LA-UR-21-31784

Title: Vehicle Tracking

Author(s):
Scott, John Mitchell
Dallmann, Nicholas
Durham, J. Matthew
Junor, William
Malone, Michael
Thornton, Remington Tyler
Tierney, Thomas Edward IV
Carlsten, Bruce Eric
Hoffmann, Mitchell Poole
Zuboraj, MD Rashedul Alam

Intended for: Report

Issued: 2021-12-02

Vehicle Tracking

LDRD-20200664DI

Bruce Carlsten, Nicholas Dallmann, Matt Durham, Mitch Hoffmann, Bill Junor, Michael Malone, John Scott, Tyler Thornton, Tom Tierney, MD Zuboraj

LA-UR-21-xxxxx
30 SEP 2021

Los Alamos
NATIONAL LABORATORY

Contents

Figures

Contents

Tables

Acknowledgements

The authors would like to thank the Los Alamos Laboratory Directed Research and Development Program for its support of this effort through its Director's Initiative portfolio of funding. The project benefitted heavily from our interactions with reviewers including Tess Light, Steve Judd, David Cremer, Kip Bishofberger and Steve Ellis.

1 Introduction

The detection and tracking of vehicles moving at hypersonic speeds (> Mach 5) at altitudes of ~100 km has been of growing interest due to the ongoing development of hypersonic glide vehicles (HGVs) by the U.S., Russia and China [1]. This challenge is notably different from traditional ballistic missile systems whose re-entry vehicles move at hypersonic velocities but at much higher altitudes on a ballistic course. Hypersonic glide vehicles are also designed to be able to make sudden turns along their flight path making prediction of their trajectories difficult. Their ability to maneuver, combined with their high velocity and altitude make them difficult to track. The combination of these characteristics presents a challenge for anyone attempting to defend against a hypersonic glide vehicle attack by generally shortening the time available to react. That reaction can involve moving potential targets or attempting to detect the hypersonic glide vehicle and intercept it before it reaches the intended target. Figure 1 compares the trajectory of hypersonic vehicles compared to that of a traditional ballistic missile.

The flight altitude of hypersonic glide vehicles (~40-100 km) makes them more difficult to detect early in their flight with over-the-horizon radar as compared to traditional ballistic missile systems. Their lower altitude means that radar looking over the horizon will see them when they are closer to the radar shortening the time for response. Their maneuverability also means that they can attempt to avoid detection by skirting along the edge of the range of statically positioned radar installations. The challenges of their detection and tracking are noted in [2].

A potential solution to this difficult tracking problem is to attempt to detect HGVs from above instead of from the ground or at sea. A space-based sensing technology could be an appropriate solution to this detection problem. The problem is determining what kind of space-based system one would use. If one were to deploy sensors on satellites this raises a number of questions. What would the sensing technology be? How many satellites would you have to deploy with this technology How big would the satellites be?

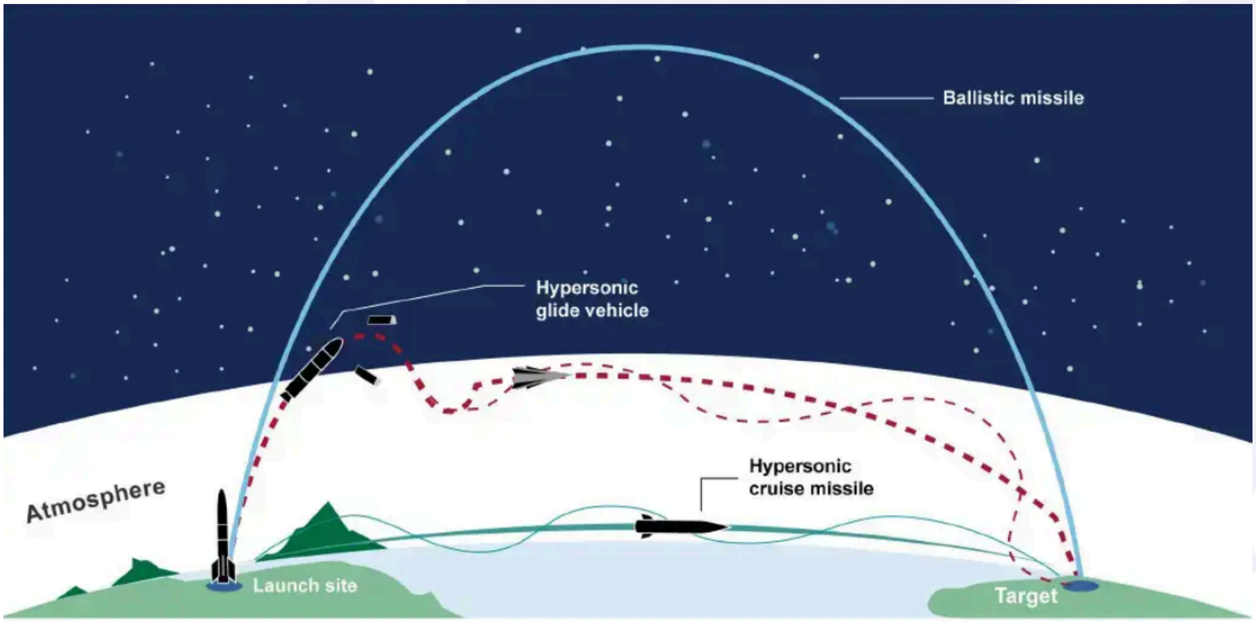


Figure 1. A comparison of flight profiles from hypersonic glide vehicles and hypersonic cruise missiles from [1].

Introduction

LANL has a history of deploying satellite sensing technologies dating back to the 1960s with Project Vela [3]. However, before deploying any kind of solution, one needs to determine what the potential is for that system to be able to perform the function one intends. Consequently, the ability to have a tool that could help one design a sensing system, in this case, a satellite-based system would be useful. For the problem of detecting and tracking HGVs, a potential solution would be a constellation of satellites deployed in low earth orbit (the low earth orbit region is defined as <2000 km in altitude) using radar as the sensing technology. However, the questions asked above about how many satellites and how big are important as that will drive the overall complexity and cost of such a system. One wants to be able to design a system that is feasible at tracking but not so costly that it cannot be built.

Given this challenge, the work presented in this report summarizes the development of an analytical tool that can help one determine design parameters for a constellation of satellites (expected to be in low earth orbit) using radar-sensing technology to track vehicles moving at altitudes relevant to HGVs. We begin by developing a capability to describe the relative positions of sensors and targets to track. This is followed by a discussion of the radar cross section of potential targets and how one can detect the targets. For targets, we use previously published HGV geometries. The angle of view of a target and the relative position, velocity and acceleration of the source, detector and target of the radar signal are obviously important parameters to understand for such a system. For signal processing, understanding what impacts the signal-to-noise ratio is key to understanding how much power one will need for a satellite given assumptions about radar frequency band and transmitting and receiving antenna properties. Lastly, one needs to have a method to track the vehicle given the detection by radar. The geolocation of a target will have an uncertainty driven by the signal quality and how many views one has of the target given the number of satellites within range of the target. The final section on geolocation ends with an example of a given vehicle trajectory scenario being tracked with different satellite configurations.

1.1 References

[1] Tingley and Trevithick, “Missile Defense Agency Lays Out How it Plans to Defend Against Hypersonic Threats.” June 19, 2021. https://www.thedrive.com/the-war-zone/41164/missile-defense-agency-lays-out-how-it-plans-to-defend-against-hypersonic-threats.

[2] Watts, Trotti, Massa. “Primer on Hypersonic Weapons in the Indo-Pacific Region.” August, 2020. Atlantic Council Scowcroft Center for Strategy and Security. ISBN-13:978-1-61977-111-6.

[3] Grant, Virginia. “Cold War Watchmen.” Natioal Seucrity Science Magazine, Los Alamos National Laboratory, Summer 2020 issue, pg 44.

2 Geometric and Signal Simulation

The goal of the geometric simulation piece of the project is to simulate *representative* scenarios of sensor-target relative positions. Notably, we make no attempt to certify that we can predict the exact measurements that would arise from a given constellation for a given flight, but merely that we can replicate the type, frequency, and robustness of such measurements in order to determine the efficacy of the proposed system. This relaxed requirement on position accuracy combined with our desire to evaluate sweeps of many constellation design variables led us away from high-precision commercial tools such as STK. Instead, we have created an integrated Python package optimized for speed and flexibility, which is capable of efficiently simulating large sensor constellations along with a few types of target flight profiles. Careful architecture allows us to build large constellations up from a single orbit and a number of rotations while allowing queries of the entire constellation over many time steps to be calculated by pre-compiled machine code—thus sidestepping the speed issues associated with interpretive programming while retaining its flexibility.

All vector outputs are given in Earth-centered inertial (ECI) coordinates. This system is centered at Earth's center with the Z axis pointing geographic North and the X and Y axes completing a right-handed set intersecting the equator. Timing of the simulation is aligned such that the X axis intersects the prime meridian at t=0. For internal conversion between Earth-fixed and ECI coordinates, we assume a spherical Earth and constant rotation rate. Base units are kilometers and seconds.

2.1 Orbit Simulation

Each type of vehicle we expect to simulate has its own class in the simulation framework. By far the most used is the conic orbit class, which simulates the trajectory of the RADAR satellites. This is constructed in two steps.

First, we calculate the position, $\overrightarrow{R_{2D}}$, and velocity, $\overrightarrow{V_{2D}}$, in the 2-D orbit frame as depicted in Figure 2. Position and velocity in this frame are simple functions of the orbit semimajor axis, a, eccentricity, e, and true anomaly, θ. All but θ of these parameters are constants, and θ is calculated directly from the time. Unfortunately, the calculation of True Anomaly from time involves solving the Kepler equation (equation 1) for the eccentric anomaly, E, which must be done numerically for $e \neq 0$. To avoid the costly task of solving this for every satellite at every time step, the entire solution space is pre-solved and the solution is interpolated when needed. Fortunately, we are interested mostly in circular orbits where $e = 0$ and this step is not necessary.

$$M = E - e \sin E \tag{1}$$

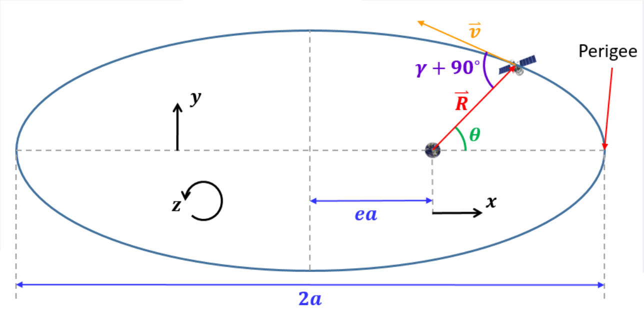


Figure 2. 2-D conic orbit

Then, we reorient the position and velocity vectors into the ECI system using the inclination, i, argument of perigee, ω, and ascending node, Ω. The set $\{\omega, i, \Omega\}$ constitutes a 3-1-3 extrinsic Euler sequence, from which we generate the *constant* rotation matrix, $[L]$. To orient the position and velocity in 3D space, we simply multiply by $[L]$.

$$\vec{R} = [L]\overrightarrow{R_{2D}}\,,\ \vec{V} = [L]\overrightarrow{V_{2D}} \qquad (2)$$

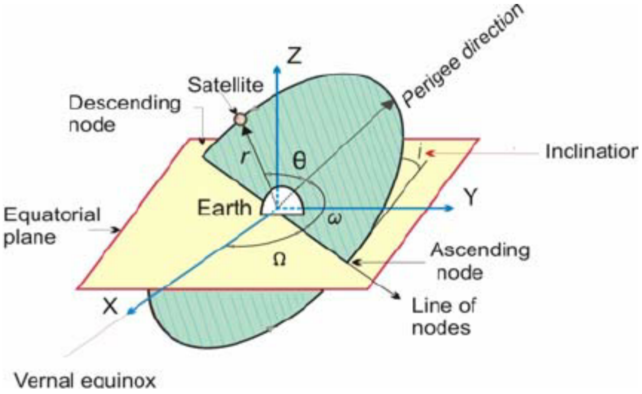


Figure 3. 3-D orbit orientation using Inclination, perigee argument, and ascending node.

2.2 Target Simulation

We created a few options for target simulation. So far, the most used has been our simple case of a great-circle route. The target is given a starting and ending point in latitude-longitude-altitude coordinates and time, and then is simulated as flying along a perfect circular route from start to end at constant speed. This over-simplified solution is useful as a baseline case. We can say with relative certainty that any solution which cannot track this target cannot track any target of interest.

The next target simulator is designed for flexibility, and allows for realistic flight profiles. By stitching several great circle segments end to end, it creates a composite flight path which may be composed of complicated maneuvers. For vertical motion, the simulator receives an arbitrary function which simulates arbitrary physical constraints and control inputs. As an example, we generated a trajectory with a profile function designed to follow a more realistic HGV trajectory. Using a proportional control system, this function simply pitched up or down to keep the trajectory aligned with the example as shown in Figure 4.

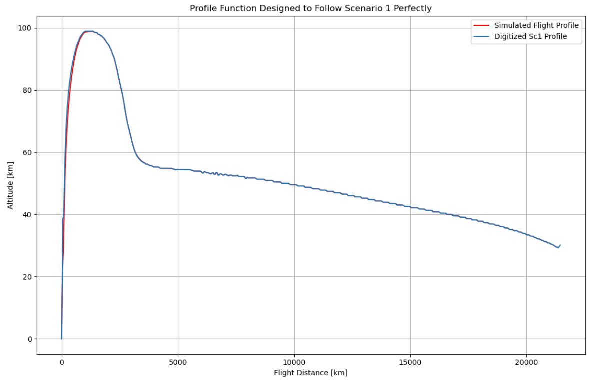


Figure 4. Profile function following an example HGV flight path

We believe that given more time and effort, this simulation could be extended to provide insight into flight vehicle design using flight test data. Incorporating realistic physics, we would allow unknown design parameters, such as force and moment coefficients, to vary. Then, a gradient-descent algorithm would iteratively simulate the trajectory until a parameter set is found that causes the simulation to best represent of the flight data. As an example of this process Figure 5 shows the use of this approach to determine the approximate energy delivery of a launch vehicle. The profile function follows the recorded launch and applies a constant thrust, then varies thrust to weight ratio and physically simulates the free-fall phase in order to determine the correct energy delivery by the Launch Vehicle.

Finally, it is also necessary to simulate the target's attitude in order to determine the RADAR cross section. In the great circle case, we simply align the vehicle x axis with its velocity, and the z axis with its position. In the general case, the attitude is fully described by three angles which are closely related to the trajectory—The angles of attack, sideslip, and roll. By making the assumption of coordinated flight—the acceleration vector is perpendicular to the wing span—we fix the sideslip angle to zero, and the roll angle centers the acceleration vector in the vehicle coordinate system. We then use an aerodynamic parameter—the lift curve slope—to determine the angle of attack. In both cases, the attitude is stored as a 3x3 rotation matrix. Multiplication of the unit vector pointing to the satellite by this matrix yields the view vector in the target's frame of reference, which determines the RCS.

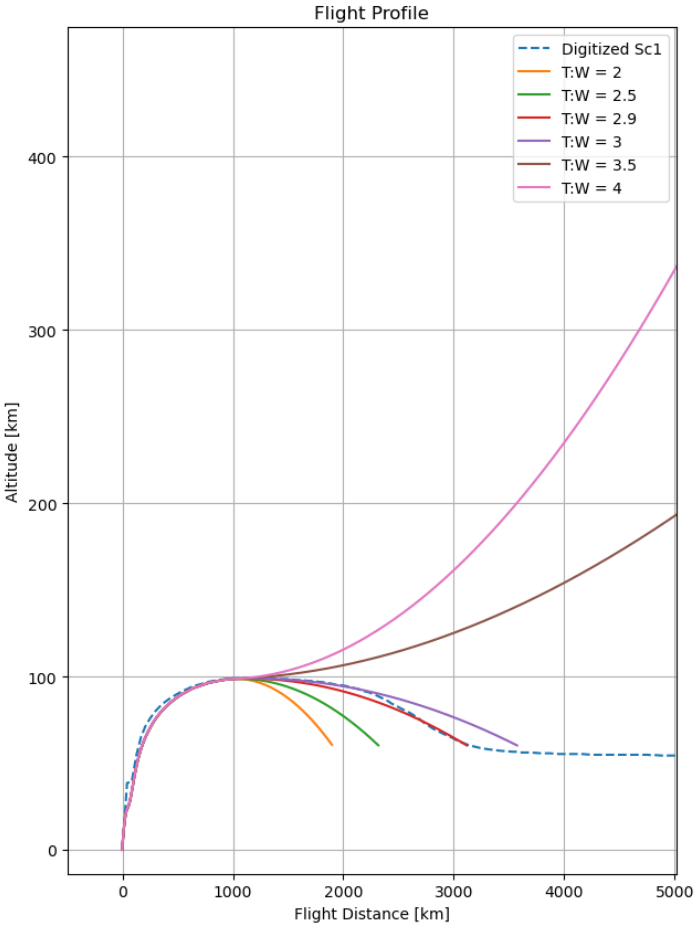


Figure 5. Optimization of LV thrust-to-weight ratio

2.3 Signal Simulation

Before we can evaluate the tracking capability of a given system, we must translate the geometric simulation, signal processing, and RADAR cross section analyses into a realistic list of measurements. First, we simulate the whole scenario to 1 second resolution and identify times when each satellite or pair of satellites is within a pre-determined range. This gives us a list of time intervals with corresponding satellite indices to search, which we refer to as links. Second, we translate those links into a list of measurements. Fundamentally, this process consists of an integration of the well-known RADAR power equation modified for the bi-static case:

$$P_r = \frac{P_t G_t G_r \lambda^2 \sigma}{(4\pi)^3 R_t^2 R_r^2} \qquad (3)$$

The transmit power and wavelength, and the gains of the transmitter and receiver antennas (P_t, λ, G_t, and G_r) are input parameters affecting the design of the constellation and individual spacecraft which we may vary with the goal of optimizing our system. We determine the remaining variables at each instant in time using the RCS and geometric simulation outputs. The required signal integration time, and thus the interval timing of the measurements, is the time required to acquire enough signal energy such that the SNR is sufficient for target identification. Thus, integration time is also a function of the instantaneous ranges and vehicle attitude. For slowly varying P_r, the integration time is determined simply by equation 4.

$$t_i = \frac{K_b T}{P_r} SNR_{Target} \qquad (4)$$

However, recall that at reasonable signal power and gain levels, integration on the order of 2 s may be required. Over this time, with sensors moving at 7 km/s tracking a target with a varying cross section, the received power level may vary widely. Our integration time then becomes the minimum t_i that satisfies equation 5. (Note: there are other terms in this relation that are dependent on our handling of acceleration effects)

$$\int_0^{t_i} P_r \geq SNR_{target} K_b T \qquad (5)$$

To determine the measurement times, we set up an initial value problem which keeps track of the total received signal energy over the course of each link as well as the total integrated acceleration error. The IVP solver calls a derivative function which incorporates the physics and geometry of the problem. The derivative function calculates the continuous position and velocity of the satellites involved in the link, and the position, velocity and orientation of the target. From these values it determines the angle – dependent RCS, received power, and acceleration error. We also built In a side-effect to this function that creates a list of measurement values every time it is called in a state that allows a new measurement. Measurements are then fed into any live tracking algorithms that inform the acceleration estimation, or saved for use in the full tracking solution.

2.4 Tracking Parameters

The product of the signal processing module on any given satellite is the signal delay and Doppler shift. From these we determine the range to target, R, and its rate of change, $\dot{R}$. An understanding of the effect of scene geometry on these variables informs the signal processing to extract the maximum available information from the signal. Noticing that the bi-static values for R, and $\dot{R}$ are simply the sum of those mono-static quantities for the two satellites involved, we can characterize the effect of the vector positions, velocities, and accelerations on the mono-static case only to gain insight on all of the signals of interest. First, the relative range is defined by (equation 6).

$$R = |\vec{R}_t - \vec{R}_{sat}| \tag{6}$$

From here on, we will drop the subscripts, set the satellite location as the origin, and represent the vector velocity and acceleration by their initials such that $\vec{R}_t - \vec{R}_{sat} = \vec{R}$, $\dot{\vec{R}}_t - \dot{\vec{R}}_{sat} = \vec{V}$, and $\ddot{\vec{R}}_t - \ddot{\vec{R}}_{sat} = \vec{A}$. We show range and its derivatives in equation set 7

$$\begin{gathered} R = \left|\vec{R}\right| \\ \dot{R} = \frac{\vec{R} \cdot \vec{V}}{R} \\ \ddot{R} = \frac{\vec{V} \cdot \vec{V} + \vec{R} \cdot \vec{A}}{R} - \frac{\left(\vec{R} \cdot \vec{V}\right)^2}{R^3} \end{gathered} \tag{7}$$

A geometric understanding of the range derivatives aids our intuition on the signal effects. $\dot{R}$ is easily understood as the projection of velocity onto the position vector as shown in Figure 6

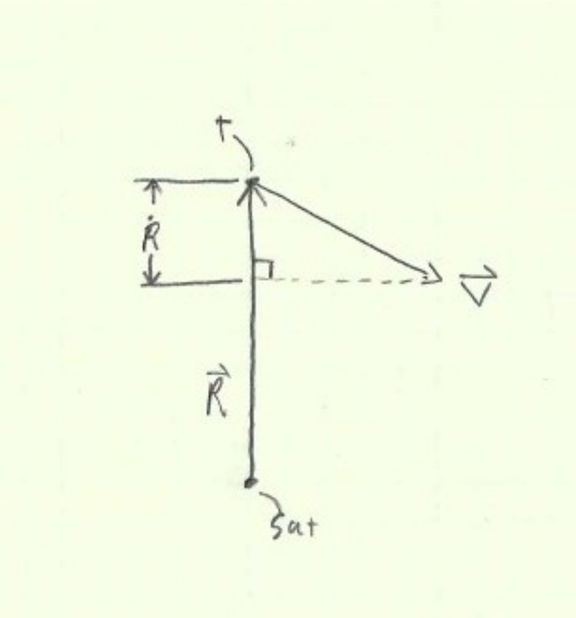


Figure 6. Geometric Interpretation of $\dot{R}$

Geometric representation of $\ddot{R}$ is more complicated and benefits from re-forming the equation to extract some more familiar quantities. The first terms in equations 8-10 are familiar as the projection of $\vec{A}$ onto $\vec{R}$ as before, but the second term merits more analysis. Noticing from the Pythagorean Theorem and from Figure 2 that second term of equation 8 is the square of the perpendicular component of $\vec{V}$ to $\vec{R}$ (Figure 7), we recognize this term as being related to the instantaneous rotation rate of $\vec{R}$. By expressing it as a cross product (9) and substituting $\vec{\omega}_{\vec{R}} = \frac{\vec{R}\times\vec{V}}{R^2}$ (10), we reveal the second term as the *centrifugal acceleration* due to the rotation of the position vector.

$$\ddot{R} = \frac{\vec{R} \cdot \vec{A}}{R} + \frac{1}{R}\left(V^2 - \dot{R}^2\right) \tag{8}$$

$$\ddot{R} = \frac{\vec{R} \cdot \vec{A}}{R} + \left| \frac{\vec{R} \times \vec{V}}{R^2} \times \vec{V} \right| \tag{9}$$

$$\ddot{R} = \frac{\vec{R} \cdot \vec{A}}{R} + \left| \vec{\omega}_{\vec{R}} \times \vec{V} \right| \tag{10}$$

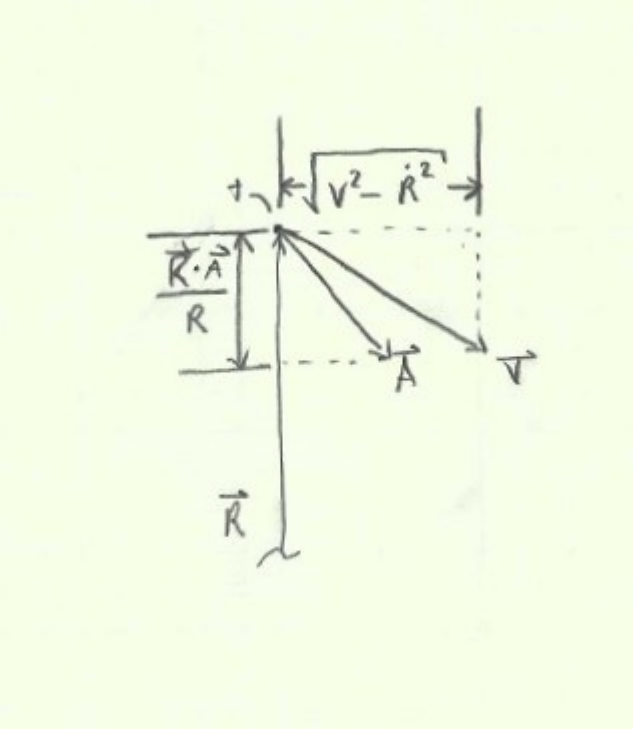


Figure 7: Geometric Interpretation of $\ddot{R}$

Any unknown portion of $\ddot{R}$ that is not accounted for when integrating the signal causes smearing of the ambiguity function as described in the signal-to-noise section. So our objective is to estimate this quantity with as much accuracy as possible when performing the integration. If we assume the receiver has no knowledge of the current target position and velocity, we must define $\ddot{R}_{est}$ solely with respect to the coordinates of the ambiguity function. We cannot predict the direction of acceleration with any certainty, so we are limited to making an educated guess about the centrifugal acceleration and correcting only for that. Since the direction of the target velocity is uniformly distributed, the average satellite-relative speed is equal to the speed of the satellite. Therefore, in the case where the receiver has no prior information regarding target position and velocity, the best estimate of $\ddot{R}$ is given by equation 11.

$$\ddot{R}_{est}\left(R, \dot{R}\right) = \frac{v_c^2 - \dot{R}^2}{R} \tag{11}$$

Of course, it is not always true that the receiver has no information on the position and velocity of the target. We assume the system will be given some sort of “tip-off” fix at which to begin a search, and after initial acquisition, each receiver will have some amount of data regarding the most recent fixes. If the satellites are allowed live cross-links, receivers may even have access to the entire live tracking solution. In either case, the estimate of centrifugal acceleration is created using the relative velocity of the target at the most recent fix.

