Julia implementations for these studies can serve as a starting point for future analysis which uses the VLC GA framework.

5.2 Conclusions

5.2.1 Optimization of Vehicle Maneuver Problems in General

This thesis aimed to develop a VLC GA framework geared towards vehicle maneuver problems. It has shown with two case studies that such problems are amenable to this approach, and that useful insights can be gleaned from the framework output. In addition, it demonstrates the problem is embarrassingly parallel and benefits greatly from distributed computing setups. The overall approach is thus limited not by the compute power that fits in a single machine, but instead the significantly larger power afforded by cluster computing solutions.

The early lessons learned coding and analyzing the Rain Catcher toy problem (Section 2.6.4) are general, and inform problems beyond the case studies discussed here. In particular, the potential over-representation of the 0 maneuver case is likely to occur in any multi-objective situation with some sort of energy measure or analogue as a metric under optimization. My experience with reduction operators suppressing the creation of large amounts of inactive genes matches previous literature, although the there might be potential utility to those inactive genes [63] which can yet be exploited. Crossover and mutation can easily switch a gene from inactive to active, and their mere presence in the chromosome effects the probabilities of where mutations might occur or how crossover might play out. It is notable that the biological inspiration, real DNA from organisms, often contains a large amount of these inactive genes.

A few times in this thesis, I made reference to the building-block hypothesis as a narrative explaining the utility and performance of GAs. I believe the performance benefits arising from defining and enforcing an intra-chromosome ordering on genes supports the BBH. The ordering encourages the crossover operator to make genotype divisions in a way which leads to logical phenotype divisions. These logical divisions enable contiguous sections of the chromosome to be a well-tuned partial solution. The crossover operator breaks chromosomes into contiguous sections, and so any "building block" partial solution needs to be represented in this contiguous form.

Both case studies exhibited a similar relationship between the potential performance an aggressively maneuvering case could realize and the various time/distance quantities between vehicles and collection sites. With ReCon, imaging performance was very sensitive to the lifetime of the collection sites, and with PEARL, a greater water-speed extended the upper limit of achievable calibration potential. Very broadly, there is some characteristic time separation and physical separation between collection opportunities, which could be put into a fraction to define a velocity-like quantity. The greater the velocity of the vehicles in comparison to this faux-velocity, the greater the performance ceiling. To simplify in terms of the Rain Catcher problem, the current formulation with instantaneous travel means there must be some set of maneuvers which collects all the drops. Conversely, a case where the platforms have infinite travel time (i.e. a static case) insists there can be no performance improvement. In between these extremes, the total potential improvement depends on the distance extent rain is falling over, and how much time is generally between two drops. Of course, the exact performance ceiling relies on the specifics of the rain program and the initial configuration of the platforms.

5.2.2 Conclusions Specific to the Case Studies

ReCon

With ReCon, we can quantify the value of agility in satellite constellations, tracing out a trade-off curve between increased photogrammetric performance and total expended

delta-V. In the global and regional scenarios modelled, with many simultaneously active and geographically diverse collection sites, the most aggressive maneuvering improved the photogrammetric objective respectively by approximately 25 and 35 percent. While this has not been placed in terms of raw dollars like previous work [58], it does present a quantifiable cost-performance tradeoff arising from satellite agility. Notably, these aggressively maneuvering solutions often left a large proportion of the satellites static, with the most aggressive maneuvering leaving 7 to 11 platforms untouched. This would indicate a hybrid solution of static and dynamic platforms best solves the ReCon problem.

We can also identify characteristics in scenarios which suggest large potential gains from maneuvering. Specifically, it is those cases with many long-lived collection sites (long-lived with respect to the the time it takes satellites to travel around the Earth) which benefit the most, even if the sites are geographically diverse. In these cases, the framework was able to find maneuver plans which performed significantly better for most sites, producing a net benefit. Conversely, a scenario with the same target sites, but with shorter lifetimes was far less amenable. It was difficult for the framework to identify net-positive maneuvers in such a situation, as improvement at one site would quite often lead to degradation at others. These trends held in both the global and regional scenario. From this experience, it appears the potential of satellite agility is most sensitive to the time horizon of imaging tasks, rather than geographic considerations.

It should be noted, the previous conclusions are specific to the photogrammetric metric used in this study. It is not a given that other metrics (for example, metrics based simply on the binary visibility of a site to a satellite) would behave in the same manner. That said, this case study shows the applicability of the VLC GA framework to planning satellite maneuvers for Earth imaging-tasks with a complex performance metric.