The possible need for $\ddot{R}$ correction thus requires us to evaluate the following 4 scenarios of system architecture. In increasing order of complexity:

1. No live tracking, no estimation of $\ddot{R}$
2. No live tracking, $\ddot{R}_{est}$ is calculated as in equation 11.
3. Each satellite keeps its own tracking record, using only links it was involved in.
4. The whole constellation is live cross-linked giving each receiver knowledge of the full solution.

The impact of geometric acceleration in terms of system requirements is outlined in the signal-to-noise section of this report

3 Radar Cross Section

3.1 Introduction

RCS refers to ‘radar cross section’ of an object. This is an important parameter to calculate scattered power coming from a target when the target is illuminated by a plane wave at the far-field range. The RCS parameter is usually denoted as σ and defined as [[4]:

$$\sigma = \lim_{r\to\infty} 4\pi r^2 \frac{|E^s|^2}{|E^i|^2} \quad 12$$

Where, E^i refer to incident field on the object and E^s refer to scattered field coming from the object. From the definition of RCS, it is obvious that σ is a power parameter and expressed as area (m^2). Usually, it is expressed in dBSm or dB ($\sigma_{dbsm} = \sigma_{db} = 10\log_{10}\sigma$) as a ratio to $1m^2$ cross-section area of an object.

RCS, in general quantifies the visibility of the target. The better the scattered signal from a target, the larger the RCS is. Since it provides an estimation of target area in the far-field, it is an important element to calculate signal-to-noise ratio (SNR) for receiver systems, contributing to link-budgeting. The classic Radar Range equation is [5]:

$$P_r = P_t \frac{G_t G_r \lambda^2 \boldsymbol{\sigma}}{(4\pi)^3 R_t^2 R_r^2} \quad 13$$

RCS can be monostatic, bi-static or multi-static, depending on the location of transmitter (Tx) and receiver (Rx) , as shown in Fig. 1. When $\alpha \neq 0$, the system is at-least a bistatic system, separating Tx and Rx. However, when $\alpha = 0$, the system becomes mono-static, merging receiver and transmitter location into a single line.

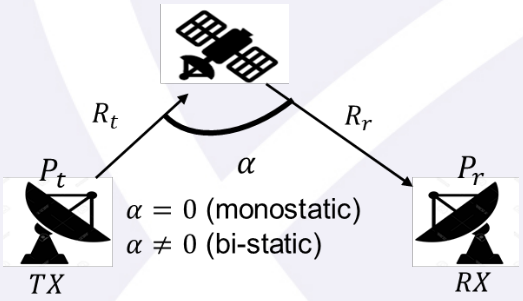


Figure 8. Bi-static radar system for $\alpha \neq 0$, separating transmitter and receiver. The system converges to mono-static when $\alpha = 0$. If multiple transmitters (Tx) and receivers (Rx) are present, the system becomes multi-static.

RCS of a target is usually measured in far-fields. Therefore, it location parameters are chosen as θ and ϕ, in spherical co-ordinates, where the target is assumed to be at the center of the sphere. For a bi-static case, σ depends on several parameters:

1. Location of the target: $\theta_t, \phi_t, \theta_r, \phi_r$, where the subscripts ‘$t$’ and ‘$r$’ refers to ‘transmitter’ and ‘receiver’ respectively

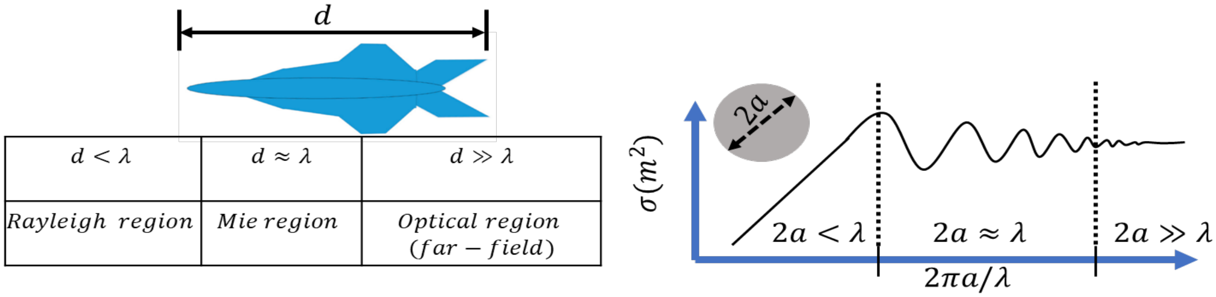
2. Scattering processes such as reflection, refraction, diffraction, creeping-waves etc.
3. Material content of the target: metallic, dielectric, absorbing materials, plasma etc.

In this project, the main goal is to model RCS parameter for Hypersonic Glide Vehicle (HGV), which are assumed to be metallic. The idea is to model multi-static RCS parameters using bi-static analysis of RCS in various locations and develop a multi-static software suite for accurate prediction of the target. We divided the RCS analysis in two broad sections: 1) First, we will discuss briefly about frequency dependence, different scattering processes and associated numerical methods in analyzing RCS. Second, we will summarize the RCS calculation adopted for HGV model and post processing of polarization dependent RCS data for any arbitrary pair of transmitter (Tx) and receiver (Rx).

3.2 Brief Discussion on RCS Scattering

3.2.1 *Frequency dependence*

The scattered field coming from an object is heavily dependent on the frequency of the incoming signal. The scattered power depends mostly on the ratio (d/λ) [6], where d refers to largest dimension of the object and λ depends on the wavelength of the incident signal. This is shown in Fig. 2.



$d < \lambda$	$d \approx \lambda$	$d \gg \lambda$
Rayleigh region	*Mie region*	*Optical region* *(far − field)*

(a)

(b)

Figure 9. (a) Wavelength (Frequency) dependence in RCS. Three region is shown as a function of d/λ. (b) Wavelength dependence of RCS of sphere is shown. In physical optics, $\sigma = \pi a^2$ for optical region, i.e. a constant. For Rayleigh and Mie region, this is not true.

As seen in Figure 9, RCS of a sphere is a function of wavelength (frequency) for ratio $(\frac{2a}{\lambda} \leq 1)$. In Rayleigh region, σ varies linearly with wavelength until it hits Mie (resonance) region where σ fluctuates (resonates) a lot. In Mie region, which can be true for $1 \leq \frac{d}{\lambda} \leq 10$, σ varies significantly, making it very unreliable for analysis. Typically, for any object, the far-field RCS (optical-region) value is independent of frequency. For our analysis, we chose 6.9GHz as our frequency of interests, in optical-region.

3.2.2 *Scattering Processes*

Signals scattered from a target may be represented by a power quantity, however, the received signal may be consequence of several complex physical processes. This can be shown in Figure 10. We will summarize briefly several scattering processes commonly seen and modeled in RCS analysis:

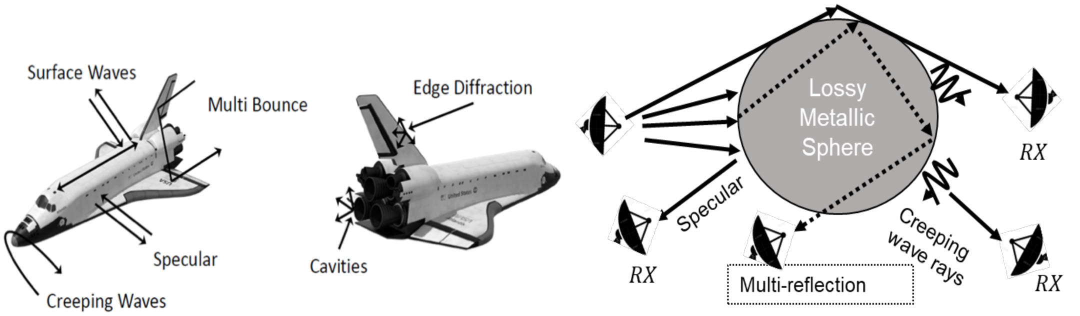


Figure 10. (left) Scattering of signal shown in typical aircrafts (Courtesy: NASA) [7]. (Right) Different scattering processes are shown in cartoon for a large sphere. The creeping wave contributes to signal received by a receiver located in the shadow region (right).

1. Specular Reflection: This is regular reflection from a surface. It follows typical law of optical reflection. This process contributes to the significant portion of scattered power coming from a target.
2. Diffraction: Diffraction is the second large contributor to scattered signal. Diffraction is very complicated processes and difficult to analyze. Typically, when there is an edge or any end of boundary encountered, field becomes infinity at sharp edges/corners and diffraction is observed (edge/cone diffraction). Also, there can be diffracted signal in shadow region if a smooth surface with minimal roughness is encountered (creeping wave/ smooth diffraction). We observed diffracted signal in shadow region for smooth objects such as sphere, cylinder etc. in numerical analysis.
3. Surface Waves: Whenever there is a discontinuity in the constituent material of the target, an induced current (surface wave) is introduced which radiates incoherently at far-field. This surface wave contribution is mostly noise-like and does not contribute significantly in the signal.
4. Multi-reflection from target: If the target is made of composite materials that contain dielectrics or other transparent materials, incident wave can penetrate through the surface and then leak to other part so of the body, radiating in space. This is multi-bounce reflection. Often, cavity features as shown in Fig. 3 can also contribute to multi-bounce signal and energy fading. These are greatly handled numerically by Shooting and Bouncing Rays (SBR) method. We will explain that later.

3.2.3 *Numerical Methods*

RCS analysis is purely numerical in a sense that except some regular geometrical shape, there is no concrete analytic formula to determine RCS for arbitrary-shaped object. In electromagnetic analysis perspective, there are two ways to analyze RCS: 1) Exact method 2) Approximate method [8]. In Exact method, RCS of a target can accurately be simulated by solving Maxwell's equations in the domain of interest and then determine scattered fields. This is the most accurate way but may not be practical due to restriction of computational time, resource and cost. Another method is approximate method that analyzes RCS at reasonable accuracy with limited resources. While approximate methods are not necessarily 100% accurate, they can provide a good trade-off between level of accuracy and computational labor. Below, we will go through brief description of some common methods we went through our analysis.

1. Finite-Difference Time-Domain (FDTD): This method solves Maxwell's equation in differential form, by discretizing the equation in time-domain and solve those using appropriate boundary conditions [8]. FDTD is considerably faster method and calculates RCS for any composite type materials by solving exact fields inside the domain.
2. Method of Moments (MoM): This method is very accurate and popular for Perfect Electric Conductor (PEC) type material. In this method, Maxwell's equations in integral form are used to determine exact currents. This is usually done by discretizing the surface of the object into small-size ($\lambda/10$) patches. It uses basis functions to expand these currents and uses linear algebra to determine exact currents [4,[8]. These currents are then used in radiation integral to determine the radiated scattered field and RCS.
3. Physical Optics (PO): In this method, the object is discretized to tangent planes, which do not necessarily coincide with body material. It solves Stratton-Chu integral equation to solve scattered field in far-field region [8]. It is an approximate method that works very well for planar geometries, but fails to predict RCS accurately for curved and complex features.
4. Geometric Optics (GO): It is an approximate method that utilizes geometric ray tracing to find reflection from the target. It approximates vanishing current except from the specular point. This method falls off for flat surfaces and infinitely long radius of curvature.
5. Shooting and Bouncing Ray Method (SBR): It is a ray tracing method where the scattered field determined from an open scatterer. Rays that mimic incident plane wave are impinged upon the target and then traced from the target to an exit aperture [8[10]. This method utilized both GO and PO method. Electric field is traced within the rays with Geometric Optic (GO) rules. This is an approximate method that can simulate electrically large object with greater specular reflection [[8[10].

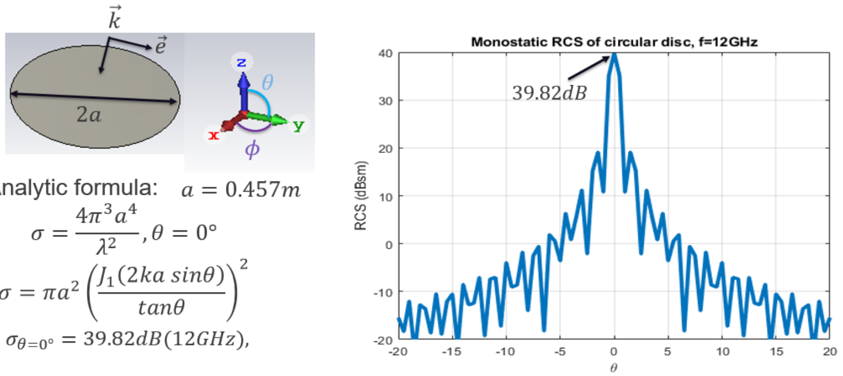
3.2.4 *Our Choice of Analysis*

In our study, we assumed our HGV model as PEC based. Therefore, MoM is the best method for obtaining exact solutions [4]. We used CST Microwave Studio (CSTMWS) [11] as our electromagnetic/RF simulation suite for analyzing RCS numerically. In this suite, a hybrid of MoM method known as 'Multilevel Fast Multi-Pole Method' (MLFMM) is used to simulate exact RCS analysis for metallic objects [9] . The solver that used this method is called ***Integral Equation (IE)*** solver. MLMFMM differs from MoM in that it groups the basic functions and calculates the interaction between these groups. Through this modification, this approach can handle electrically large problems with dramatically faster results than MoM. We used this method to get an exact solution of our problem for certain set of angles. This method was used to obtain a benchmark data for our analysis. Since our HGV model is electrically very large, we adopted SBR method in CSTMWS to simulate mostly specular reflection from the target. Below, we will provide details on our numerical analysis of HGV RCS modeling.

3.3 Numerical Analysis of RCS for HGV Model

3.3.1 *Validation of Simulation Process*

We started our numerical analysis by validating our simulation first using IE solver. This was done by comparing our results to analytic formula provided by physical optics (PO) approximation. We used simple shapes such as spheres, circular disc and flat plate to validate our simulation. The numerical results are given in Figures Figure 11-Figure 13.We mostly compared monostatic RCS data since most of the analytic results are provided in Monostatic form.



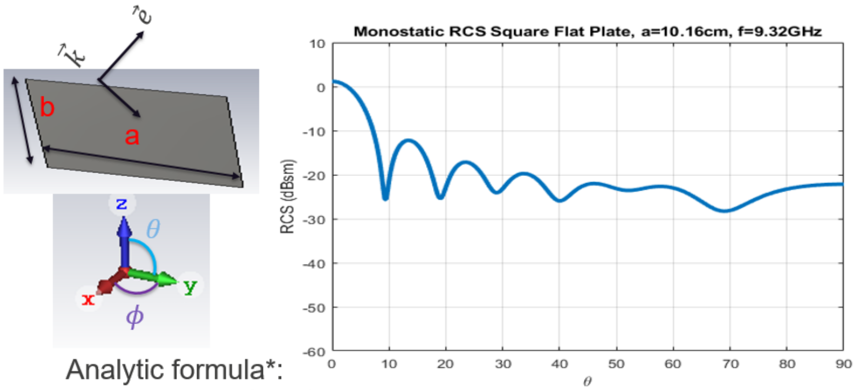
Analytic formula: $a = 0.457m$

$$\sigma = \frac{4\pi^3 a^4}{\lambda^2}, \theta = 0°$$

$$\sigma = \pi a^2 \left(\frac{J_1(2ka\, sin\theta)}{tan\theta}\right)^2$$

$$\sigma_{\theta=0°} = 39.82dB(12GHz),$$

Figure 11. RCS simulation data from IE solver for circular disc of radius 45.7cm. Analytic formula is also given to compare the result. $\vec{k}$ and $\vec{e}$ denotes direction of propagation and electric field respectively [4].

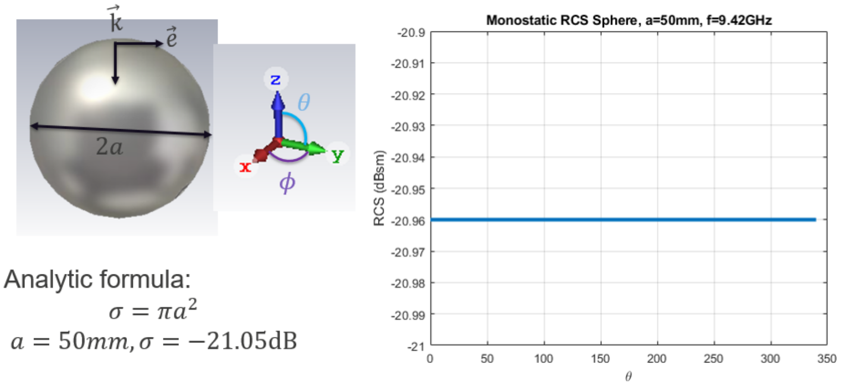


Analytic formula*:

$$\sigma = \frac{4\pi(ab)^2\cos^2\theta}{\lambda^2}\left(\frac{\sin(ak\sin\theta\cos\phi)}{ak\sin\theta\cos\phi}\,\frac{\sin(bk\sin\theta\sin\phi)}{bk\sin\theta\sin\phi}\right)$$

$$a = b = 10.16cm, \sigma_{\theta=0°} = 1.1137\text{dB}$$

Figure 12. RCS simulation data from IE solver for flat plate: a = b =10.16cm. Analytic formula is also given to compare the result. $\vec{k}$ and $\vec{e}$ denotes direction of propagation and electric field respectively [7].



Analytic formula:

$$\sigma = \pi a^2$$

$$a = 50mm, \sigma = -21.05\text{dB}$$

Figure 13. RCS simulation data from IE solver for sphere of radius 5cm. Analytic formula is also given to compare the result. $\vec{k}$ and $\vec{e}$ denotes direction of propagation and electric field respectively. It is obvious that RCS is constant for sphere.

3.3.2 *RCS Comparison Analysis of NASA RAM C-III Vehicle*

One of the most well-known example of re-entrant flight vehicle is NASA RAM (Radio Attenuation Measurement) C-III. The reason we chose this in our analysis is twofold: 1) the dimension of this vehicle is known 2) The vehicle is a complex geometry constituting of common features such as disc, half-sphere (dome) and conical cylinder. Although we do not have RCS data for RAM C-III, the modeling of RAM C-III would give us idea about amount of reflected signal as a function of angles. We modeled the RAM C-III as shown in Figure 14. Then we broke down the features if RAM C-III into different common geometric shape (e.g. disc, conical cylinder, and sphere) for certain angular incidence. The RCS comparison results are shown in Figure 15. As shown, the simulated results for discreet features such as sphere, disc, conical cylinder matches with RAM C-III results with reasonable accuracy (i.e. in terms of profile, trend and value in certain angle ranges ($80° \leq \theta \leq 180°$)). However, the deviation in results was found mostly between simulated RAM C-III data and analytic data for conical cylinders. The analytic plot was based on PO-based approximation method and it can be concluded that PO method fails to capture reasonable amount of scattered signal for curved surfaces such as conical cylinder and RAM C-III. The result also shows that in mono-static case, the specular reflection is pretty dominant in these type of vehicles even if the surface is smooth. These things have been considered later in analysis of HGV model.

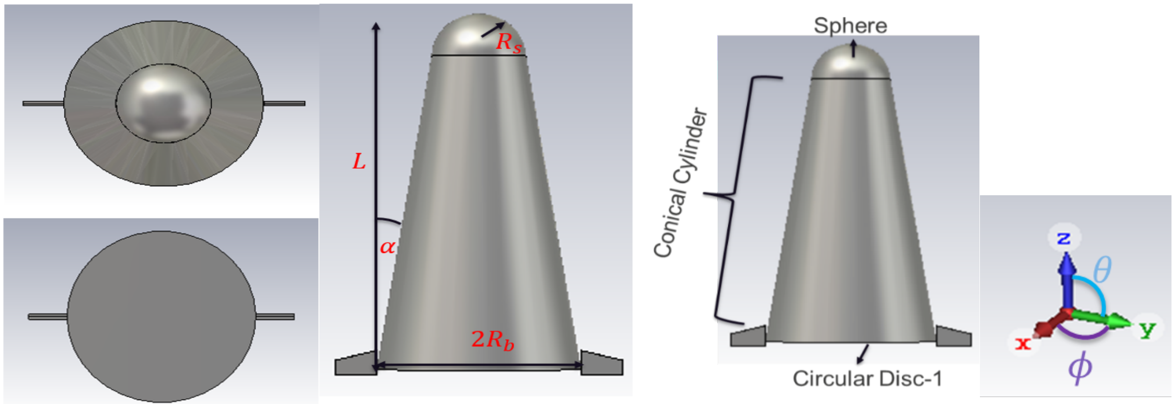


Figure 14. (Left) CSTMWS model of NASA RAM C-III Vehicle. The dimensions are: $R_s = 15.54cm, L = 129.5cm, \alpha = 9°, R_b = 33.53cm$. (Right) Breakdown of RAM C-III vehicle into three simple shapes

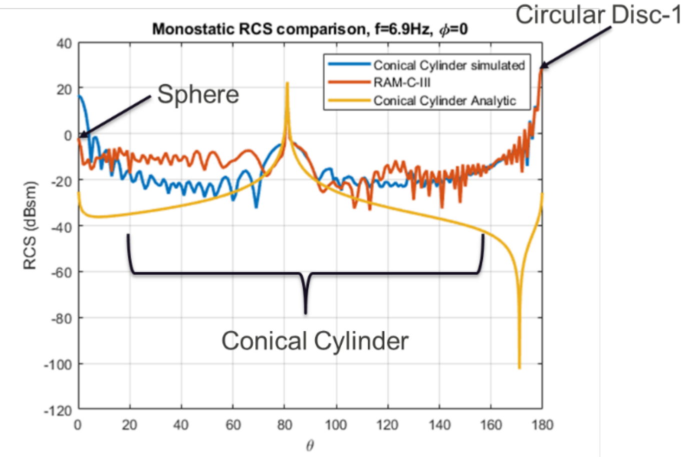


Figure 15. Comparison of monostatic RCS data between simulated RAM C-III and analytic models and simulated models. RAM C-III data matches reasonable with conical cylinder data for broad range of angles but fails to match considerably with PO approximation. The sphere and disc represent analytic model data as shown above

3.4 HGV Model

After validating our simulation model, for RCS analysis, we modeled our own HGV that mimic a HGV published in literature []. The model utilizes basic dimension parameters such as maximum length, cone angle and butt-length, as shown in Figure 16. The d/λ ratio for maximum length is 84.41, which makes it electrically large RCS problem.