PEARL

The case study of PEARL demonstrated that over a 90 day transit from Boston to the Azores, there is significant flexibility in the chosen route, and this flexibility can be exploited to increase the utility of PEARL as a LEO satellite vicarious calibration system. In addition, the comparison between the current PEARL and a hypothetical upgrade with better motors demonstrated the calibration potential is sensitive to PEARL's water-speed, with a larger maximum speed increasing performance.

In particular, the data showed an improved PEARL benefited from the ability to travel to points further South, and not just from the potential to loiter longer in some favorable location accessible to the baseline platform. The tracks from both speed cases expressed higher calibration potential the further South PEARL travelled. An examination of the density of the chosen LEO satellites showed this was consistent with increased satellite activity towards the equation, not to mention the simple fact that more sunlight is available towards the equator.

As an overall conclusion, this thesis validates using a VLC GA framework to plan reasonable sea-tracks tuned to a metric. The population of individuals did in fact converge to sensible west-to-east trajectories that do form an approximate Pareto front.

5.3 Future Work

ϵ - NSGA and Other GA Improvements

The framework developed and presented in this thesis used NSGA-II as the basis of its multi-objective GA algorithm. There exist improvements onNSGA-II that might increase the performance and/or utility of this framework. For example, in his work, Legge [58] incorporated aspects of ϵ-NSGA-II and ϵ-MOEA [55].

A particular improvement of interest is the use of an archive to store best performing individuals that can be re-introduced into the population later. In many of the runs undertaken for this thesis, the entire population of individuals ends up as

non-dominated solutions. Once this point is reached, diversity could be increased by removing some duplicate individuals and replacing them with archived individuals. Even without a full archive, previous work has applied substitution operators to a population in order to increase diversity [66]. It is entirely possible, that the Pareto fronts found in this thesis are just small sections of a larger front, constrained because diversity falls away too quickly. These techniques for substitution and archive population may improve convergence or the quality of the front.

Hyper-parameter exploration

There are a number of hyper-parameters that must be chosen by a human designer when running the optimization framework. Judgement must be applied in choosing

1. The size of the population
2. The rate of single gene mutation
3. The rates of gene insertion and deletion

For the case studies presented, the choice of these parameters was ad-hoc. Several sets of values until reasonable convergence (reasonable often being defined as the entire population of individuals spreading out over a Pareto front) was reached in reasonable time. A more rigorous exploration of the hyper-parameter space may lead to improved performance and general insight towards best practices. Of course, the choice of hyper-parameters, although quite similar between the two problems presented in chapters 3 and 4, is almost certainly sensitive to the fitness space of the specific optimization problem. Thus, such a study may not produce generalizable results.

Additional Schemes for VLC Size Changes and Crossover

The framework uses a relatively simplistic form of gene insertion and deletion, as well as VLC-compatible crossover. Other literature describe more complex choices which in general lead to better convergence and/or convergence. For example, a

form of crossover that removes the need for separate insertion and deletion events is described in [57].

Of note, there are significant alternatives to the choice of crossover points used in gene recombination between two individuals. The framework as it stands now simply chooses random points along the chromosomes of both individuals, but methods exist which incorporate similarity metrics between sections of chromosome and/or take inspiration from biological processes [49] [28].

Hybrid Algorithms - Combining GAs and Traditional Methods

As a heuristic class of algorithms, GAs cannot guarantee even the local optimality of the solutions found. There may be potential in applying more traditional optimization techniques warm-started with the a non-dominated solution found by this framework. Such a technique could move the non-dominated point into the nearest local optimum if not already there.

Of course, the challenges discussed earlier in this thesis with these methods still apply. The complexity of the fitness space may not be amenable, even locally, for such methods, especially those that require a gradient to exist. On the other hand, the challenges are still significantly reduced with the use of the output of the GA framework. In particular, the choice of the dimensionality of the maneuver space is decided. Traditional optimization methods warm-started by a GA non-dominated solution can simply use the number of maneuvers present in said solution.

Additional Exploration in the ReCon Case

The case study presented in Chapter 3 is a limited case. The space of potential orbits to maneuver among is limited to circular orbits with altitude limits, and maneuvering is limited to the double Hohmann transfer scheme. There is no reason to expect that this subspace contains the most optimal maneuver plans for a ReCon, and it was chosen to simplify the problem. Expanding the domain of maneuvering schemes and orbits may provide utility to ReCons.

Such an expansion of the search space may introduce such dimensionality that

the problem becomes computationally intractable. At the limit, an optimizer could be allowed to specify any physically possible maneuver, in all 3 dimensions with no synchronization or correlation explicitly enforced by the higher-level framework.

Additional metrics can also be explored. This case study chose a metric agnostic of any specific photogrammetric technique, instead relying on the Cramer-Rao lower bound as an information-theory performance limit. In a more concrete case, when the specifics of instrumentation and data processing are known, this framework can be re-run incorporating those details.

Additional Exploration in the PEARL Case

The calibration potential metric used in the PEARL case study is quite general. It stands in to represent a calibration of a some set of LEO satellites, but a real-life application would introduce complexity. Any commercial calibration-as-a-service scheme would impose additional logic on the minimum frequency between calibration events and may prioritize some customers over others. In addition, varied modalities of calibration may be offered. It is conceivable that these different modalities may require different geometric considerations and would be valued/priced by the commercial entity differently. A benefit from the GA approach of this thesis is that the higher-level framework is agnostic of the inner-workings of any fitness calculation. No gradient information is required, or even approximated by the higher-level framework, and thus the fitness function itself can be thought of as a black-box.

There is also potential to add in whole new dimensions of performance. For example, one of PEARLs hosts a large amount of water quality and oceanographic instruments. The current case study ignores the value provided by those instruments. Future analysis can extend the case study and add their value as an additional objective and explore the interplay between the geographic factors for oceanographic measurement and those identified here in this work for LEO vicarious calibration.

Appendix A

Installation and Usage Instructions for Julia Code

The Julia code used in this thesis is available at https://github.mit.edu/be22600/VLCGA. Any recent version of Julia should work with this code. This git source control will be updated with bug fixes and extensions as they arise. The following dependency packages are needed to fully use this code:

- **JLD2**: Used as serialization for storing some of the pre-computed values from the PEARL scenario.
- **ClusterManagers**: Provides an easy interface to SLURM
- **NetCDF**: Used for reading the data products from RTOFS.
- **Standardized Astrodynamic Algorithm Library (SAAL)**: Not a Julia package. You need this for TLE propagation and some coordinate transforms.

With these dependencies you can run the code from this thesis. The Github provides more documentation for the specifics of running the scenarios.

A.1 Notes on Running on SLURM Clusters

Here are some tips to avoid gotchas when running this code in a distributed SLURM environment:

- The interface to SLURM provided by the ClusterManagers package makes use of the file system when spawning up new worker processes (but only then, not afterward). For large amount of workers, you may need to increase the *ulimit* settings so a large amount of files can be simultaneously held open by a single process.

- The location of the SAAL shared libraries need to be included in your LD_LIBRARY_PATH for any manager/worker process which uses it.

- The SAAL requires a license file (which is provided for free with it). You may need to edit your environment variables to point at this file. If the SAAL is upset, it will fail with a segfault when you attempt to propagate TLEs, with no useful info!

- You may need to modify the code loading the SAAL libraries if the names of the shared library files change. Note, they are different between Windows and Linux.

- Based on how SLURM classifies "steps" and "jobs" you probably will need to hold your manager node in a separate job then the worker nodes. Start an interactive session with one slot for your manager process. Before launching Julia, use the *salloc* command to setup a job for all your workers. At this point you can start Julia for your manager process and have it spawn the worker processes from within Julia.

Bibliography

[1] Abi| goes-r series. https://www.goes-r.gov/spacesegment/abi.html. Accessed 04/13/2022.

[2] Climate data record (cdr) program, climate algorithm theoretical basis document, avhrr land bundle - surface reflectance and normalized difference vegetation index. https://www.ncei.noaa.gov/pub/data/sds/cdr/CDRs/Normalized\%20Difference\%20Vegetation\%20Index/AlgorithmDescription_01B-20b.pdf. Accessed 04/13/2022.

[3] Find pareto front of multiple fitness functions using genetic algorithm - matlab gamultiobj. https://www.mathworks.com/help/gads/gamultiobj.html. Accessed 04/13/2022.

[4] Follow pearl. https://followpearl.mit.edu/. Accessed 04/13/2022.