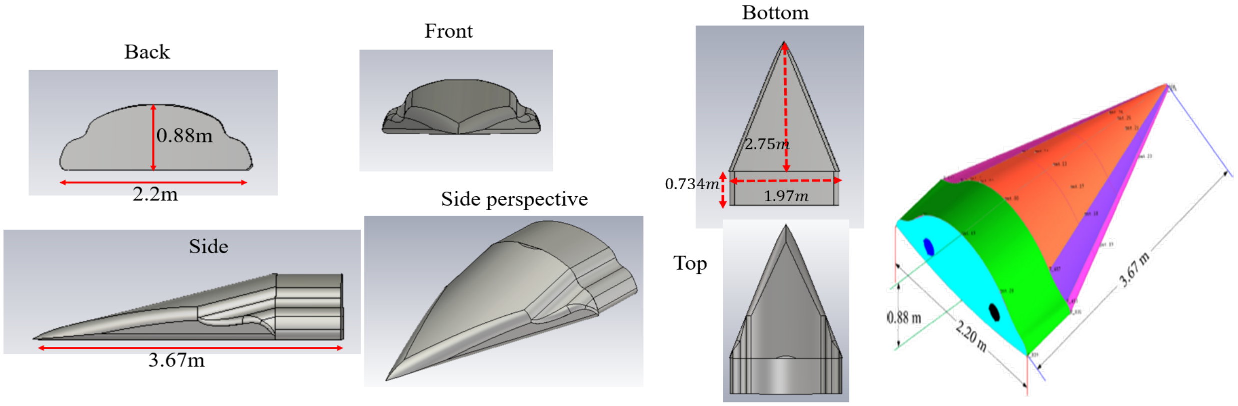


Figure 16 (Left) HGV model for our analysis that follows an existing HGV model (Right)

3.4.1 *RCS Analysis of HGV Model*

Before we would proceed with overall RCS analysis of the HGV for all possible angles, we wanted to determine average simulation time per bistatic pair of angles so that we can predict the total simulation time for our study. The required bi-static pair of angles that should accurately predict multi-static model of the RCS for our problem was identified as 254k pairs, where 10° resolution was considered to be enough to provide sufficient data for accurate interpolation of RCS in intermediate angles (within 10° resolution). For this huge number of bi-static angle pairs, it is necessary to determine which computational method is efficient to provide us RCS data with reasonable accuracy. Since HGV model is assumed to be fully metallic (PEC) and also electrically large ($\frac{d}{\lambda} = 84.41$), we short-listed two solvers to compare time and accuracy. One is IE solver that uses MLFMM. This is our benchmark process that is proved to provide accurate results with error margin less than 1% (Figures Figure 11Figure 13). The other one is Asymptotic Solver (AS) that uses SBR method which is very common in simulating electrically large problems with reasonable accuracy. Before we compare two solvers, a convergence study was conducted for IE solver since it is the reference solver for our analysis. We discretized the surface mesh size into two different lengths ($\lambda/10$ and $\lambda/15$). Then we compared the results side by side. For convenience of our analysis, we choose a set of angles ($\theta = 90°, \phi = 270°$) where the geometric feature is easily comprehensible. The physical features visible in this angle include a triangle and a rectangle (Fig. 10). The total physical area for this surface is $4.154m^2$. According to the physical optics approximation, the monostatic RCS from normal incidence of an area of A_p is $\sigma \approx \frac{4\pi A_p^2}{\lambda^2}$.

Radar Cross Section

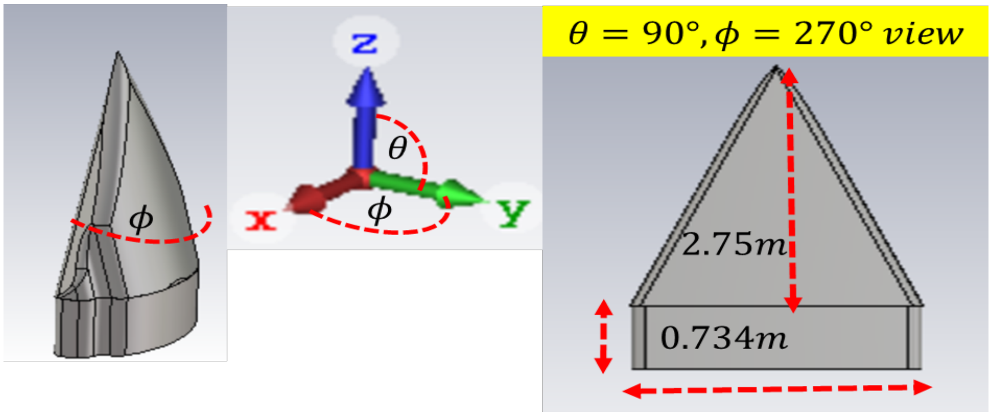


Figure 17. HGV model file simulated only in X-Y plane ($\theta = 90°, \phi = 270°$). The flat view of the HGV has a triangle and rectangle feature on it.

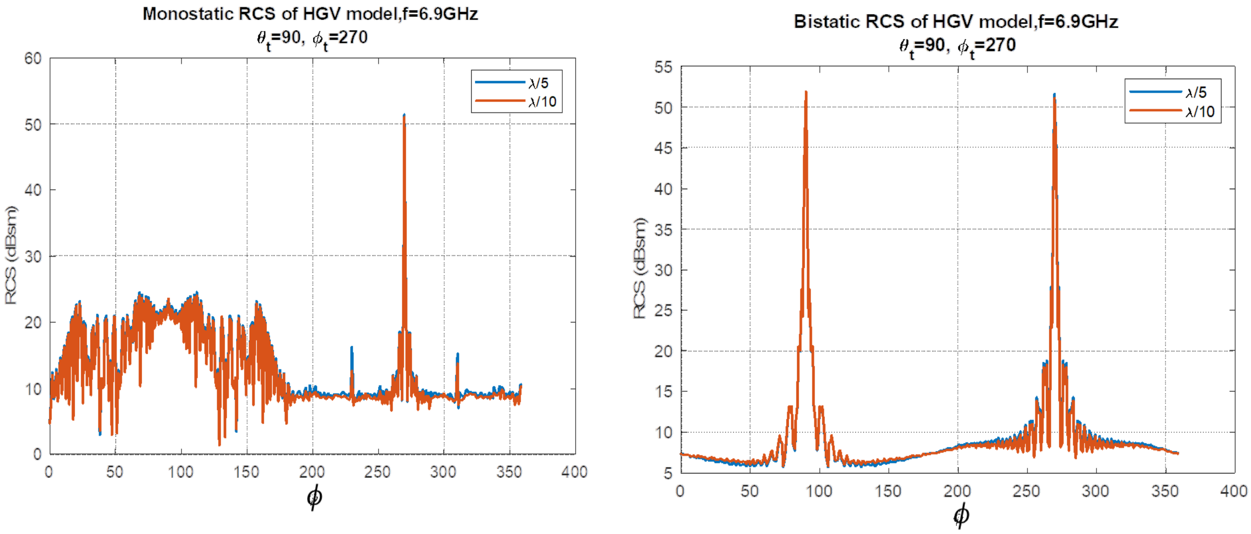


Figure 18. Monostatic and bistatic data for HGV vehicle, $\theta_t = 90°, \phi_t = 270°, \theta_r = 90°$, RCS is plotted against ϕ_r. In bistatic case, specular reflection is maximum for $\phi_r = 90°$, which is seen in shadow region.

The total physical area, A_p is $4.154m^2$, when looking at $\theta = 90°, \phi = 270°$. Therefore, theoretically predicted monostatic RCS is $50.59dBsm$ which is close to simulated one $(52dB)$ (Figure 18). The bi-static results also matches with less than 1% error. It is important to note that the mesh size $\lambda/10$ provides almost similar accuracy compared to $\lambda/15$ meshes. Therefore, we stick to $\lambda/10$ for comparison to AS solver results. A comparison between two different mesh cases and solver time is shown in Table 1.

Table 1. RCS Simulation Stats

Solver	Integral Equation Solver	Integral Equation Solver
Max cell size	$\lambda/5$	$\lambda/10$
Number of mesh cells	0.416M	1.309M
Simulated angles	$(\theta = 90°, \phi = 0° \; to \; 360°)$	$(\theta = 90°, \phi = 0° \; to \; 360°)$
Solver time	61.5h	130.5h
Solver time per angle	10.25m	21.75m

As shown, simulation time takes longer due to its exact nature of the solution. While the mesh number increases almost 3 folds by reducing mesh size, the accuracy remains almost same, implying the mesh is converged for $\lambda/10$.

Next, we compared results between two solvers (between IE and AS solvers) in Figure 19 for similar scenario. As seen, the data matches in considerable accuracy for most of the range of ϕ_r.

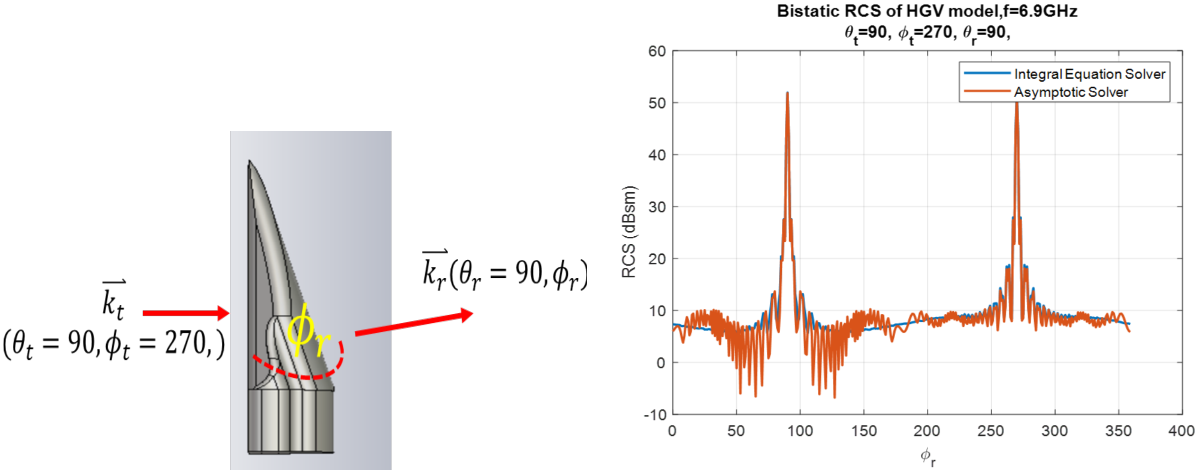


Figure 19. Comparison of IE and AS solver results for similar case as shown in Figure 17. The simulation data represents X-Y plane only. The results match with excellent accuracy in all regions except $120° < \phi_r < 150°, 50° < \phi_r < 70°$ where the fluctuation can be attributed to relying on specular reflection process for those angles. The fluctuation is maximum 5dB which can be minimized by post-processing of the data since the trend is same. The result also followed theoretically predicted data $\approx 51dB$ at the peak.

We also considered X-Z plane for simulation so that the results are in considerable accuracy. This is shown in Fig. 13. Although there are several locations where the AS solver data differ from IE solver data, there general trend is similar. The deviation can be attributed to couple of factors [9]: 1) IE solver is exact whereas AS is approximate, 2) IE solver includes other scattering processes such as diffraction, surface waves, where AS solver relies mainly on specular reflection [9].

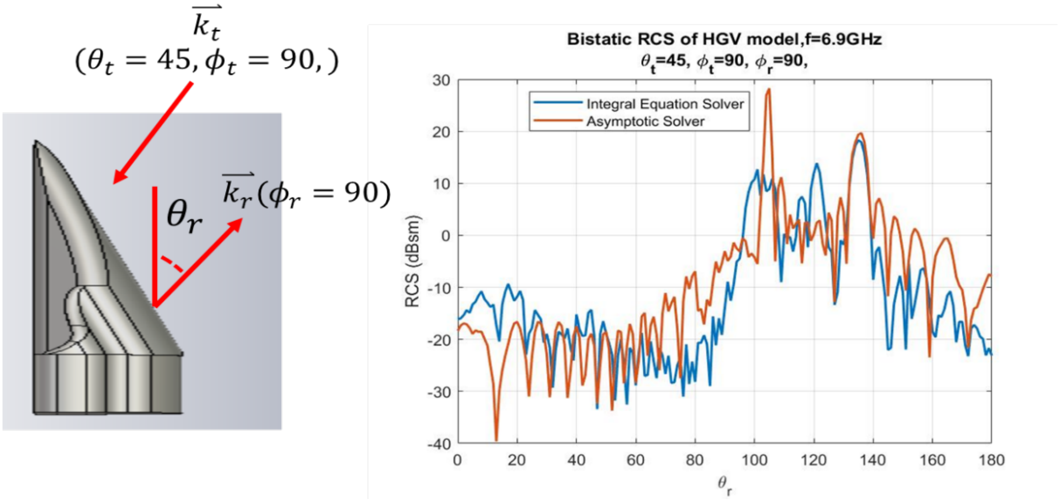


Figure 20. Comparison of IE and AS solver results for similar case as shown in Fig. 10. The simulation data represents X-Y plane only. As seen in the plot the maximum reflection observed in AS solver at $\theta_r = 135°$, $90°$ apart from transmitter angle, $\theta_t = 45°$. This is a simple verification of specular reflection. This follows general optical law of reflection.

The objective of this study was to compare simulation time between two solvers. The results are shown in Table 2

Table 2. RCS Simulation Stats

Solver	Integral Equation Solver	Asymptotic Solver
Max cell size	$\lambda/10$	N/A
Number of mesh cells	1.61M	21038 $surfaces$
Simulated angles	$(\theta_t = 90°,\ \phi_t = 270°,\ \phi_r = 90°,\ \theta_r = 90°,\ \phi_r = 0\ to\ 360°)$ $(\theta_t = 45°,\ \phi_t = 90°,\ \phi_r = 90°,\ \theta_r = 0\ to\ 90°)$	
Solver time	47.5h (180 angles)	2.25m ((180 angles))
Solver time per angle	~16m	0.75s

Table 2 shows that AS solver reduces simulation time dramatically from 16 minutes to only ¾ seconds. This is a reasonable approach to analyze $250k$ RCS pair of angles within reasonable accuracy. Also, our basic assumption is that most of the signals coming from or captured by satellites would be specular in nature, therefore AS solver seems a promising solution analyzing the data. The simulated RCS data for the HGV model is stored in the repository in file name titled: "RCSdata_S_data_lin_pol_lin_format.xlsx".

3.5 Polarization Dependence and Post-Processing of RCS data:

The RCS simulation was conducted using CSTMWS [11], for two orthogonal basis of incident electric field: 1) E_θ^i and 2) E_ϕ^i i.e. we simulated and tabulated scattered field for each incident field and stored them together in the repository file. This is due to the fact that polarization affects scattered power from a target significantly. For example, a purely θ-polarized (i.e. E_θ^i) incident field can scatter waves which can have both θ and ϕ components i.e. E_θ^s and E_ϕ^s. Hence, RCS analysis can be imagined as two-port network. This can be represented as the following equation [7[13[14]:

$$\begin{bmatrix} E_\phi^s \\ E_\theta^s \end{bmatrix} = \begin{bmatrix} S_{\phi\phi} & S_{\phi\theta} \\ S_{\theta\phi} & S_{\theta\theta} \end{bmatrix} \begin{bmatrix} E_\phi^i \\ E_\theta^i \end{bmatrix} \tag{14}$$

Where S-parameters are in complex form, i.e. having both magnitude and phase. The RCS is related to this S-matrix using the following relation:

$$\begin{bmatrix} \sigma_{\phi\phi} & \sigma_{\phi\theta} \\ \sigma_{\theta\phi} & \sigma_{\theta\theta} \end{bmatrix} = 4\pi r^2 \begin{bmatrix} \left|S_{\phi\phi}\right|^2 & \left|S_{\phi\theta}\right|^2 \\ \left|S_{\theta\phi}\right|^2 & \left|S_{\theta\theta}\right|^2 \end{bmatrix} \tag{15}$$

We simulated all the S-parameter elements ($S_{\theta\theta}, S_{\theta\phi}, S_{\phi\theta}, S_{\phi\phi}$) for each pair of angles. This data is a basis for any combination of transmitter and receiver antenna polarizations. For example, if the transmitter and receiver antenna are both from circular polarization, then this matrix can be used to convert the linear-polarization basis to circular polarization using the following unitary transformation:

$$S_c = \begin{bmatrix} S_{RR} & S_{RL} \\ S_{LR} & S_{LL} \end{bmatrix} = [T] \begin{bmatrix} S_{\phi\phi} & S_{\phi\theta} \\ S_{\theta\phi} & S_{\theta\theta} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} [T]^{-1}$$

$$\text{Where } T = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -j \\ 1 & j \end{bmatrix} \tag{16}$$

'R' represents Right-handed circular polarization (RHCP) and 'L' represents left-handed circular polarization (LHCP). The matrix $[\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}]$ is used here to compensate for the propagation direction of transmitted and received plane wave into unitary transformation [15]. This is shown in radar cross section appendix. However, for any arbitrary polarization of transmitter and receiver, the unitary transformation can be quite complex. For our analysis, we chose linear components such as E_θ and E_ϕ as basis of transformation [4[5]. We preferred to represent any polarization (Tx or Rx) of wave as linear combination of these two components. Typically, polarization of a wave is represented as coordinates of a fictitious sphere, called Poincaré Sphere [4]. We also followed this standard convention in our analysis. To elaborate this, we refer to Poincaré Sphere as shown in Figure 21.

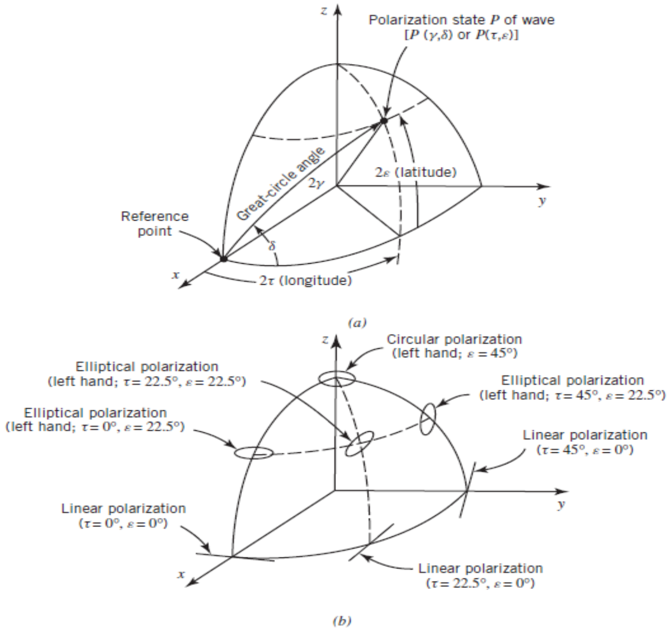

z
Polarization state P of wave
$[P(\gamma,\delta)$ or $P(\tau,\varepsilon)]$
Great-circle angle
2γ
2ε (latitude)
y
Reference point
δ
x
2τ (longitude)
(a)
z
Circular polarization (left hand; $\varepsilon = 45°$)
Elliptical polarization (left hand; $\tau = 22.5°$, $\varepsilon = 22.5°$)
Elliptical polarization (left hand; $\tau = 45°$, $\varepsilon = 22.5°$)
Elliptical polarization (left hand; $\tau = 0°$, $\varepsilon = 22.5°$)
Linear polarization ($\tau = 45°$, $\varepsilon = 0°$)
y
Linear polarization ($\tau = 0°$, $\varepsilon = 0°$)
x
Linear polarization ($\tau = 22.5°$, $\varepsilon = 0°$)
(b)

Figure 21. Poincaré Sphere as shown in [4]. Any polarization of a wave can be represented using its polarization state $P(\gamma,\delta)$ or $P(\epsilon,\tau)$. (a) Definitions of coordinates/states are shown (b) location of states for different states are shown.

The polarization of a wave can be represented by longitude (2τ) and latitude angle (2ϵ) in the Poincaré Sphere [4]. Unit vector of electric field of any arbitrary polarization can be represented as:

$$\hat{E} = a\widehat{E_\phi} + b\,\widehat{E_\theta}e^{j\angle\delta} \qquad 17$$

From conversion relation of longitude and latitude angles to (γ,τ) , $\vec{E}$ can be represented as linear combination of , $\overrightarrow{E_\theta}$ and , $\overrightarrow{E_\phi}$. The conversion relations are [4]:

$$\begin{gathered}\gamma = \tan^{-1}\frac{b}{a}, \qquad \delta = \angle\theta - \angle\phi \\ \sin(2\epsilon) = \sin 2\gamma \sin\delta \\ \sin(2\tau) = \tan 2\gamma \cos\delta\end{gathered} \qquad 18$$

For any given set of (ϵ,τ), using the above relations, a, b and δ can be derived. For example, for circular polarization, $a = b = 1, \delta = \pm\frac{\pi}{2}$. For convenience of our analysis, we chose $a = 1$ and then used γ to calculate b. We developed a python code that uses these inputs and above equations and converts any arbitrary polarization (linear, circular or elliptical) into a linear combination of vectors with appropriate phase (δ_t, δ_r) and magnitudes a_t, b_t, a_r, b_r. Next, a unitary transformation of basis for different set of polarizations can be mathematically formulated to obtain RCS for arbitrary polarization of transmitter and receiver:

$$\sigma(\theta_t,\phi_t,\theta_r,\phi_r,\epsilon_t,\tau_t,\epsilon_r,\tau_r) = \frac{1}{A}\begin{bmatrix}a_t & b_t e^{j\delta_t}\end{bmatrix}\begin{bmatrix}|S_{\theta\theta}|^2 & |S_{\theta\phi}|^2 \\ |S_{\phi\theta}|^2 & |S_{\phi\phi}|^2\end{bmatrix}\begin{bmatrix}1 & 0 \\ 0 & -1\end{bmatrix}\begin{bmatrix}a_r \\ b_{re^{j\delta_r}}\end{bmatrix}$$

$$A = \frac{1}{\sqrt{a_t^2 + b_t^2}} \frac{1}{\sqrt{a_r^2 + b_r^2}} \qquad 19$$

Here subscripts, ‘t’ and ‘r’ refers to transmitter and receiver respectively. A is the scale factor for the transformation. We used this transformation for each set of Tx and Rx pair of angles i.e. $(\theta_t, \phi_t, \theta_r, \phi_r)$.