[5] Global real-time ocean forecast system. https://polar.ncep.noaa.gov/global/. Accessed 04/13/2022.

[6] Home - mit supercloud. https://supercloud.mit.edu/publications-and-software. Accessed 04/13/2022.

[7] An introduction to python bytecode. https://opensource.com/article/18/4/introduction-python-bytecode. Accessed 04/13/2022.

[8] Julia micro-benchmarks. https://julialang.org/benchmarks/. Accessed 04/13/2022.

[9] Matlab - mathworks - matlab and simulink. https://www.mathworks.com/products/matlab.html.

[10] Matlab execution engine - matlab. https://www.mathworks.com/products/matlab/matlab-execution-engine.html. Accessed 04/13/2022.

[11] Multi-processing and distributed computing - the julia language. https://docs.julialang.org/en/v1/manual/distributed-computing/. Accessed 04/13/2022.

[12] Multi-processing and distributed computing - the julia language. https://docs.julialang.org/en/v1/manual/distributed-computing/. Accessed 04/13/2022.

[13] Performance tips - the julia languahe. https://docs.julialang.org/en/v1/manual/performance-tips/. Accessed 04/13/2022.

[14] pmatlab. https://www.mit.edu/~kepner/pMatlab/. Accessed 04/13/2022.

[15] The radiometric calibration network (radcalnet). https://ceos.org/home-2/wgcv-radcalnet/. Accessed 04/13/2022.

[16] Slurm workload manager - overview. https://slurm.schedmd.com/overview.html. Accessed 04/13/2022.

[17] Space-track.org. https://www.space-track.org/. Accessed 04/13/2022.

[18] Standardized astrodynamic algorithm library (saal). https://www.astrodynamicstandards.com/. Accessed 04/13/2022.

[19] Top-of-atmosphere reflectance calibration of satellite and airborne sensor systems using flare vicarious calibration network. https://www.labsphere.com/wp-content/uploads/2021/09/FLARE.pdf. Accessed 04/13/2022.

[20] Welcome to python.org. https://www.python.org/.

[21] Why should you use afspc astrodynamics standards. https://www.astrodynamicstandards.com/Resources/Why%20Should%20You%20Use%20AFSPC%20Astrodynamic%20Standards.pdf. Accessed 04/13/2022.

[22] Why we created julia. https://julialang.org/blog/2012/02/why-we-created-julia/. Accessed 04/13/2022.

[23] *Earth Science Reference Handbook: A Guide to NASA's Earth Science Program and Earth Observing Satellite Missions*. 2006.

[24] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. The r*-tree: An efficient and robust access method for points and rectangles. In *Proceedings of the 1990 ACM SIGMOD international conference on Management of data*, pages 322–331, 1990.

[25] Jeff Bezanson, Stefan Karpinski, Viral B Shah, and Alan Edelman. Julia: A fast dynamic language for technical computing. *arXiv preprint arXiv:1209.5145*, 2012.

[26] Tom Bloom. What's the best answer? it's survival of the fittest. *New York Times*, Aug 1990.

[27] Christopher Boshuizen, James Mason, Pete Klupar, and Shannon Spanhake. Results from the planet labs flock constellation. 2014.

[28] Alexandru Horia Brie and Philippe Morignot. Genetic planning using variable length chromosomes. In *ICAPS*, pages 320–329, 2005.

[29] Malachy Browne, David Botti, and Haley Willis. Satellite images show bodies lay in bucha for weeks, despite russian claims. *New York Times*, Apr 2022.

[30] Keki Burjorjee. The fundamental problem with the building block hypothesis. *arXiv preprint arXiv:0810.3356*, 2008.

[31] Eric Chassignet, Harley Hurlburt, E. Joseph Metzger, Ole Smedstad, James Cummings, George Halliwell, Rainer Bleck, Remy Baraille, Alan Wallcraft, Carlos Lozano, Hendrik Tolman, Ashwanth Srinivasan, Steve Hankin, Peter Cornillon, Robert Weisberg, Alexandra Barth, Ruoying He, Francisco Werner, and John Wilkin. Us godae: Global ocean prediction with the hybrid coordinate ocean model (hycom). *Oceanography (Washington, D.C.)*, 22(2):64–75, 2009.

[32] Luis Cruz-Piris, Ivan Marsa-Maestre, and Miguel A Lopez-Carmona. A variable-length chromosome genetic algorithm to solve a road traffic coordination multi-path problem. *IEEE Access*, 7:111968–111981, 2019.

[33] Indraneel Das and John E Dennis. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM journal on optimization*, 8(3):631–657, 1998.

[34] Olivier L. de Weck, Daniel Roos, and Christopher L. Magee. *Engineering Systems: Meeting Human Needs in a Complex Technological World*. The MIT Press, 10 2011.

[35] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and Tanaka Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In *International conference on parallel problem solving from nature*, pages 849–858. Springer, 2000.

[36] Edsger W Dijkstra et al. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[37] Matthew P Ferringer and David B Spencer. Satellite constellation design tradeoffs using multiple-objective evolutionary computation. *Journal of spacecraft and rockets*, 43(6):1404–1411, 2006.

[38] Carlos M Fonseca, Peter J Fleming, et al. Genetic algorithms for multiobjective optimization: Formulation discussion and generalization. In *Icga*, volume 93, pages 416–423. Citeseer, 1993.

[39] Alex Fraser, Donald Burnell, et al. Computer models in genetics. *Computer models in genetics.*, 1970.

[40] David E Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison Wesley, Boston, MA, January 1989.

[41] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., USA, 1st edition, 1989.

[42] Antonin Guttman. R-trees: A dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, pages 47–57, 1984.

[43] Maha Haji, Johannes Norheim, and Olivier L de Weck. A framework for the design of renewably powered offshore auv servicing platforms. In *Ocean Sciences Meeting 2020*. AGU, 2020.

[44] Maha N Haji, Jimmy Tran, Johannes Norheim, and Olivier L de Weck. Design and testing of auv docking modules for a renewably powered offshore auv servicing platform. In *International Conference on Offshore Mechanics and Arctic Engineering*, volume 84386, page V06BT06A024. American Society of Mechanical Engineers, 2020.

[45] I Harvey. The saga cross: The mechanics of crossover for variable-length genetic algorithms. In *Parallel Problem Solving from Nature, 2: Proceedings of the Second Conference on Parallel Problem Solving from Nature, Brussels, Belgium, 28-30 September, 1992*, pages 269–278, USA, 1992. Elsevier Science Inc.

[46] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

[47] Nozomi Hitomi and Daniel Selva. Constellation optimization using an evolutionary algorithm with a variable-length chromosome. In *2018 IEEE Aerospace Conference*, pages 1–12. IEEE, 2018.

[48] HL Hooper and JR Herrnstein. Active rendezvous between a low-earth orbit user spacecraft and the space transportation system (sts) shuttle. In *NASA, Goddard Space Flight Center, Flight Mechanics (Estimation Theory Symposium, 1989*, 1989.

[49] Benjamin Hutt and Kevin Warwick. Synapsing variable-length crossover: Meaningful crossover for variable-length genomes. *IEEE transactions on evolutionary computation*, 11(1):118–131, 2007.

[50] Stojce Dimov Ilcev. *Global Satellite Meteorological Observation (GSMO) Theory*. Springer International Publishing, 2018.

[51] DA Kaya, JA Peugh, and BC Bohannon. Afspc astrodynamic standards-the way of the future. Technical report, MASSACHUSETTS INST OF TECH LEXINGTON LINCOLN LAB, 2001.

[52] Il Yong Kim and Oliver L De Weck. Adaptive weighted-sum method for bi-objective optimization: Pareto front generation. *Structural and multidisciplinary optimization*, 29(2):149–158, 2005.

[53] Joshua D Knowles and David W Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary computation*, 8(2):149–172, 2000.

[54] Christopher Koettl. New satellite images show more russian forces massing on three sides of ukraine. *New York Times*, Feb 2022.

[55] Joshua B Kollat and Patrick M Reed. The value of online adaptive search: a performance comparison of nsgaii, ε-nsgaii and εmoea. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 386–398. Springer, 2005.

[56] Chris Lattner and Vikram Adve. Llvm: A compilation framework for lifelong program analysis & transformation. In *International Symposium on Code Generation and Optimization, 2004. CGO 2004.*, pages 75–86. IEEE, 2004.

[57] CY Lee and EK Antonsson. Variable length genomes for evolutionary algorithms. In *GECCO*, volume 2000, page 806, 2000.

[58] Jr. Legge, Robert S. *Optimization and valuation of recongurable satellite constellations under uncertainty*. PhD thesis, Massachusetts Institute of Technology, 2014.