3.6 References:

[4] C. A. Balanis, *Advanced engineering electromagnetics*. John Wiley & Sons, 2012.

[5] C. A. Balanis, *Antenna theory: analysis and design*. John wiley & sons, 2015.

[6] A. E. Fuhs, "Radar cross section lectures," NAVAL POSTGRADUATE SCHOOL MONTEREY CA, 1982.

[7] L. A. A. Harrison, *Introduction to Radar Using Python and MATLAB*. Artech House, 2019.

[8] R. M. O’Donnell, "Radar Systems Engineering Lecture 9 Antennas," *IEEE New Hampshire Section,* 2010.

[9] J. Smit, J. Cilliers, and E. Burger, "Comparison of MLFMM, PO and SBR for RCS investigations in radar applications," 2012.

[10] T. Savides and B. Dwolatzky, "Radar simulation using the shooting and bouncing ray technique," in *CCECE 2003-Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology (Cat. No. 03CH37436)*, 2003, vol. 1: IEEE, pp. 307-310.

[11] "CST Studio Suite." https://www.3ds.com/products-services/simulia/products/cst-studio-suite/ (accessed.

[12] Q. Niu, S. Yang, Z. He, and S. Dong, "Numerical study of infrared radiation characteristics of a boost-gliding aircraft with reaction control systems," *Infrared Physics & Technology,* vol. 92, pp. 417-428, 2018.

[13] D. Jenn, *Radar and laser cross section* engineering. American Institute of Aeronautics and Astronautics, Inc., 2005.

[14] B. R. Mahafza, Radar Systems Analysis and Design Using MATLAB Third Edition. CRC press, 2013.

[15] G. Ruck, Radar Cross Section Handbook: Volume 1. Springer, 1970.

3.7 Radar Cross Section Appendix

For any set of transmitter and receiver angles $(\theta_t, \phi_t, \theta_r, \phi_r)$, the bistatic scattering matrix of a radar system can be represented by a 2×2 matrix. This is due to polarization dependent scattering from the surface of an object. For example, the scattered wave from an incident θ-polarized wave can have two orthogonal components i.e. E_θ and E_ϕ. Therefore, any scattered wave can be characterized by two orthogonal basis i.e. (θ, ϕ) or (RHCP, LHCP) or $(\widehat{a_\parallel}, \widehat{a_\perp})$. For now, we show the transformation in circular polarization basis. Any circular polarized wave can be represented using two orthogonal component of linear basis (E_θ, E_ϕ), whose amplitude are equal and phase shifted by 90°. Using the notation mentioned in the chapter, arbitrary polarized can be written:

$$\hat{E} = a e^{j\angle\delta}\,\widehat{E_\theta} + b\widehat{E_\phi} \tag{20}$$

For circular polarization, $a = b = 1$ and $\delta = \pm\frac{\pi}{2}$. If the wave is RHCP, $\delta = -\frac{\pi}{2}$ and if the wave is LHCP, $\delta = +\frac{\pi}{2}$. Hence, the unit vector of electric field of a RHCP wave can be written as the following linear equation:

$$\widehat{E_R} = \widehat{E_\phi} - j\widehat{E_\theta} \tag{21}$$

Similarly for LHCP

$$\widehat{E_L} = \widehat{E_\phi} + j\widehat{E_\theta} \tag{22}$$

$$\begin{bmatrix} E_R \\ E_L \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -j \\ 1 & j \end{bmatrix}\begin{bmatrix} E_\phi \\ E_\theta \end{bmatrix} = [T]\begin{bmatrix} E_\phi \\ E_\theta \end{bmatrix} \tag{23}$$

where $[T] = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -j \\ 1 & j \end{bmatrix}$ is the basis transformation matrix. By transforming the column-matrices, (E_θ, E_ϕ) can represented using (E_R, E_L):

$$\begin{bmatrix} E_\phi \\ E_\theta \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ j & -j \end{bmatrix}\begin{bmatrix} E_R \\ E_L \end{bmatrix} = [T]^{-1}\begin{bmatrix} E_R \\ E_L \end{bmatrix} \tag{24}$$

Now, for standard scattering matrix algebra can be shown in following equations:

$$[T]^{-1}\begin{bmatrix} E_R^s \\ E_L^s \end{bmatrix} = \begin{bmatrix} S_{\phi\phi} & S_{\phi\theta} \\ S_{\theta\phi} & S_{\theta\theta} \end{bmatrix}[T]^{-1}\begin{bmatrix} E_R^i \\ E_L^i \end{bmatrix} \tag{25}$$

The matrix $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ is included in unitary transformation to make general reference for transmitting and receiving wave. Typically, the polarization of any wave is represented as if all wave are in transmitting mode.

$$\begin{bmatrix} E_R^s \\ E_L^s \end{bmatrix} = [T]\begin{bmatrix} S_{\phi\phi} & S_{\phi\theta} \\ S_{\theta\phi} & S_{\theta\theta} \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}[T]^{-1}\begin{bmatrix} E_R^i \\ E_L^i \end{bmatrix} \tag{26}$$

Using definition of scattering matrix for circular polarization,

$$\begin{bmatrix} E_R^s \\ E_L^s \end{bmatrix} = \begin{bmatrix} S_{RR} & S_{RL} \\ S_{LR} & S_{LL} \end{bmatrix} \begin{bmatrix} E_R^i \\ E_L^i \end{bmatrix} \tag{27}$$

Then,

$$\begin{bmatrix} S_{RR} & S_{RL} \\ S_{LR} & S_{LL} \end{bmatrix} = [T] \begin{bmatrix} S_{\phi\phi} & S_{\phi\theta} \\ S_{\theta\phi} & S_{\theta\theta} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} [T]^{-1} \tag{28}$$

The resultant matrix is the transformation matrix for circular polarization basis. Now, if the transmitting and receiving wave are RHCP and LHCP respectively. The scattered power from the target is S_{LR}, which can be found from the following unitary transformation:

$$S_{LR} = \frac{1}{2}[1 \quad j] \begin{bmatrix} S_{\phi\phi} & S_{\phi\theta} \\ S_{\theta\phi} & S_{\theta\theta} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -j \end{bmatrix} \tag{29}$$

The algebraic expressions for all the transformation is given below:

$$S_{RR} = \frac{1}{2}(S\phi\phi + S_{\theta\theta}) - \frac{1}{2}j(S_{\theta\phi} - S_{\phi\theta}) \tag{30}$$

$$S_{RL} = \frac{1}{2}(S\phi\phi - S_{\theta\theta}) - \frac{1}{2}j(S_{\phi\theta} + S_{\theta\phi}) \tag{31}$$

$$S_{LR} = \frac{1}{2}(S\phi\phi - S_{\theta\theta}) + \frac{1}{2}j(S_{\phi\theta} + S_{\theta\phi}) \tag{32}$$

$$S_{LL} = \frac{1}{2}(S\phi\phi + S_{\theta\theta}) - \frac{1}{2}j(S_{\phi\theta} - S_{\theta\phi}) \tag{33}$$

As seen, S_{RR} and S_{LL} are expected to become larger in magnitude compared to S_{RL} and S_{LR}. This is due to the fact that scattered signal from metallic targets typically do not change polarization (i.e. $S_{\theta\phi}$ and $S_{\phi\theta}$ are smaller compared to $S_{\theta\theta}$ and $S_{\phi\phi}$). This has been also observed in numerical analysis.

4 Signal to Noise

We have studied the radar link budget and determined under what conditions the signal received by the receiver can support a determination of range and Doppler for the extended Kalman filter tracking engine described in the following section. In general a coherent signal is integrated over time to bring the integrated signal amplitude to a level sufficiently over the noise. We refer to this equivalently as a high enough SNR or a high enough E_w/N_0, where E_w is the energy in the waveform and $N_0 = f k_B T_r$ is the noise power spectral density, also an energy, and where f is the receiver's noise figure, k_B is Boltzmann's constant and T_r is the receiver temperature. The received power is of the form

$$P_r = P_t\,\sigma \frac{A_{phy}^2}{4\pi\lambda^2 R_t^2 R_r^2}\eta \tag*{34}$$

where P_t is the transmitted power, σ is the flight vehicle's radar cross section, A_{phy} is the area of the transmit and receive antennas, λ is the RF carrier frequency wavelength, R_t and R_r are the distances from the flight vehicle to the transmitter and receiver, respectively, and various inefficiencies (air attenuation, antenna loss, etc.) are incorporated in η. The ratio of energy in a single transmitted bit of the signal over the noise power spectral density is

$$\frac{E_b}{N_0} = \frac{{}^{P_r}\!/_{f_{bit}}}{N_0} \tag*{35}$$

where f_{bit} is the bit transmission rate, and

$$\frac{E_w}{N_0} = \frac{1}{N_0}\int_0^{t_{int}} P_r\ dt \tag*{36}$$

where t_{int} is the time over which the signal is coherently integrated and we are allowing the received power to vary over time due to changes in the radar cross section and ranges (and where we are arbitrarily defining the time of the start of the coherent integration as $t = 0$).

In the absence of significant flight-vehicle acceleration, the SNR requirement becomes

$$\tfrac{E_w}{N_0} \geq 10 \ \text{ or equivalently } \ \textstyle\int_0^{t_{int}} P_r\ dt\ > 10N_0 \tag*{37}$$

which can always be achieved with a long enough integration time (but which is probably only relevant for our tracking problem if the integration time doesn't exceed a couple of seconds or so).

The SNR requirement is more completed if the flight vehicle has significant acceleration,

$$\int_0^{t_{int}} P_r\,dt > 10N_0\left(1+\left(\frac{t_{int} f_{carrier}}{c}\int_0^{t_{int}} \ddot{R}dt\right)^2\right)^{1/2}\ , \tag*{38}$$

where here $\ddot{R}$ is the unknown effective acceleration in the overall range (R_t plus R_r) due to the flight vehicle's actual acceleration plus the geometric changes that occur even with no acceleration and which can be large if the flight vehicle is close to either the transmitter or receiver. While some of the total effective acceleration can be estimated (from the known motion of the flight vehicle, transmitter, and receiver), effects due to the flight vehicle's actual acceleration cannot. The equation above leads to an instantaneous minimum received power requirement

$$P_r \geq 10\sqrt{2}N_0\sqrt{\frac{\ddot{R}f_{carrier}}{c}} \quad (39)$$

which, if not met, means the signal will not help in closing the link budget at that instant in time.

The following sections show where these equations come from and contain results from numerical experiments intended to validate the theory and to determine the constants used. Since we tried to faithfully numerically model a transmitter/flight vehicle/receiver system, the first section following this introductory summary describes how we generated signals and the noise associated with that process. We then provide background on the ambiguity function approach we use to determine range and Doppler. Following that, we describe the effect of adding white average Gaussian noise to the signal (to represent the noise power spectral density), the relation between the peak noise and the peak signal amplitudes, how we determined the minimum E_w/N_0 value for detecting the signal after time integration, and then the impact from including the flight vehicle's acceleration.

4.1 Background – Gold Code signal and signal orthogonality

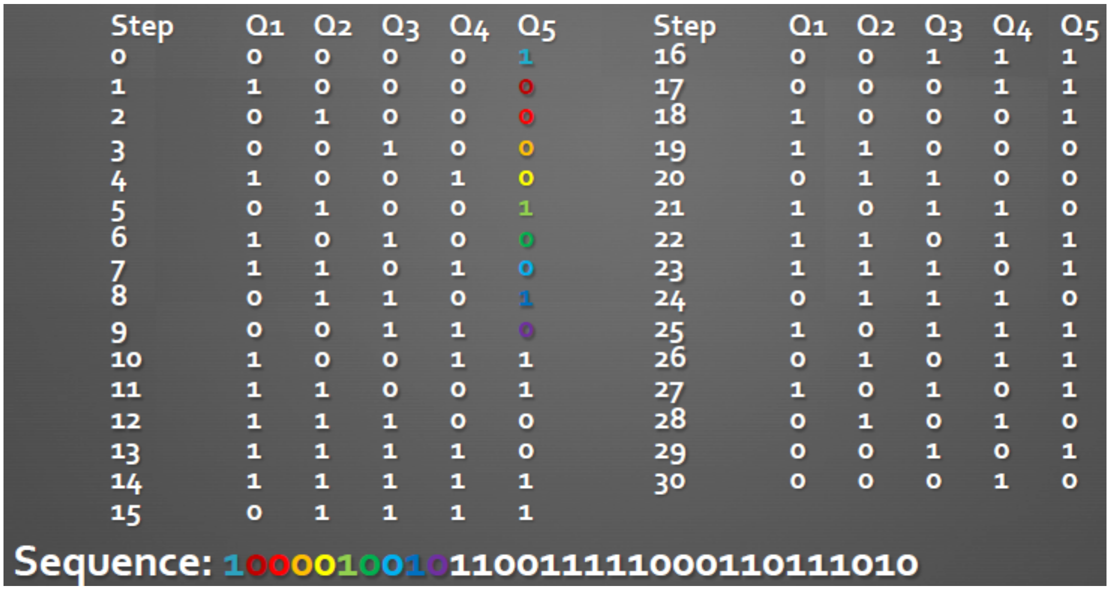
For studying the SNR requirements, we assumed each satellite transmitted its own orthogonal signal, specifically a Gold Code. A Gold Code generation requires two steps:

1. Generation of a Maximum Length Sequence (MLS) with a Linear Feedback Shift Register (LFSR)
2. Generation of the actual Gold Code by XORing 2 MLS's generated with different taps.

An example for step 1 is shown in Figure 22. MLS example using a 1+x3+x5 polynomial and seed of (0,0,0,0,1).using an LRSR of length 5 with a $1+x^3+x^5$ polynomial for the taps, which means the input (in Q_1) is the last time step's output (Q_5 in Fig. 1) XOR'ed with the value in the third register (Q_3), and that all the other registers Q_n are given the value in the previous register Q_{n-1}. The actual MLS is the series of values that end in Q_5. All sequences need a seed, here the initial seed is (0,0,0,0,1) and the MLS sequence is

1000010010110011111000110111010 1

before it repeats itself. The length of the MLS is 2^n-1, or 31 in this example.



Step	Q1	Q2	Q3	Q4	Q5	Step	Q1	Q2	Q3	Q4	Q5
0	0	0	0	0	1	16	0	0	1	1	1
1	1	0	0	0	0	17	0	0	0	1	1
2	0	1	0	0	0	18	1	0	0	0	1
3	0	0	1	0	0	19	1	1	0	0	0
4	1	0	0	1	0	20	0	1	1	0	0
5	0	1	0	0	1	21	1	0	1	1	0
6	1	0	1	0	0	22	1	1	0	1	1
7	1	1	0	1	0	23	1	1	1	0	1
8	0	1	1	0	1	24	0	1	1	1	0
9	0	0	1	1	0	25	1	0	1	1	1
10	1	0	0	1	1	26	0	1	0	1	1
11	1	1	0	0	1	27	1	0	1	0	1
12	1	1	1	0	0	28	0	1	0	1	0
13	1	1	1	1	0	29	0	0	1	0	1
14	1	1	1	1	1	30	0	0	0	1	0
15	0	1	1	1	1						

Sequence: 1000010010110011111000110111010

Figure 22. MLS example using a 1+x3+x5 polynomial and seed of (0,0,0,0,1).

A different MLS is achieved with the same polynomial if a different seed is used, so there are also 2^n-1 unique MLS's possible with this length LRSR and polynomial.

For step 2, we XOR two MLS's generated with different taps, shown in Figure 23. Construction of a Gold Code using two MLS sequences.

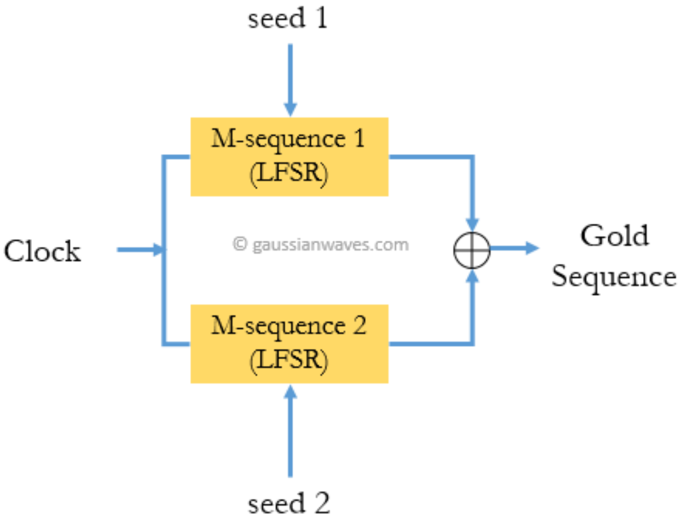
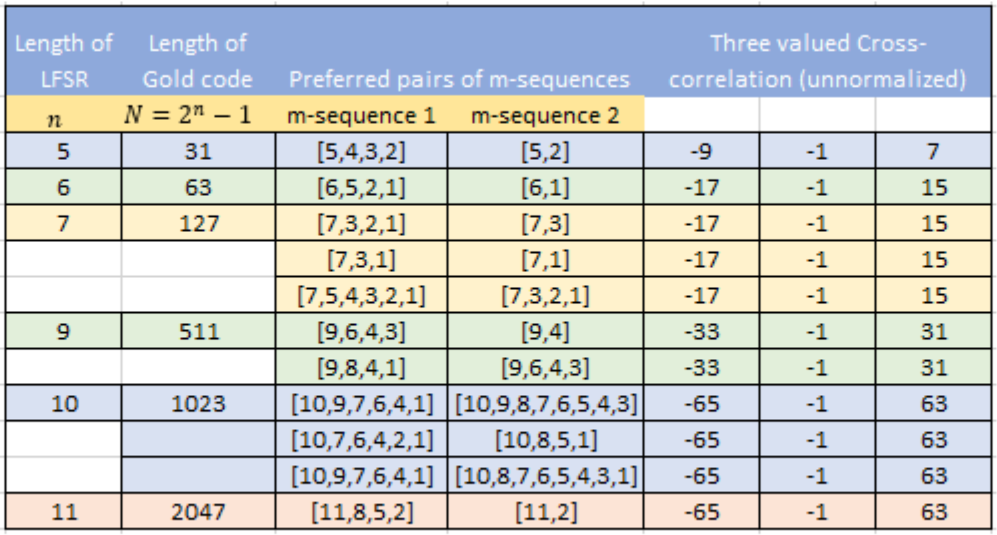


Figure 23. Construction of a Gold Code using two MLS sequences.

Here we use different polynomials for both MLS's, and convention is to use (0,0,…,0,1) for the first MLS seed. Each seed for the second MLS gives a different Gold Code (so there are again 2^n-1 Gold Codes in each family). Each Gold Code has an auto-correlation of 2^n-1 with itself and maximum cross-correlation of 2(n+1)/2+1 with other Gold Codes in its family. "Preferred" pairs of MLS's have the lowest cross-correlation magnitudes (Figure 24).

Signal to Noise



Length of LFSR	Length of Gold code	Preferred pairs of m-sequences		Three valued Cross-correlation (unnormalized)		
n	$N = 2^n - 1$	m-sequence 1	m-sequence 2			
5	31	[5,4,3,2]	[5,2]	-9	-1	7
6	63	[6,5,2,1]	[6,1]	-17	-1	15
7	127	[7,3,2,1]	[7,3]	-17	-1	15
		[7,3,1]	[7,1]	-17	-1	15
		[7,5,4,3,2,1]	[7,3,2,1]	-17	-1	15
9	511	[9,6,4,3]	[9,4]	-33	-1	31
		[9,8,4,1]	[9,6,4,3]	-33	-1	31
10	1023	[10,9,7,6,4,1]	[10,9,8,7,6,5,4,3]	-65	-1	63
		[10,7,6,4,2,1]	[10,8,5,1]	-65	-1	63
		[10,9,7,6,4,1]	[10,8,7,6,5,4,3,1]	-65	-1	63
11	2047	[11,8,5,2]	[11,2]	-65	-1	63

Figure 24. List of preferred pairs of MLS sequences, where [n,m] indicates taps at the m and n registers, or a polynomial of the form $1+x^m+x^n$.

For numerically modeling isolating the signal from one satellite, we use a dot product between the total received signal and the code of the satellite signal we want to isolate. First, we use a 1 or -1 (instead of a 0) for each satellite signal and multiply by the transmitted signal with a varying time shift (also either a 1 or a -1) and sum the results. For the following simulations with 6 separate Gold Codes on 6 satellites, we use the 11-register preferred pair LFSRs with polynomials $1+x^2+x^5+x^8+x^{11}$ and $1+x^2+x^{11}$. We use the seed (1,0,0,0,0,0,0,0,0,0,0) for $1+x^2+x^5+x^8+x^{11}$ and the six seeds

(1,0,0,0,0,0,0,0,0,0,0)

(0,1,0,0,0,0,0,0,0,0,0)

(0,0,1,0,0,0,0,0,0,0,0)

(0,0,0,1,0,0,0,0,0,0,0)

(0,0,0,0,1,0,0,0,0,0,0)

(0,0,0,0,0,1,0,0,0,0,0)

for the $1+x^2+x^{11}$ polynomial, for each of the six satellite signals.

The signals from the first three satellites and the summed signal from all six satellites is shown in Figure 25. The dot product using the transmitted signal from each of the first three satellites varying the time delay is shown in Figure 26.

The magnitude of the cross correlations of these codes is not a limiting factor in this simulations (because we only have 6 satellites total), but also wouldn't be for a large constellation of thousands of satellites, because longer LFSR's can be picked with arbitrary low levels of cross correlations.

Signal to Noise

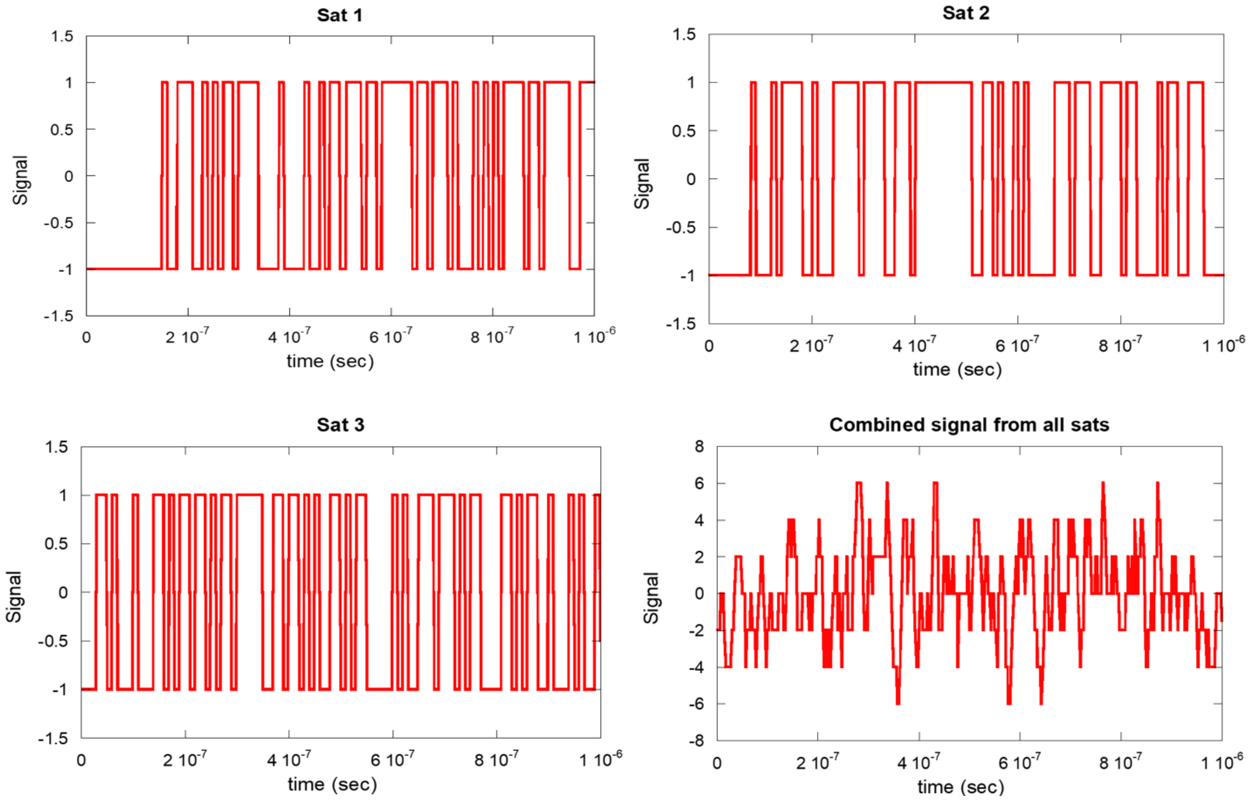
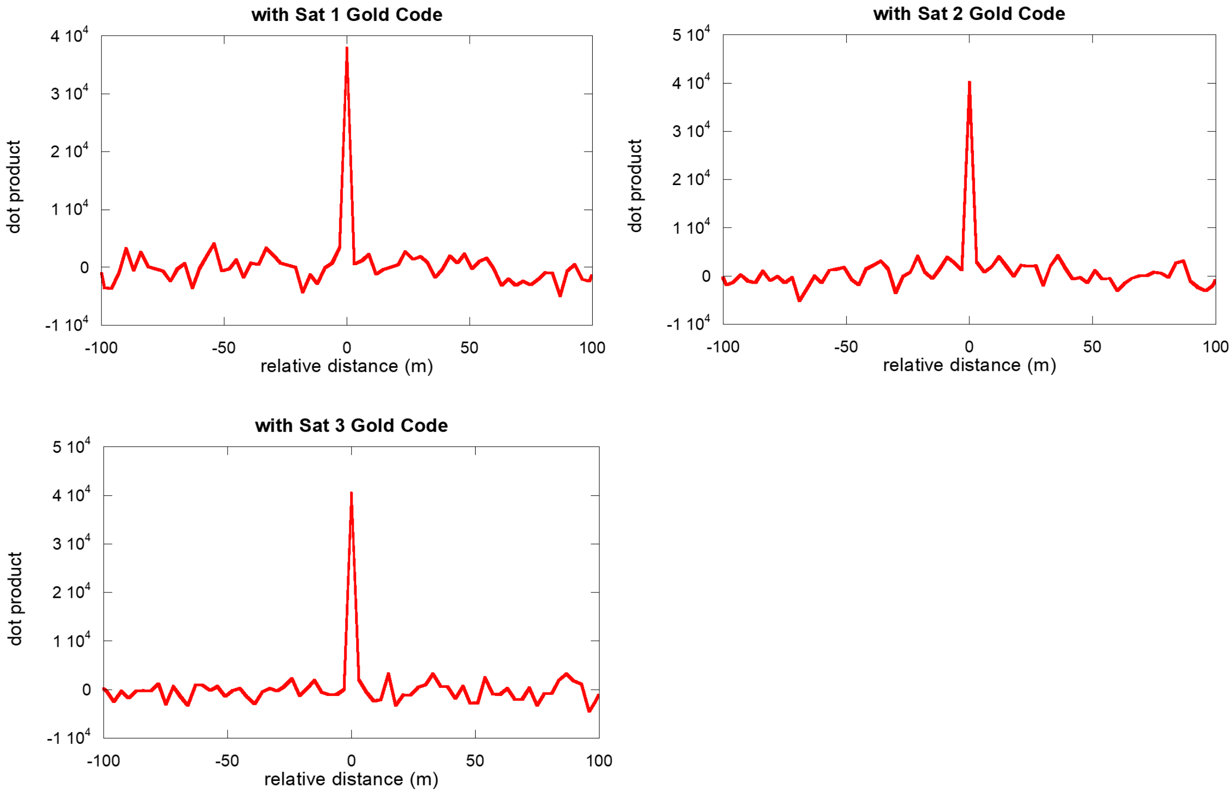


Figure 25. First part of transmitted code from satellites 1-3 (note the effect of the seed on the Gold Code start) and summation of all six satellites' Gold Codes at the receiver on satellite 1.



with Sat 1 Gold Code

with Sat 2 Gold Code

with Sat 3 Gold Code

Figure 26. The dot product between the total received signal the codes from satellites 1, 2, and 3, respectively, varying the time delay (normalized to a relative distance of zero). The peaks are larger than 2047 because the dot product is taken over a time greater than one full waveform.

4.2 Background – ambiguity function

We use an ambiguity function to determine range and Doppler (i.e., range rate of change) for all monostatic and bistatic links as shown in Figure 27

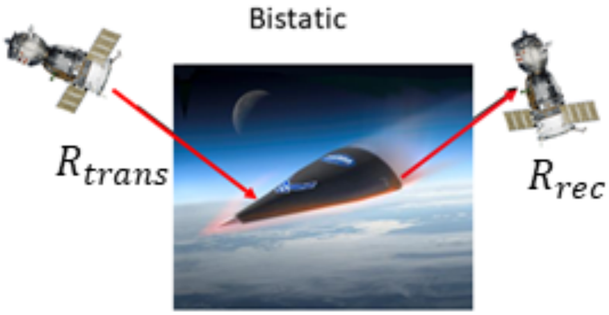


Figure 27. Definition of R_{trans} and R_{rec}.

When integrating the ambiguity function

$$A(\Delta\tau, \Delta f) = \int_{0}^{T} s_{rec}(t) s^{*}_{trans}(t + \Delta\tau) \exp\{-j2\pi\, \Delta f\, t\} dt \tag{40}$$

over a time interval T and with a physical range offset $\Delta\tau$ and a frequency offset Δf, where $s_{trans}(t)$ and $s_{rec}(t)$ are the transmitted and received signal, we expect to $A(\Delta\tau, \Delta f)$ to have the form shown in Fig. 7.

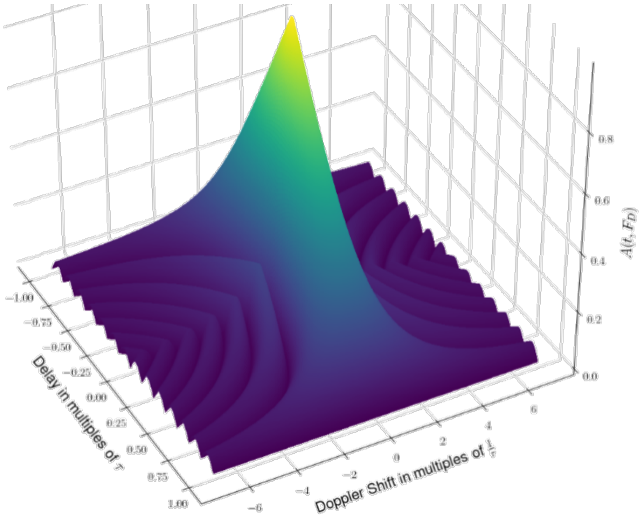


Figure 28. Typical ambiguity function integration

The offsets in the peak location in $\Delta\tau$ and Δf give the range and Doppler information, specifically

$$c\,\Delta\tau/2 = R_{trans} + R_{rec} \text{ and } \frac{c\Delta f}{f_{carrier}} = \frac{d}{dt}\left(R_{trans} + R_{rec}\right) \tag{41}$$

Numerically, at time *t* we calculated the retarded time of the flight vehicle and then the retarded times of all the satellites for the RF that hits the flight vehicle at its retarded time. Then we accumulated the Gold code signals (using the 11 register, 2047 bit code described above) from both satellites with the retarded times

$$signal(t) = signal(t) + (GC_{trans})cos\left(2\pi f_{carrier}(t - t_{ret-FV} - t_{ret-trans})\right) \tag{42}$$

We used the following formula to find the dot product with the Gold code from either satellite $\left(\text{where } df = 1 + \frac{\Delta f}{f_{carrier}}\right)$:

$$\begin{aligned} dot\,product &= dot\,product \\ &\quad + signal(t)(GC_{trans}|_{df})cos\left(2\pi(f_{carrier} \times df)\right)(t - \Delta\tau - t_{cent} + (df-1)t) \end{aligned} \tag{43}$$

The area resolution of the ambiguity function is

$$\Delta\tau\Delta f = \frac{1}{\text{number of codes} * \frac{\text{bits}}{\text{code}}} \tag{44}$$

This can lead to excellent resolutions in both spatial distance and Doppler velocity, with the resolution in range being the bit "size" and the resolution in velocity being related to the integration time and the range resolution:

$$\Delta R_{res} = \frac{c}{2f_{bit}} \tag{45}$$

$$\Delta v_{res} = \frac{c}{f_{carrier}t_{int}} \tag{46}$$

as long as the SNR requirement is satisfied. (The spatial resolution simply results from knowing which code bin the flight vehicle is in; the velocity resolution arises from the Doppler condition $\frac{\Delta v_{res}}{c} = \frac{\Delta f}{f_{carrier}}$ and the frequency resolution $\Delta f = 1/t_{int}$).

We show our model geometry in Figure 29. We position a transmitter vertically (in *y*) 1000 km above the flight vehicle and a receiver horizontally (in *x*) 1000 km to the right of the flight vehicle. We let the flight vehicle move in the +*y* direction, nominally at 300 msec. We use a carrier of 6.9 GHz and a code bit frequency of 100 MHz. We integrate for different times with about 10k time samples, with a nominal integration time of 1.45 msec.

Signal to Noise

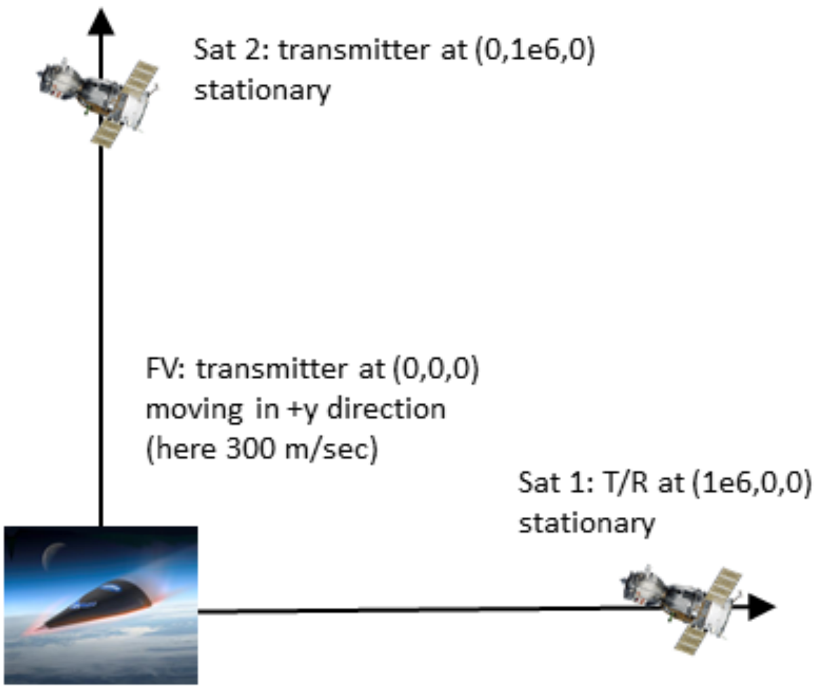


Figure 29. Nominal SNR study geometry.

In the following plots (Figure 30Figure 31), the vertical axis is velocity (in m/sec) and the horizontal axis is range (in meters). We see the ambiguity function integration for 1.45 msec (Figure 30) and 14.5 msec (Figure 31).

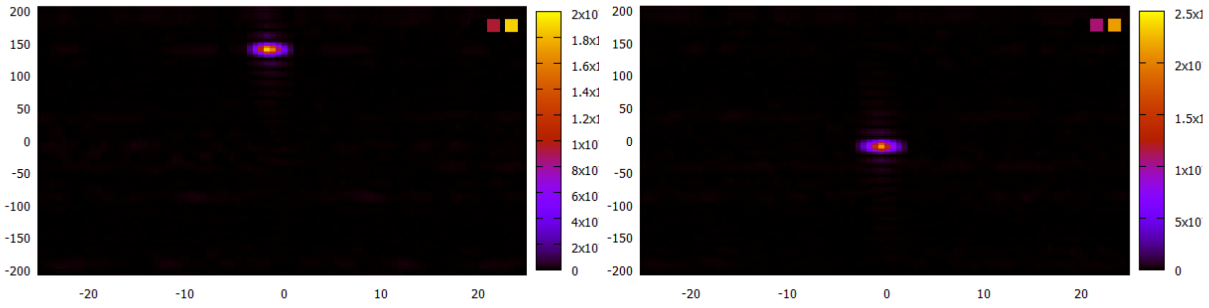


Figure 30. Bistatic (left) and monostatic (right) ambiguity function integration, 1.45 msec.

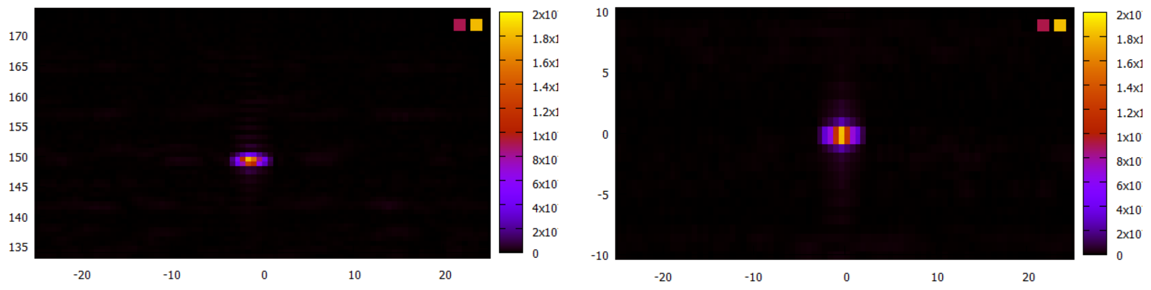


Figure 31. Bistatic (left) and monostatic (right) ambiguity function integration, 14.5

As expected, we have 10x better velocity resolution with the longer integration time. In Figure 30Figure 31, we see a Doppler frequency shift for the bistatic geometry (as the flight vehicle is moving to the transmitter above it), but none for the monostatic geometry (as the flight vehicle motion is perpendicular to its view).

Since we are subsampling the time, we can see some aliasing (Figure 32), here for 10001 time steps over 14.5 msec. We can remove the aliasing from our ambiguity window by slightly changing the number of time steps (e.g., 9876 time steps over the same interval).

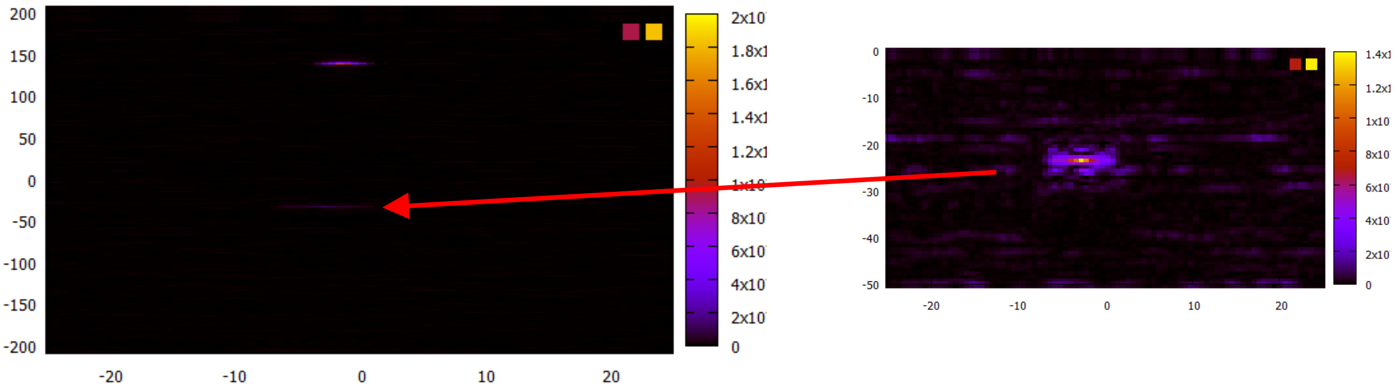


Figure 32. Some aliasing is seen at about -30 m/sec for the bistatic case with an integration time of 14.5 msec and 10001 time steps.

4.3 Effect of noise

The images of the signals in Figure 30Figure 31 do not include noise. In this section we add in additive white Gaussian noise (AWGN). Numerically, we do it with the coding below:

```
signal=0.

       do 130 i=1,3,2

       mmodulo=(t-tret(i)-tret(2))/deltat

       if(t-tret(i)-tret(2).lt.0.) mmodulo=mmodulo-1

       codetime=(t-tret(i)-tret(2))-float(mmodulo)*deltat

      mbit=1+codetime/delt

        naddsig(i)=mgcode(i,mbit)

        if(naddsig(i).eq.0) naddsig(i)=-1

c

c  we're actually adding field, not power, amplitudes

c

         sign=rand()

  126    z1=(-2.*log(rand()))**.5*cos(2.*xpi*sign)
```

```
         if(abs(z1).gt.8) goto 126
      pmult=25.
      rsecond=rbi
      if(i.eq.1) rsecond=rmono
      powerr=power*rcs*antarea**2/(4.*xpi*xlambda**2)/rfv**2/rsecond**2
      energyr=powerr*pmult*band
      ampsig=sqrt(energyr)*dt
      pnoise=fn*xkb*temp*band
      energyn=pnoise
      signoise=sqrt(dt)*sqrt(energyn)/sqrt(2.)
      if(i.ne.1) signoise=0.

      sig=naddsig(i)*ampsig
      signal=signal+sig*cos(2.*xpi*(t-tret(i)-tret(2))
     1     *freqsig)+signoise*z1
  130 continue
```

We define the code signal elements in Figure 33.

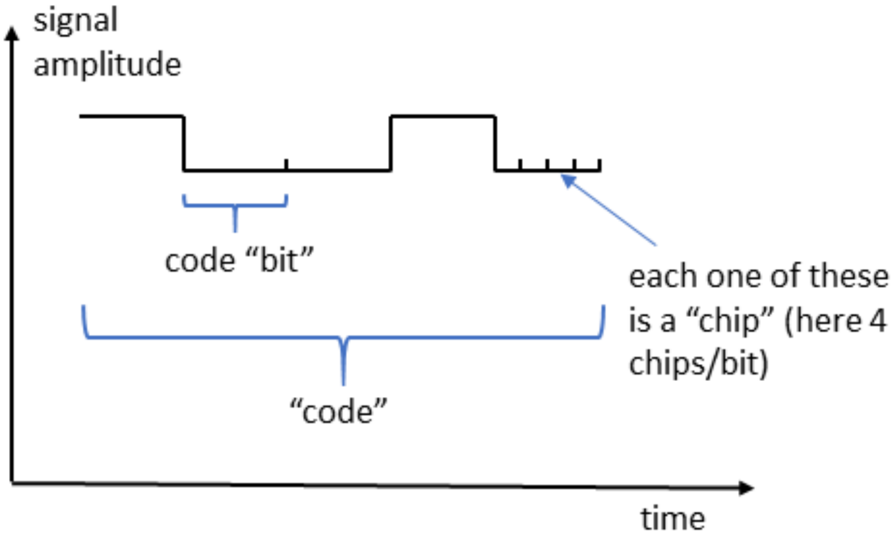


Figure 33. Elements of the code waveform.

We assume we are sampling the code and the noise energies at a chip rate (i.e., subdividing the bits of a code). The energy of the chip and the energy of the noise during the chip are then:

$$E_{chip} = P_r\ \Delta t_{chip} \tag{47}$$

$$E_{noise} = f k_B T_r B \Delta t_{chip} = f k_B T_r \tag{48}$$

where we assume the bandwidth B is transformed limited.

The following equations are used to predict scalings of the signal and noise amplitudes for coherent integration. Using normalized signal and noise amplitudes

$$A_{signal,\, chip} = \sqrt{E_{chip}} \tag{49}$$

$$A_{noise,\, chip} = \sqrt{E_{noise}} \tag{50}$$

we have these amplitudes for time durations of Δt:

$$A_{signal,\, \Delta t} = \sqrt{E_{bit}}\frac{\Delta t}{\Delta t_{chip}} \tag{51}$$

$$A_{noise, \Delta t} = \sqrt{E_{noise}}\sqrt{\frac{\Delta t}{\Delta t_{chip}}}\ . \tag{52}$$

Since the bit $SNR = \frac{Power_{signal}}{Power_{noise}} = \left(\frac{Amplitude_{code}}{\sigma_{noise}}\right)^2$ where $Amplitude_{code}$ is the rms amplitude of the code, we confirm that the time integrated SNR scales linearly with Δt. We can write the peak amplitude of the time integrated signal and the noise as

$$Amplitude_{signal} = N_{code\ cycles} N_{bits\ per\ code} N_{chips\ per\ bit} \sqrt{2} Amplitude_{code} \tag{53}$$

$$Amplitude_{peak\ noise} = \sqrt{N_{code\ cycles} N_{bits\ per\ code} N_{chips\ per\ bit}}\ F \sqrt{Power_{noise}} \tag{54}$$

where F is a multiplicative factor between rms and peak noise that we'll discuss more later. In Figure 34 we confirm the $N_{code\ cycles} N_{bits\ per\ code} N_{chips\ per\ bit}$ and $\sqrt{N_{code\ cycles} N_{bits\ per\ code} N_{chips\ per\ bit}}$ scalings of the integrated signal and noise respectively. The peak amplitude of the noise can be written as

$$Amplitude_{peak\ noise} = \sqrt{N_{code\ cycles} N_{bits\ per\ code} N_{chips\ per\ bit}} \left(F \frac{\sigma_{noise}}{Amplitude_{signal}} Amplitude_{signal}\right) \tag{55}$$

Signal to Noise

(the ratio $\frac{\sigma_{noise}}{Amplitude_{code}}$ is about 20 in these plots).