[59] Seereeram Li, Ravichandran Mehra, Robert Smith, and Randal Beard. Multi-spacecraft trajectory optimization and control using genetic algorithm techniques. In *2000 IEEE Aerospace Conference. Proceedings (Cat. No. 00TH8484)*, volume 7, pages 99–108. IEEE, 2000.

[60] Joaquim RRA Martins, Peter Sturdza, and Juan J Alonso. The complex-step derivative approximation. *ACM Transactions on Mathematical Software (TOMS)*, 29(3):245–262, 2003.

[61] Ciara N. McGrath, Ruaridh A. Clark, Astrid Werkmeister, and Malcolm Macdonald. Small satellite operations planning for agile disaster response using graph theoretical techniques. In *70th International Astronautical Congress*, USA, October 2019. The paper was originally presented at the 70th International Astronautical Congress, 21-25 October 2019, Washington, D.C.

[62] Brad L Miller, David E Goldberg, et al. Genetic algorithms, tournament selection, and the effects of noise. *Complex systems*, 9(3):193–212, 1995.

[63] Julian F Miller and Stephen L Smith. Redundancy and computational efficiency in cartesian genetic programming. *IEEE Transactions on Evolutionary Computation*, 10(2):167–174, 2006.

[64] Sarah J Morgan. Reconfigurable satellite constellations for mobile target tracking. Master's thesis, Massachusetts Institute of Technology, 2021.

[65] Daniele Mortari and Matthew P Wilkins. Flower constellation set theory. part i: Compatibility and phasing. *IEEE Transactions on Aerospace and Electronic Systems*, 44(3):953–962, 2008.

[66] Jianjun Ni, Kang Wang, Haohao Huang, Liuying Wu, and Chengming Luo. Robot path planning based on an improved genetic algorithm with variable length chromosome. In *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pages 145–149, 2016.

[67] Una-May O'Reilly and Franz Oppacher. The troubling aspects of a building block hypothesis for genetic programming. In *Foundations of genetic algorithms*, volume 3, pages 73–88. Elsevier, 1995.

[68] Joseph D Ortiz, Dulci M Avouris, Stephen J Schiller, Jeffrey C Luvall, John D Lekki, Roger P Tokars, Robert C Anderson, Robert Shuchman, Michael Sayers, and Richard Becker. Evaluating visible derivative spectroscopy by varimax-rotated, principal component analysis of aerial hyperspectral images from the western basin of lake erie. *Journal of great lakes research*, 45(3):522–535, 2019.

[69] Joseph D Ortiz, Dulcinea Avouris, Stephen Schiller, Jeffrey C Luvall, John D Lekki, Roger P Tokars, Robert C Anderson, Robert Shuchman, Michael Sayers, and Richard Becker. Intercomparison of approaches to the empirical line method for vicarious hyperspectral reflectance calibration. *Frontiers in Marine Science*, 4:296, 2017.

[70] Nikolaos K. Pavlis, Simon A. Holmes, Steve C. Kenyon, and John K. Factor. The development and evaluation of the earth gravitational model 2008 (egm2008). *Journal of Geophysical Research: Solid Earth*, 117(B4), 2012.

[71] C Radhakrishna Rao. Information and the accuracy attainable in the estimation of statistical parameters. *Reson. J. Sci. Educ*, 20:78–90, 1945.

[72] Stephen J Schiller and John Silny. The specular array radiometric calibration (sparc) method: a new approach for absolute vicarious calibration in the solar reflective spectrum. In *Remote Sensing System Engineering III*, volume 7813, page 78130E. International Society for Optics and Photonics, 2010.

[73] J. Silny and S. Schiller. Polarimetric calibration of a remote sensor. US9372119, United States Patent and Trademark Office, 21 June 2016.

[74] Alexandra N. Straub. Expanded tradespace analysis and operational considerations for reconfigurable satellite constellations. Master's thesis, Massachusetts Institutue of Technology, 2020.

[75] David Vallado. A summary of the aiaa astrodynamic standards effort. *Advances in the Astronautical Sciences*, 109:1849–1872, 01 2002.

[76] Haley Willis and Sarah Kerr. A new satellite image shows food lines in mariupol, as russian forces push deeper into the city. *New York Times*, Mar 2022.

[77] Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, 3(4):257–271, 1999.