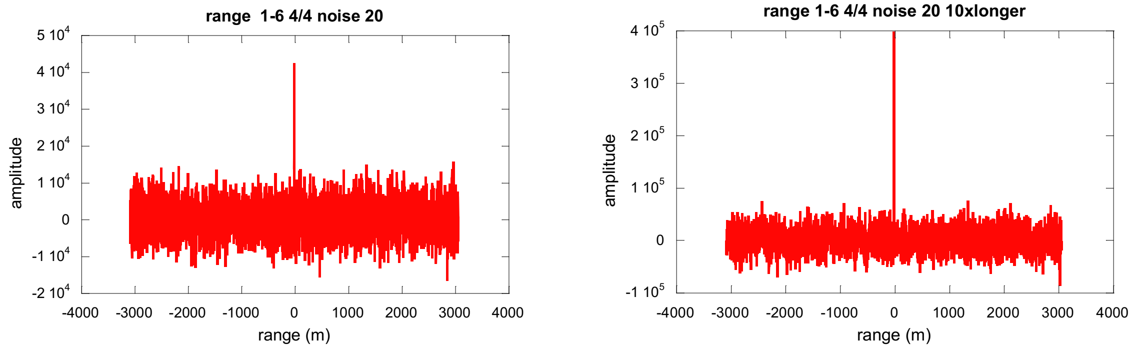


Figure 34. Signal and noise amplitudes for two integration times differing by a factor of 10.

4.4 Relation between peak noise and peak amplitude

In Figure 35, we see a generic signal with a noise background, with a definition of a noise floor.

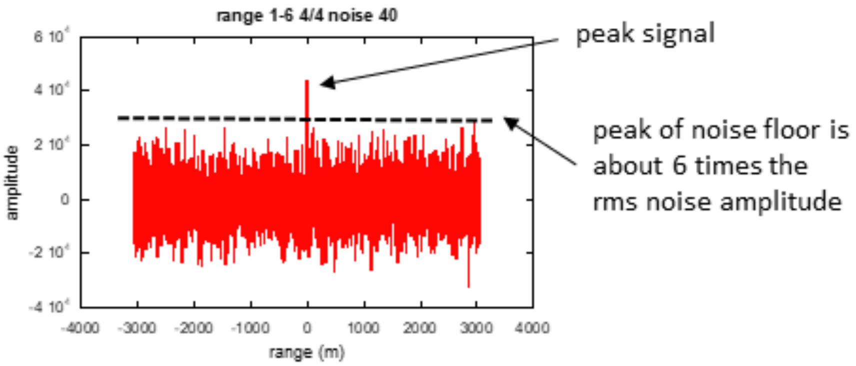


Figure 35. Relationship between the peak signal and the noise floor.

The ratio of the amplitude of the signal over the peak noise

$$\frac{Amplitude_{signal}}{Amplitude_{peak\ noise}} = \frac{\sqrt{N_{code\ cycles} N_{bits\ per\ code}}}{\frac{F}{\sqrt{2}} \frac{\sigma_{noise}}{\sqrt{Power_{signal}}}} \qquad 56$$

needs to be > 1, with F about 6 and we recognize

$$\frac{\sigma_{noise}}{\sqrt{Power_{signal}}} = 1/\sqrt{SNR} \qquad 57$$

Inverting this, we find the minimum requirement is

$$SNR \geq \frac{F^2/2}{N_{code\ cycles} N_{bits\ per\ code}}$$

58

The factor F is our confidence factor that any peak in the noise will be below the signal for all ranges within our range window. Specifically, consider the normal distribution in Figure 36.

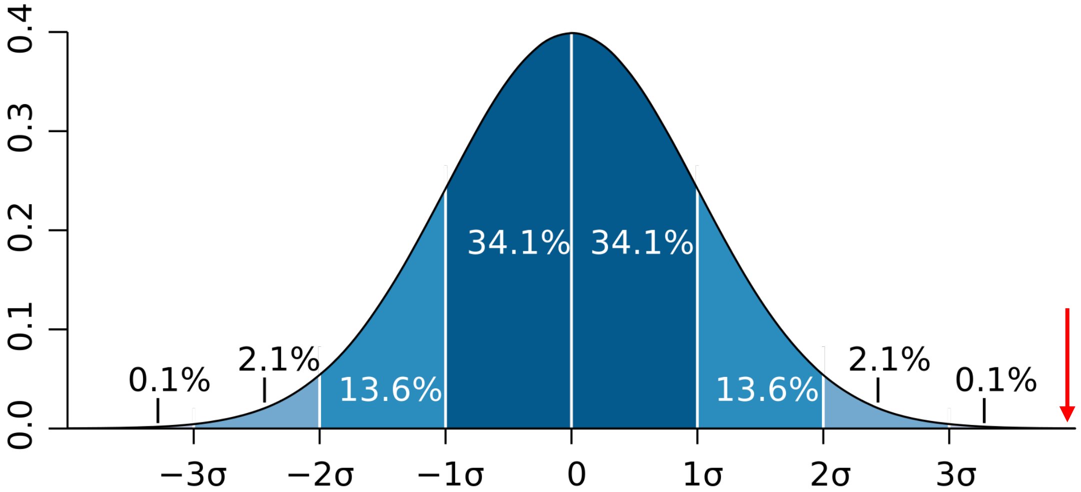


Figure 36. This is the distribution of noise amplitudes at any location. We're concerned that a very unlikely amplitude (location of the arrow) may appear within our range window.

For relatively large Fs , the probability that an event will be greater than $F\sigma$ (where σ is the rms width of the distribution) for a single trial is approximately $P_{F\sigma} = \frac{e^{-F^2/2}}{2F\sqrt{\pi/2}}$. The probability of the peak within $F\sigma$ is then:

$$P_{peak\ within\ F\sigma} = 1 - (1 - P_{F\sigma})^N$$

59

We can characterize these probabilities in the following table:

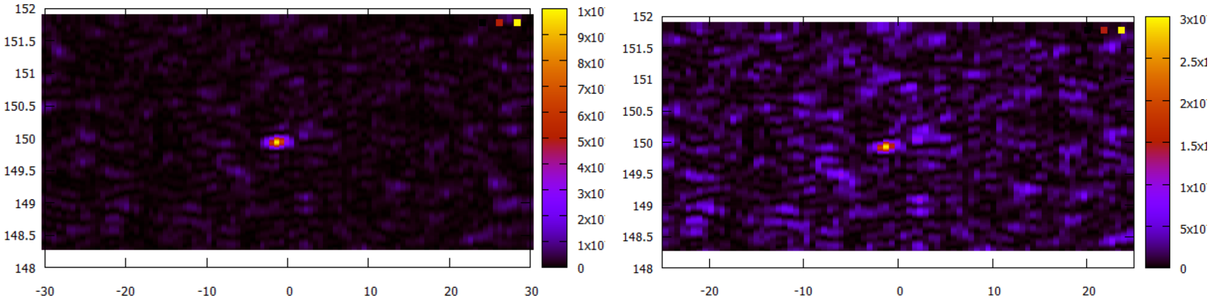
Table 3. Noise Probability Table

# of trials	F for $P_{80\%}$	F for $P_{90\%}$	F for $P_{95\%}$
10000	4.1	4.3	4.4
100000	4.6	4.7	4.9
1000000	5.1	5.2	5.3
1.00E+07	5.5	5.6	5.7
1.00E+08	5.9	6.0	6.1

We see we have > 95% probability of not having a single random event larger than 6σ, even for numbers of trials approaching 100M. This factor of 6 leads to our estimate for the minimum time-integrated SNR (i.e., E_w/N_0) of about 18.

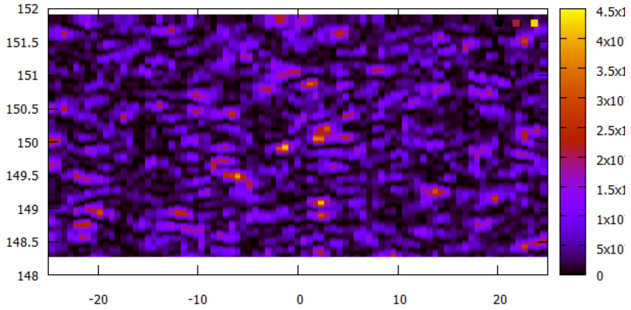
4.5 Ambiguity function integration to validate SNR estimate

In Figure 37, we coherently integrating a signal for a time $t = N_{steps}\Delta t = N_{codes}N_{bits/code}N_{chip/bit}\Delta t_{chip}$ for an integration time t =0.145 sec (Δf_{res} is about 6.9 Hz and Δv_{res} is about 0.3 m/sec) and with differing amounts of transmitter power.



200 kW (significantly above noise)

50 kW (still pretty clear)



12.5 kW (noise threshold)

Figure 37. We are decreasing the transmit power from 200 kW to 12.5 kW to determine the threshold where we can no longer clearly determine the peak of the ambiguity function. For this case (1000 km ranges, 6.9 GHz carrier frequency, 1-m^2 antenna areas, and a 1-m^2 radar cross section), the minimum transmitter power is about 12.5 kW.

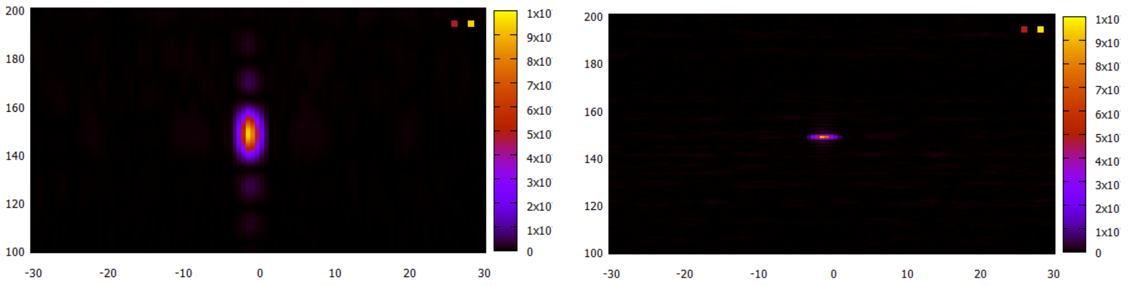
For the case in Figure 37, (1000 km ranges, 6.9 GHz carrier frequency, 1-m^2 antenna areas, and a 1-m^2 radar cross section), the minimum transmitter power is about 12.5 kW. This corresponds to an $E_w = P_r\, t = 5.3 \times 10^{-19}$ J and $N_0 = f k_B T_r = 8.28 \times 10^{-21}$ J (with a noise figure $f = 2$), leading to a minimum E_w/N_0 of

$$\frac{E_w}{N_0} \geq 10 \tag{60}$$

We would have expected a factor of 18 from the earlier analysis ($= F^2/2$). This ratio corresponds to a factor F of about 4.5, which is probably due to the lower number of range intervals we are including the ambiguity calculation (which would be enabled by an ongoing estimate on the location of the flight vehicle from the extended Kalman filter.)

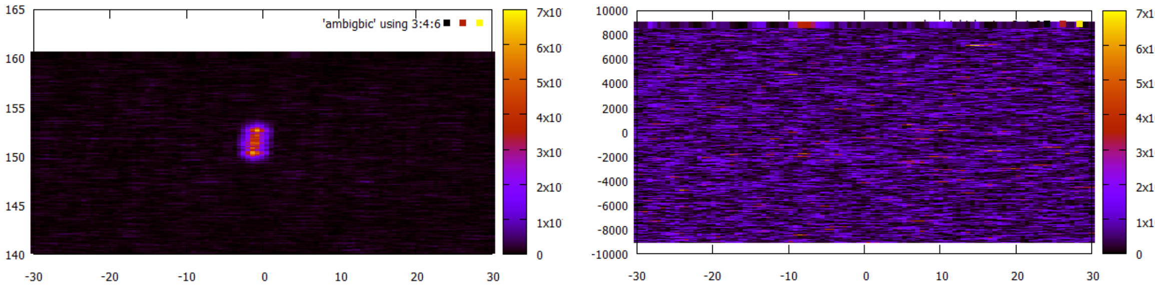
4.6 Effect of acceleration on the SNR

The effect of acceleration on the SNR arises because acceleration leads to a smearing along velocity in the ambiguity function, as shown in Figure 38.



1.45 msec (no noise)

14.5 msec (no noise)



145 msec (no noise)

145 msec, 50 kW and standard noise

Figure 38. Top and bottom left: increasing integration times without noise, with 3 g's of vertical acceleration. The velocity resolution improves at first with increasing time integration, but eventually the velocity gets smeared out as the acceleration times the integration time exceeds the nominal velocity resolution. The image in the bottom left shows that the smeared image is more sensitive to noise (compare to the easily determined case in Fig. 16 with 50 kW of transmit power).

The smearing of velocities due to the acceleration makes the image more sensitive to noise. The native velocity resolution and the smeared velocity magnitude are:

$$\Delta v_{resolution} = \frac{c}{f t_{integrated}}$$

61

$$\Delta v_{smear} = \ddot{R} t_{integrated}$$

62

This leads to an increased SNR requirement if acceleration dominates of

$$\frac{E_w}{N_0} \geq \left. \frac{E_w}{N_0} \right|_{min} \frac{\Delta v_{smear}}{v_{res}} \geq 10 \frac{\Delta v_{smear}}{v_{res}}$$

63

For short enough times, the original expression ignoring acceleration is fine. Adding the two constraints in quadrature, we find

Signal to Noise

$$\frac{E_w}{N_0} \geq 10\sqrt{1+\left(\frac{\ddot{R}t_{int}^2 f_{carrier}}{c}\right)^2} \tag{64}$$

Since the acceleration and received power can vary over these integration periods, we should more properly represent these terms with integrations over time. Note that the required transmitted power is minimized for an integration time of $t_{int,\min power} = \sqrt{\frac{c}{\ddot{R}f_{carrier}}}$, and the minimum power requirement is then

$$P_r \geq 10\sqrt{2}N_0\sqrt{\frac{\ddot{R}f_{carrier}}{c}} \tag{65}$$

If the received power is lower than that, it does not help close the link budget. That time can still be part of an interval that closes the link budget as long as the integrated received power satisfies

$$\int_0^{t_{int}} P_r\,dt > 10N_0\left(1+\left(\frac{t_{int}f_{carrier}}{c}\int_0^{t_{int}}\ddot{R}dt\right)^2\right)^{1/2} . \tag{66}$$

As described in the Geometric and Signal Simulation section, $\ddot{R}$ has contributions both from the actual flight vehicle acceleration $\vec{A}$ and from the geometric motion of the flight vehicle and the tracking satellites. Combined, the total effective range acceleration as a function of the transmitter location, the receiver location, the actual flight-vehicle velocities relative to the transmitter and receiver $\vec{V_t}$ and $\vec{V_r}$, and flight vehicle acceleration $\vec{A}$ is:

$$\ddot{R} = \frac{\vec{R}_t\cdot\vec{A}}{R_t} + \frac{1}{R_t}\left(V_t^2 - \dot{R}_t^2\right) + \frac{\vec{R}_r\cdot\vec{A}}{R_r} + \frac{1}{R_r}\left(V_r^2 - \dot{R}_r^2\right) \tag{67}$$

Accurate knowledge of the transmitter and receiver locations and an estimate of the flight vehicle's position and velocity can reduce the uncertainty of $\ddot{R}$, but it will never be reduced below $\frac{\vec{R}_t\cdot\vec{A}}{R_t} + \frac{\vec{R}_r\cdot\vec{A}}{R_r}$. The uncertainty in the flight vehicle's position and velocity can be used to estimate the magnitude of the other terms using a first-order Taylor expansion.

Accurate knowledge of the transmitter and receiver locations and an estimate of the flight vehicle's position and velocity can reduce the uncertainty of $\ddot{R}$, but it will never be reduced below $\frac{\vec{R}_t\cdot\vec{A}}{R_t} + \frac{\vec{R}_r\cdot\vec{A}}{R_r}$. The uncertainty in the flight vehicle's position and velocity can be used to estimate the other terms. With a nominal relative velocity of 10 km/sec, a nominal impact parameter of 500 km, tracking position uncertainties of 100 m/sec and 1 km, the maximum geometric acceleration is about 250 m/sec^2, with a tracking estimate error of about 5 m/sec^2. Thus, it is likely the uncertainty in $\ddot{R}$ will be dominated by the actual flight vehicle acceleration $\vec{A}$.

We can use Equations 66 and 67 along with the Kalman filter tracking results to work through the constellation system requirements. For nominal parameters (1.5-m antenna radii, 10-GHz carrier, RCS of about 0.1 m^2, and 10-kW transmitter power, and ignoring losses in Equation 36), the received power is about 4.4x10^{-18} W at a range of 1000 km and N_0 is about 8.3x10^{-21} J. Without acceleration, the link budget closes in about 0.019 seconds for these parameters.

With acceleration, the required minimum power required to close the link budget is shown in the plots below, as function of flight vehicle acceleration in g's. The nominal received power from the paragraph above (4.4x10^{-18} W) can close the link budget for accelerations up to about 4 g's, but a higher received power is needed for higher accelerations, which means some combination of higher transmitter power, larger antenna sizes, or smaller maximum ranges.

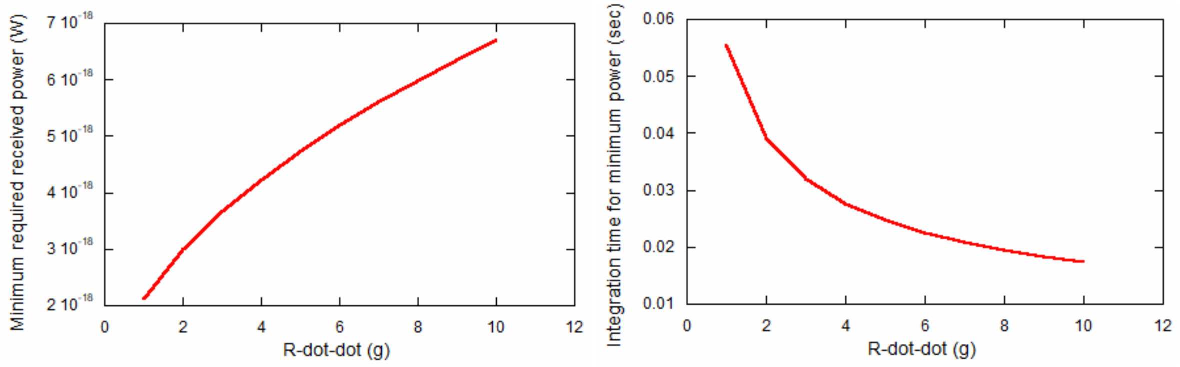


Figure 39--Plots of the minimum received power required to close the link budget as a function of vehicle acceleration (in g's) (left) and corresponding integration time (right).

These plots can be used to estimate system requirements with the Kalman filtering tracking, including constellation size. For example, to track a flight vehicle with accelerations up to 10 g's, a minimum received power of 6.7x10^{-18} W is needed. Say a maximum transmitter power of only 5 kW and a maximum antenna radius of 0.5 m are possible, and that the radar is only 25% efficient. With these constraints, and keeping the carrier frequency and radar cross section the same, Equation 36 tells us that the maximum range for closing the link budget is only 560 km. The Kalman filter tracking results will tell us how often we need to close the link budget, which, with the geometric satellite results, will in turn tell us how many satellites will be needed in our constellation to achieve that range with the frequency required by the Kalman filter tracking.

Note that although the ambiguity function we used is a function of only $R, \dot{R}$, it is theoretically possible to search as many range derivatives as we desire. However, we also observe that regardless of how many derivatives we search, the next derivative down will always lead to a similar smearing effect, but now scaling with a higher power of the integration time. Noting that the 2D ambiguity function is a slice of the 3D ambiguity function, and poor prediction of $\ddot{R}$ simply moves the signal we are searching for away from the slice plane, we can redefine our ambiguity function as a *projection* of the likely $\ddot{R}$ range onto the slice plane, at the cost of significantly more computational time.

5 Geolocation

The geolocation of the vehicles uses the measurements from the satellites, which are range $\mathbf{r}$ and range-rate-of-change $\dot{\mathbf{r}}$ for each bistatic or monostatic pair. There are two approaches for tracking the vehicle. The first option, is to take each set of measurements that occur within some predefined time of each other and construct where the vehicle was at that time. A filtering algorithm is then used to connect all the measured positions with an estimate equation of state to predict where the vehicle is going to go. The other option is to use each $(\mathbf{r}, \dot{\mathbf{r}})$ pair to update the filter directly. Both of these options were considered and will be discussed below.

5.1 Equation of State

Before getting into the geolocation algorithms, let's define the equation of state that is being assumed for the vehicle. In defining the equation of state many of the coordinate systems, and parameters that are used in the geolocation algorithms will be discussed.

The vehicle's path is assumed to be a great circle. A great circle is defined as the largest circle that can be drawn on any given sphere [18]. A vehicle traveling in a great circle can be described with orbital mechanics. The parameters for orbital mechanics are

- a the semi-major axis,
- i the inclination,
- Ω the longitude of the ascending node,
- ν the true anomaly,
- e the eccentricity,
- ω the argument of periapsis.

Since, the vehicle is assumed to be traveling in a circle, e and ω are assumed to be zero. This implies that a is the radius of the orbit. The gravitational parameter μ, which is normally defined by GM_0 where G is Newton's gravitational constant and M_0 is the mass of the object that is being orbited around, is not assumed constant since the trajectory of the vehicle, although assumed to be a great circle, is not in orbit around the earth.

The mean anomaly M is defined as

$$\begin{aligned} M &= E_a - e * \sin(E_a), \\ \text{where } E_a &= 2\text{arctan}\left[\tan(\nu/2)/\sqrt{\frac{1+e}{1-e}}\right], \end{aligned} \tag{68}$$

since $e = 0$, $M = \nu$. If the orbital parameters are known at a time t_0 $(a_0, i_0, \Omega_0, e_0, \omega_0, M_0)$ then at some time difference later Δt the new orbital parameters will be $\left(a_0, i_0, \Omega_0, e_0, \omega_0, M_0 + \Delta t\sqrt{\mu/a^3}\right)$. The change in the orbital parameters is only in the mean anomaly and depends on the gravitational parameter and the semi-major axis.

With all the assumptions made the list of parameters that are free to change in the algorithms below are (a, i, Ω, μ, M), knowing that ω is assumed to be zero, gives the necessary 6 parameters to map out the position and velocity of the vehicle.

5.1.1 *Converting to and from Cartesian coordinates*

The position and velocity of the satellites are given in Cartesian coordinates. Cartesian coordinates does not have any variables that are angles, which makes them easier to work with. The algorithms below calculate the position and velocity of the vehicle in Cartesian coordinates. With the equation of state being in orbital parameters, a mapping is needed to convert between the two. The functions used can be found in the Vehicle Tracking DI repository under geolocation/cart_kep_conversions.py.

5.2 Multi-static Radar Position Algorithm

Using the two step weighted-least squares algorithm presented in [19], the multi-static radar position algorithm, MSRPA, is able to predict the location of a vehicle to sub $(10\,\mathrm{m}, 10\,\mathrm{m/s})$, as shown in Figure 40. The details of the equations are in [19]. For here, it would suffice, to say the main equations are

$$\begin{aligned}
\hat{\theta}_{1,2} &= \left(G_{1,2}^T W_{1,2} G_{1,2}\right)^{-1} G_{1,2}^T W_{1,2} h_{1,2}, \\
W_1 &= [B_1 Q B_1^T]^{-1}, \\
cov(\hat{\theta}_1) &= (G_1^T W_1 G_1)^{-1}, \\
W_2 &= \left[B_2 cov(\hat{\theta}_1) B_2^T\right]^{-1}, \\
\hat{\theta}_2 &= (\mathbf{r}, \dot{\mathbf{r}}), \\
cov(\hat{\theta}_2) &= [(B_1^{-1} G_1 B_2^{-1} G_2) Q^{-1} (B_1^{-1} G_1 B_2^{-1} G_2)]^{-1},
\end{aligned} \tag{69}$$

where G_1, B_1, and h_1 are dependent on measured $\mathbf{r}$ and $\dot{\mathbf{r}}$, while G_2, B_2, and h_2, are dependent on $\hat{\theta}_1$. Q is the covariance matrix comprised of the uncertainties on $\mathbf{r}_i$, and $\dot{\boldsymbol{r}}_i$ for the ith bistatic range and their correlations. At the end of the day the results are $\hat{\theta}_2$ and its corresponding covariance matrix. The shape of Q and B_1 is $(2MN, 2MN)$, where M is the number of receivers and N is the number of transmitters, the shape of $\hat{\theta}_1$ is $(6 + 2M, 1)$, where the extra $2M$ rows come from the prediction of the range and range-rate-of-change between the vehicle and the transmitters based on the first step of the weighted least-squares algorithm.

The caveat for his approach, is that the because of all the matrix multiplication that is dependent on matrices that have some combination of number of receivers and number of transmitters, if a receiver-transmitter pair was not seen at the measured time then the results are void. An example of this is given in

Table 4. An example table showing three consecutive time readings from an earlier version of the satellite measurement simulation. Measurements were given at nominal 3 s intervals, but not necessary in time.

Time (s)	Sat 1	Sat 2	**r** (km)	$\dot{\mathbf{r}}$ (km/s)
93	172	172	2193.2	-12.9
93	172	204	1571.5	-7.2
93	172	642	1537.1	-6.9
93	172	674	2269.3	-0.8

93	204	204	949.8	-1.4
93	204	642	915.4	-1.1
93	204	674	1647.6	5.0
93	642	642	881.0	-0.8
93	642	674	1613.2	5.2
93	674	674	2345.4	11.3
94	**172**	**610**	**2267.8**	**-12.9**
94	**204**	**610**	**1651.9**	**-7.1**
94	**610**	**610**	**2355.3**	**-13.0**
94	**610**	**642**	**1617.9**	**-6.7**
94	**610**	**674**	**2356.0**	**-0.8**

In Table 4 at time 93 s all the combinations of Sat 1 and Sat 2 are seen, while at 94 s not all combinations are seen. For example, at 94 s (Sat 1, Sat 2) = (172, 610) is observed but no other combination with satellite number 172 is observed at that time. To correct this, one could use a filtering algorithm to predict where the vehicle is located at 94 s, and fill in the gap, but because the measured $\mathbf{r}$ and $\dot{\mathbf{r}}$ are averaged over a reading, this becomes tricky and just using the measured $\mathbf{r}$ and $\dot{\mathbf{r}}$ would be easier especially when the bistatic measurements will come in at semi-random times.

5.3 Unscented Kalman Filter with Range, Range-Rate-Of-Change Directly

An unscented Kalman Filter (UKF) will be used to track a vehicle with the orbital equation of state. The UKF will get updated with the bistatic measurements. The UKF was used because the problem is non-linear, and instead of using Jacobians it transforms random points selected in the state space to the measured space, and then calculates the averages to determine how to update the state vector instead [17]. Jacobians were going to be hard to use since the state vector is in Cartesian coordinates but converts to orbital coordinates to predict the next location, and the measured are in $\mathbf{r}$ and $\dot{\mathbf{r}}$.

The UKF algorithm (filterpy.kalman.UnscentedKalmanFilter) from the PYTHON package FilterPy was used [16[17]. The success of the UKF is choosing the correct random points in the state space. The random points where chosen based off the weights calculated by filterpy.kalman.MerweScaledSigmaPoints$(n, \alpha, \beta, \kappa)$ function with $n = 6$, $\alpha = 0.5$, $\beta = 2$, and $\kappa = -3$. The value of n is equal to the number of variables in the state vector, α is correlated to how tight the points are distributed, $\beta = 2$ is good if the state vector is represented by a Gaussian, and $\kappa = 3 - n$.

The UKF algorithm uses two functions, one to describe how the state vector changes as a function of time and one that maps the state vector to the measured vector. The function that describes how the state vector changes as a function of time is described in Section 2, where the state vector, in Cartesian coordinates, is transformed to orbital parameters, M is increased by Δt, and then the orbital parameters are transformed back into Cartesian coordinates. The function that maps the state vector to the measured vector follows the following steps:

1. Get list, position and velocity of satellites for each bistatic reading that is being considered (in Cartesian coordinates),
2. Get the random state vector points (in Cartesian coordinates),
3. Calculate the $\mathbf{r}$ and $\dot{\mathbf{r}}$ for each bistatic satellite pair with each random state vector.

These “random measured points” will be compared to the true measured points in the unscented transform. Results from the UKF are shown in Figure 41 and Figure 42. The UKF method requires an initial guess for the state vectors. The results figures use the truth information from the first reading as the state vector with initial errors on the order of 10 km and 0.1 km/s for each Cartesian coordinate. This is to show how fast the UKF converges to the true values. In practice, the initial state vectors and errors will most likely come from another system, although if there is a time with “complete” bistatic readings then the MSRPA can be used to get the initial guess. Figure 42 shows that the resulting errors converge quickly, and are below a km in resolution.

The UKF method follows the segmented trajectory extremely well. Plotting in orbital parameters, shows that changes in i, Ω, μ, and/or M from linear could be used as an indication the trajectory is changing course. The resulting χ^2, that is calculated from

$$\chi^2 = \Delta\mathbf{x}^T \mathcal{M} \Delta\mathbf{x}, \tag{70}$$

where $\Delta\mathbf{x}$ is the difference between true and reconstructed variables and $\mathcal{M}$ is the UKF output covariance matrix, shows an order of magnitude difference between the great circle and the segmented trajectories.

5.4 Comparing Different Satellite Configurations

Pull distributions were generated to compare satellite configurations. The pull term is defined as $pull = (true - reco)/\sigma_{reco}$ and should have a Gaussian distribution with a mean of 0 and a standard deviation equal to 1 if σ_{reco} incorporates all, and the right size, of model uncertainties. Four different satellite configurations were used to demonstrate how optimizing the process noise matrix in the Kalman Filter for one constellation effects the pull distribution in other constellations, see Figure 44. Comparing the pull distributions for (a) 32x32, (b) 25x25, and (c) 20x20 satellite constellations where the process noise matrix was optimize for the 32x32 constellation. The event window used for these plots was 1 s. The four constellations were all Walker Constellations with the same inclination, but with different number of orbital planes: 32, 25, 20, and 10 respectively. The number of satellites in each plane is the same as the total number of planes.

Just by looking at the true and reconstructed trajectories in Figure 43 as a function of time for the great circle simulation, it is obvious that only having 100 satellites (10x10) is not enough. Figure 44 shows that the pull distributions get worse when reducing the number of satellites assuming the optimization for the 32x32 constellation is correct. Obviously once the number of satellites is known, a final optimization procedure will need to be done.

Figure 45 compares two different “event building” techniques. The left plot assumes that the Kalman Filter updates every one second while the right plot assumes the Kalman Filter updates on each received measurement. Both plots are for a 32x32 constellation and each were optimized to generate a pull distribution with a mean of about zero and a standard deviation of about one. Updating the Kalman Filter with each measurement individual produces a lower uncertainty on the Cartesian coordinates and approaches a constant (assuming the tradeoff uncertainties are high).

Figure 46 shows the trajectories, comparison to truth, and pull distributions for the segmented simulation. Because of the change in direction, it is harder to get a pull distribution in both position and velocity with the ideal distribution. But with an assumption the position distribution can be close to a mean of zero and a standard deviation of one, with the velocity having a mean of about zero and a standard deviation of two.

All of this shows that the Unscented Kalman Filter algorithm for geolocation is able to accurately track the vehicle and that for global coverage, about 400 satellites minimum is needed.

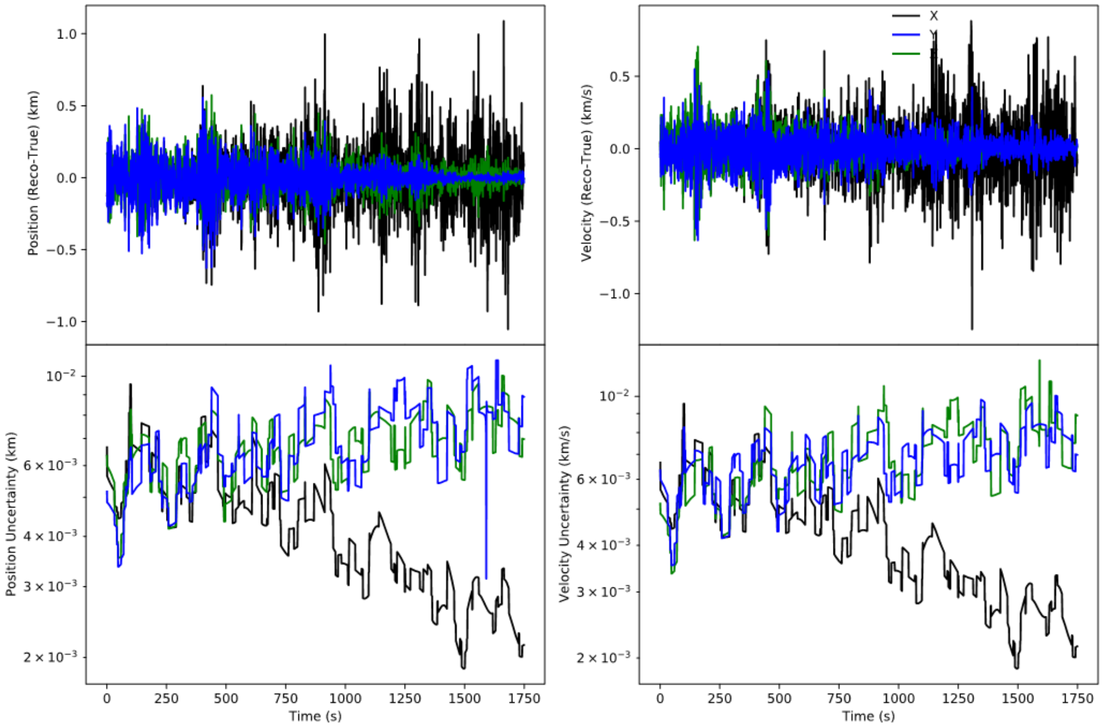


Figure 40. (top row) The differences between true and reconstructed as well as (bottom row) the resulting uncertainties using the multi-static radar position algorithm. Cartesian coordinates are used where black is x, blue is y, and green is z. The left column shows the results on the position and the right column shows the results for the velocity.

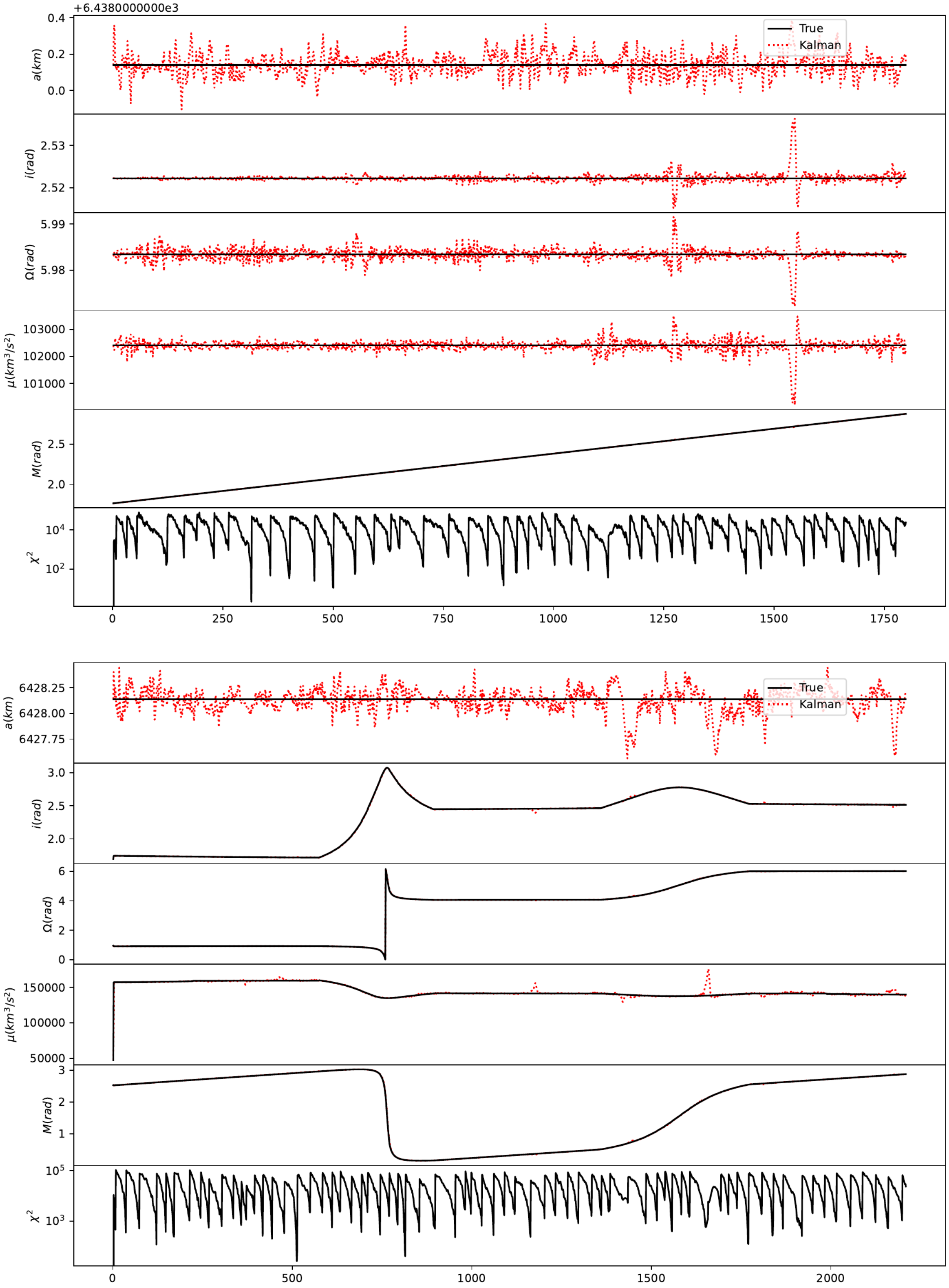


Figure 41: The results of the UKF for (top) a great circle and (b) a segmented trajectories. Each plot shows the true (solid) and reconstructed (dotted) trajectories as a function of time. From top to bottom in the plot are the orbital parameters (a, i, Ω, μ, M) and the χ^2 comparison between the true and reconstructed.

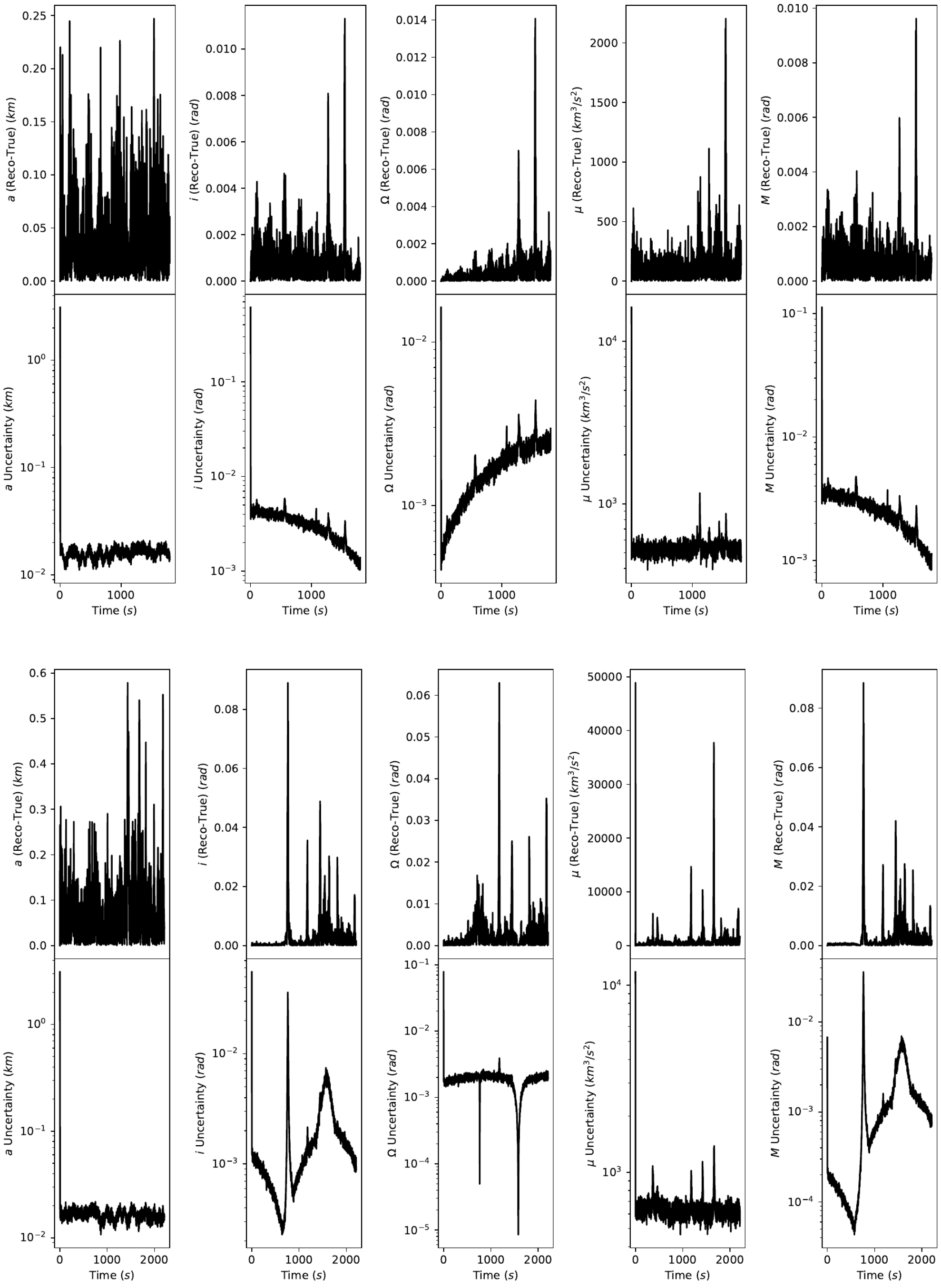


Figure 42: The results of the UKF for (top) a great circle and (bottom) a segmented trajectories. Each plot shows (top) the difference between the true and reconstructed trajectories and (bottom) the resulting uncertainties as a function of time. From left to right the curves are for the orbital parameters (a, i, Ω, μ, M)

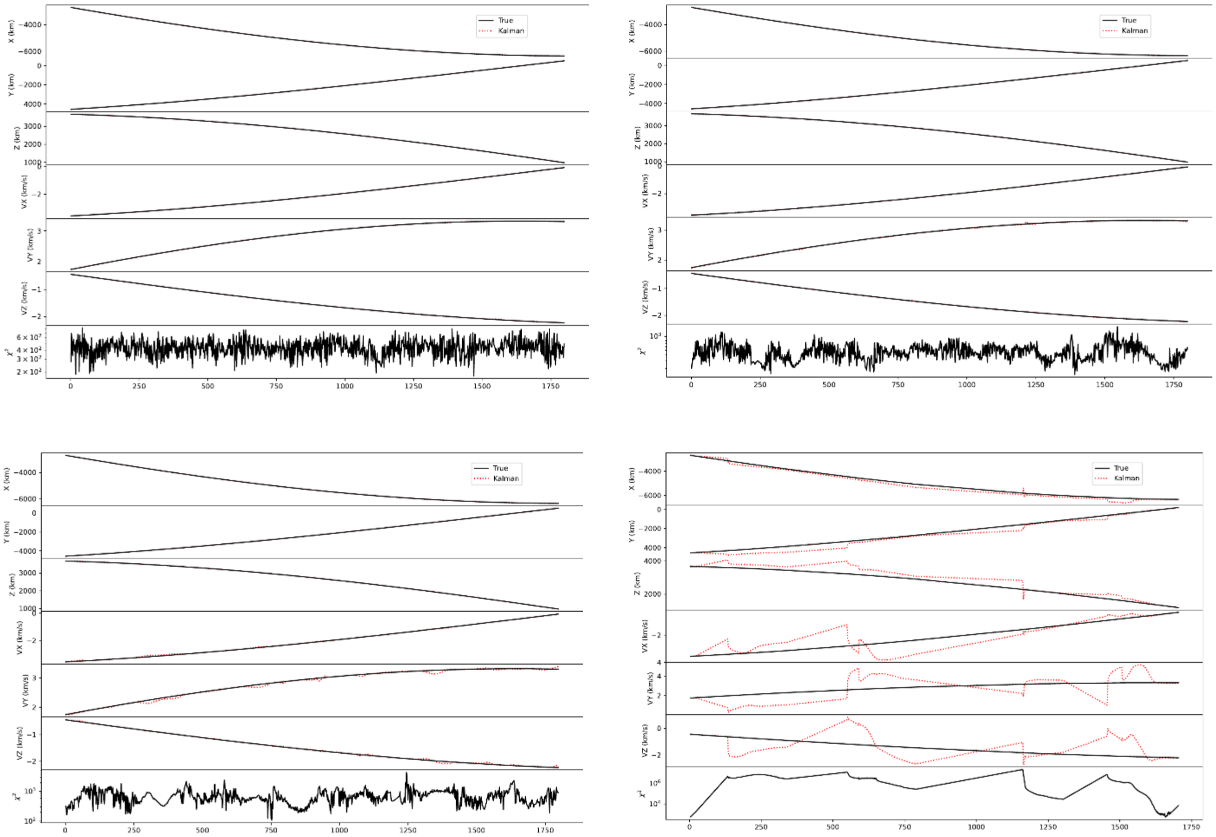


Figure 43 Comparing True and reconstructed trajectories with four different constellations: (a) 33x32, (b) 25x25, (c) 20x20, and (d) 10x10. The more orbital planes with more satellites the better the reconstruction.

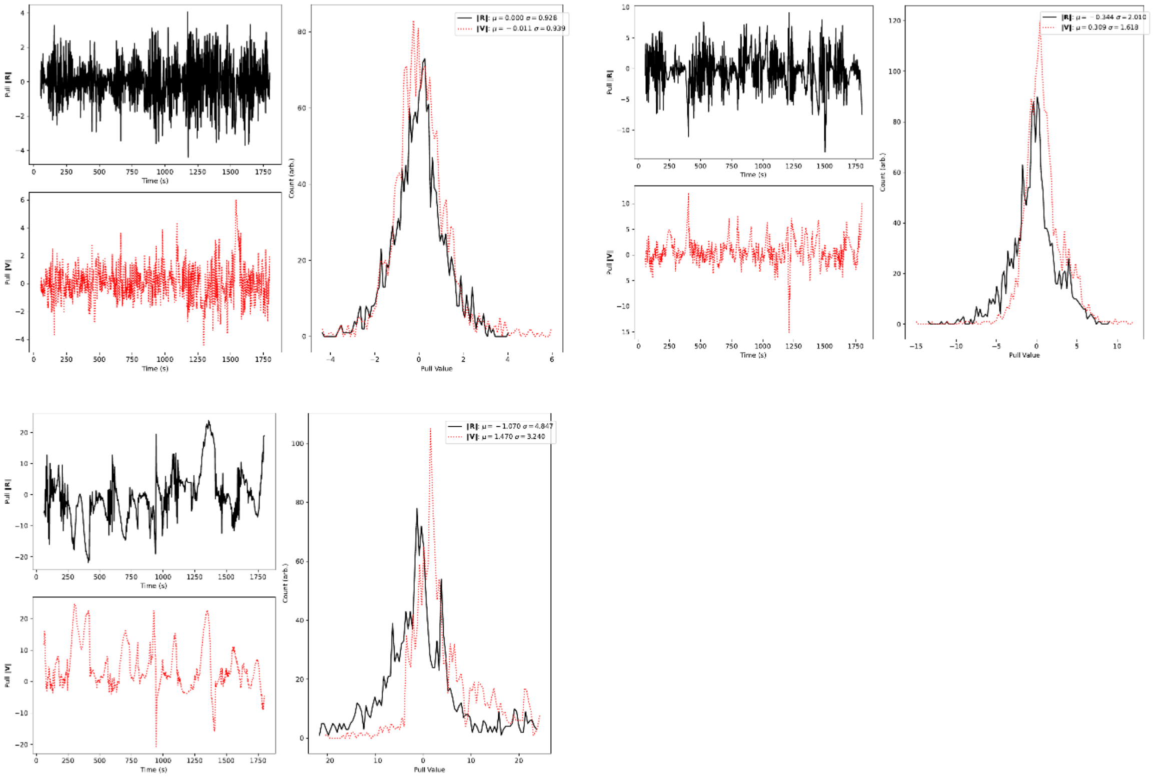


Figure 44. Comparing the pull distributions for (a) 32x32, (b) 25x25, and (c) 20x20 satellite constellations where the process noise matrix was optimize for the 32x32 constellation. The event window used for these plots was 1 s.

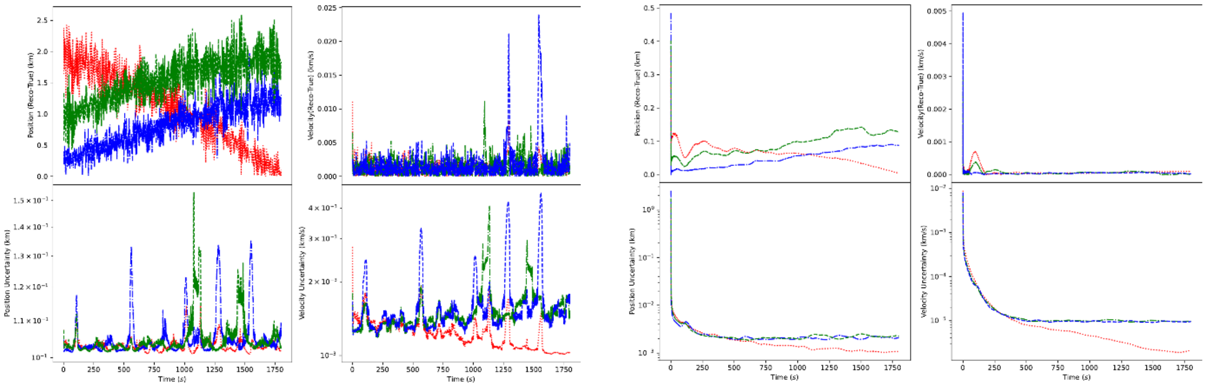


Figure 45. Comparing the differences to true and uncertainties in Cartesian coordinates for events windows (a) 1s and (b) 0s. As seen, having no defined event window, the algorithm converges to a constant error. The time the algorithm takes to converge is dependent on how big the uncertainty is on the tip off. Red = x, Green = y, Blue = z

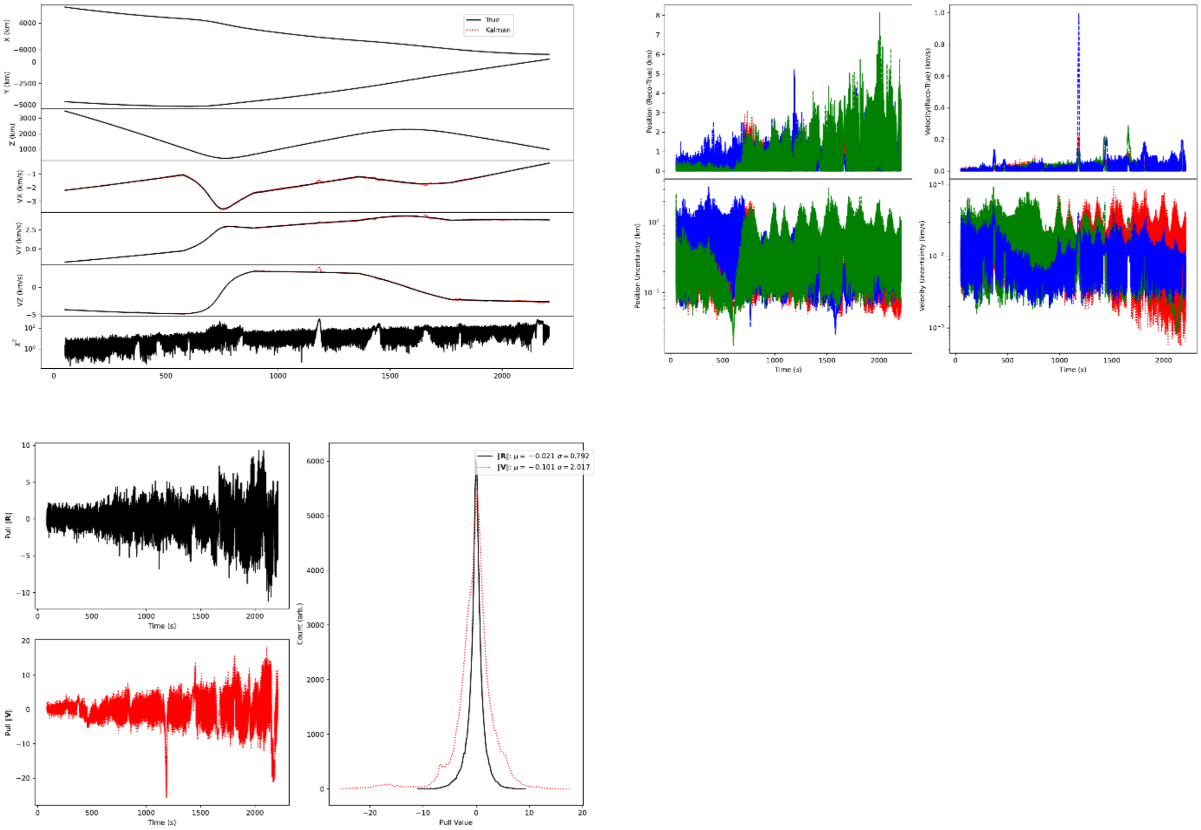


Figure 46. Plots for the segmented track (excluding the first 50 s). (a) Comparing true and reconstructed trajectories as a function of time. (b) Looking at differences between true and reconstructed for each Cartesian coordinate (Red = x, Green = y, Blue = z), and the uncertainty for each Cartesian coordinate as a function of time. (c) The pull distributions on the magnitude of the position and velocity.

5.5 Combining RCS with Vehicle Simulation and Geolocation

So far, the different components have been presented independent of each other, except for simulation and geolocation. Here, we will look at a scenario where a constellation with 1024 satellites tracks a vehicle with a simulated HGV RCS.

The RCS was simulated for every 10° in all four angles of a bistatic system. The results were stored assuming linear polarization. For this scenario, we will look at how the results change if we assume a left-right and a right-right circular polarization for the transmit-receive satellites.

For calculating the SNR, the following assumptions were made about the system:

- The digitization rate is 10 MHz.
- The power of the antenna is 1 kW.
- The wavelength of the signal is 0.043 m.
- The gain of the receive antenna is 44.18 dB , and the transmit antenna is 41.18 dB.

For the SNR calculation we assume that:

- The energy per noise power spectrum density is 7.1 dB.
- That the noise power spectrum density is 170.5 dB

For the geolocation we assume that the vehicle is traveling in a great circle and we are using the unscented Kalman filter approach. We also assumed the process noise in orbit parameter space is

$$Q_{op} = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_M^2 \Delta t^2 \end{bmatrix}, \tag{71}$$

where σ_M^2 is an extra variance in the mean anomaly that will be tuned in the simulations. To get the process noise in Cartesian coordinates Q, Q_{op} is coordinate transformed from orbital parameters to Cartesian, and the diagonal of the elements is taken. An extra σ_r^2 is multiplied to the position components of Q. This makes only two parameters that are needed to be optimized for the situation at hand.

Figure 47 shows the results of the simulations with the assumptions made above. The true trajectory for the simulation was a great circle. And a delayed correlation can be observed between the number of satellites and the pull distributions, as well as the RCS and the pull distributions. Figure 48 shows the results for only the left-right polarization, but this time the true trajectory for the simulation was segmented. The time around 1500 s in the segmented results is during a transition of the vehicle from one great circle path to another. Here the RCS and the pull distributions increase indicating that the satellites have a better view of the vehicle, but the assumed equation of state is not as accurate.

Figure 49 shows how the integration length for each bistatic pair changes as a function of RCS. As expected the integration window gets shorter as the RCS increases. Also shown, is the comparison between left-right and right-right polarization configurations. The right-right configuration has a higher RCS on average than the left-right configuration for the same vehicle flight and linear RCS calculations.

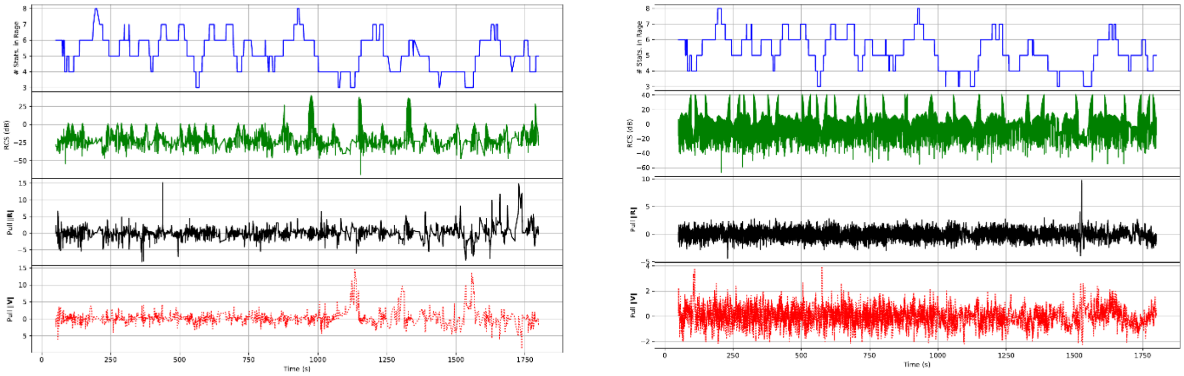


Figure 47. Showing the results from the (left) the left-right polarization and (right) the right-right polarization satellite configurations. For each plot the (1st row) is the number of satellites within 1200 km of the target, (2nd row) the RCS at the end of the integration window for each bistatic measurement that passed SNR, (3rd row) the pull distribution for the magnitude of the position from the geolocation, and (4th row) the pull distribution for the magnitude of the velocity from the geolocation.

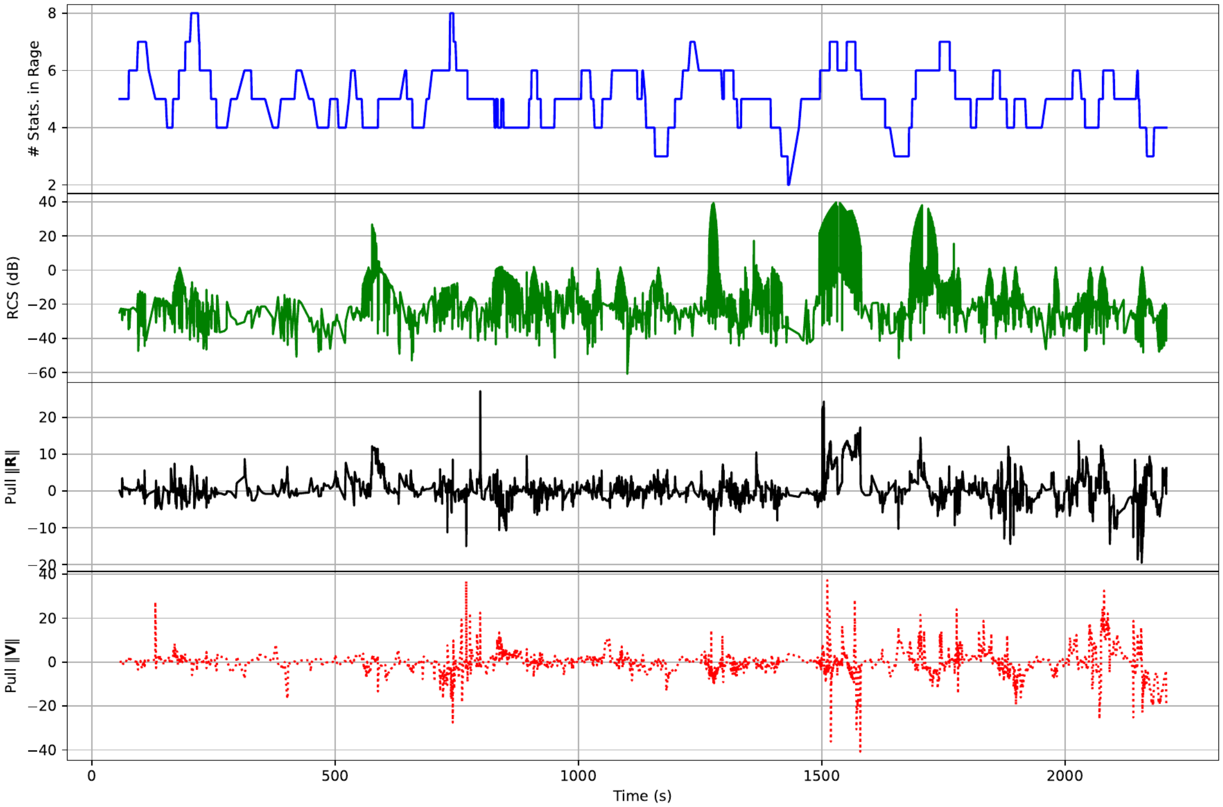


Figure 48. Showing the results of the left-right polarization configuration of the satellites with the segmented simulation. The (1st row) is the number of satellites within 1200 km of the target, (2nd row) the RCS at the end of the integration window for each bistatic measurement that passed SNR, (3rd row) the pull distribution for the magnitude of the position from the geolocation, and (4th row) the pull distribution for the magnitude of the velocity from the geolocation.

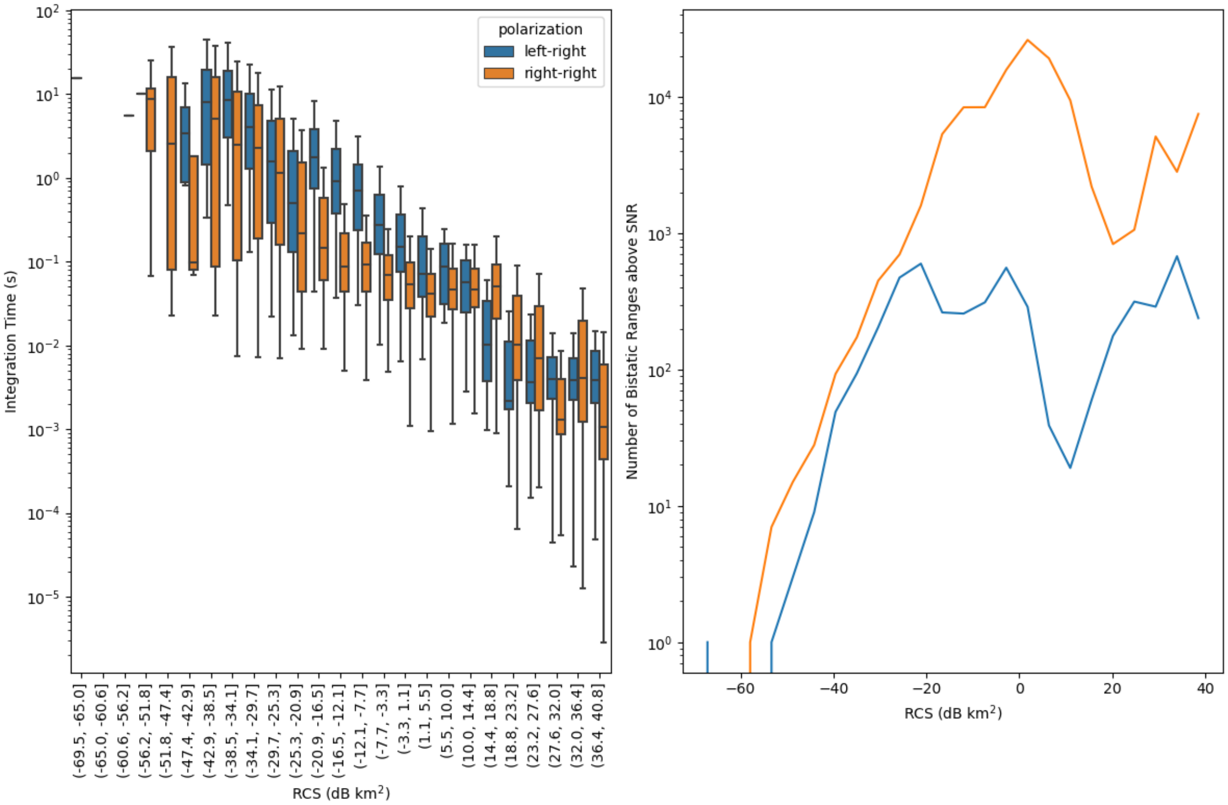


Figure 49. (left) The integration time as a function of RCS for left-right, blue, and right-right, orange, polarization configurations. (right) The number of bistatic measurements integrated over the entire vehicle flight as a function of RCS.

5.6 Terminal Phase Tracking

Here we construct a simple model of a hypersonic vehicle's terminal descent in order to develop order-of-magnitude estimates for the range and performance of these vehicles. We begin with a vehicle at altitude 50 km, traveling forward at Mach 10 and undergoing simple harmonic motion in a horizontal plane. The amplitude of the oscillations is set to 500 km, with a maximum centripetal acceleration to 10g. Taking this 10g as a maximum tolerable acceleration for the vehicle, and Mach 10 as a maximum attainable velocity, in a maximal banking maneuver the vehicle would follow an arc of radius $r = {}^{v^2}\!/_{a} =$ 120 km. Figure 50 shows the trajectory such a vehicle would follow when transitioning to a terminal phase with maximum bank, and a very gradual downward acceleration of 0.01 g, for illustrative purposes.

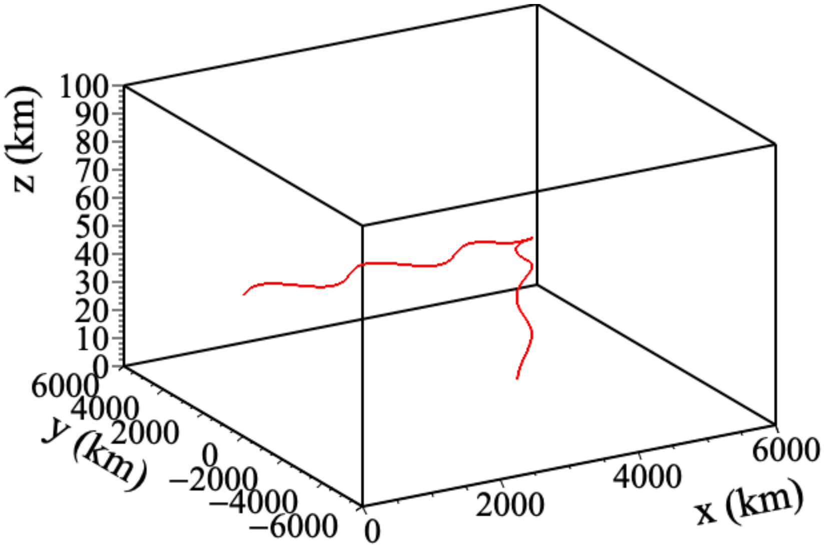


Figure 50. Trajectory of a vehicle with descending at 0.01g, with maximum centripetal acceleration of 10g and initial velocity of Mach 10.

In this simple model, the power lost to air drag by a vehicle moving at velocity v is given by $P = \frac{1}{2}\rho v^3 A C_d$, where ρ is the atmospheric density, A is the projection of the area of the vehicle perpendicular to the direction of motion, and C_d is the drag coefficient. For the order-of-magnitude estimates considered here, we set A = 1 m^2, and $C_d = 0.1$, based on typical drag coefficients for streamlined bodies. The density of air as a function of altitude is represented by an exponential distribution $\rho(h) = \rho_0 e^{-h/H}$, where ρ_0 is the atmospheric density at sea level, 1.2 kg/m^3, and the constant H = 6970 m, as determined by a fit to atmospheric data [20].

Figure 51 shows the power lost to drag as a function of Mach number (in this diagram the Mach number is computed using the speed of sound at sea level, 340 m/s). At an altitude of 50 km, a hypersonic vehicle will deposit on the order of several hundred kilowatts of power in the atmosphere, while at lower altitudes of 10 km, the energy loss rate can rise substantially to GW. For comparison, a 1000 kg vehicle moving at Mach 10 has a kinetic energy of approximately 6 GJ. Considering only this drag force, such a vehicle could cover a range of almost 3000 km at 50km altitude, and less than 400 km range at 10 km altitude.

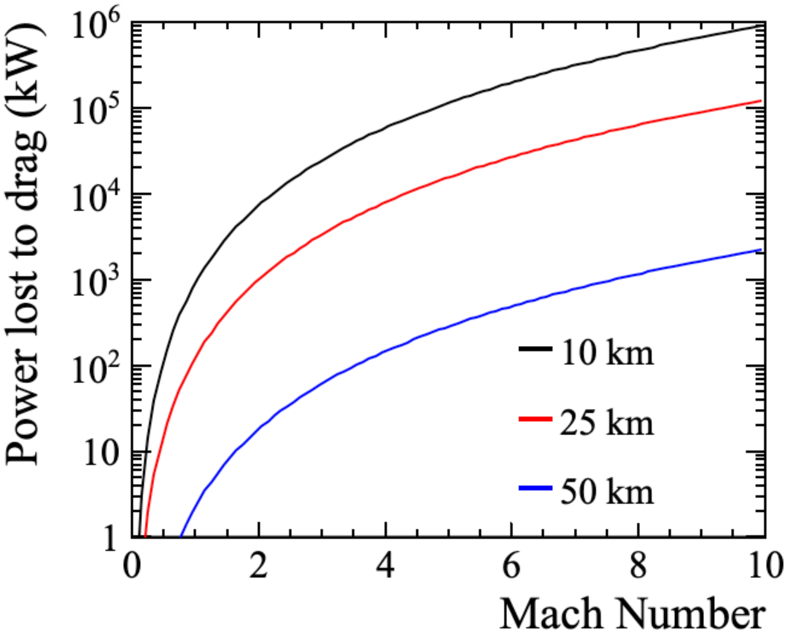


Figure 51. The power lost to air drag at various altitudes as a function of Mach number. See text for details.

Using this model of vehicle performance, a set of Monte Carlo simulation of terminal phase are run, where the vehicle performs a banking descent. The descending acceleration allowed to randomly vary between 0.01 and 1g, and the banking acceleration varies from 5g to the assumed maximum tolerable acceleration of 10g. A subset of the resulting trajectories is shown in Figure 52, and a map of the points of impact are shown in Figure 53.

The impact map shows several features of interest. First, the two lobes at positive and negative *y* axis values correspond to left and right turns of the vehicles. There are areas inside the lobes that are excluded as points of impact by the finite acceleration tolerance of the vehicles. Finally, an area on the outer edge of the most forward trajectories is excluded by the time taken to reach the ground. This simple model assumes that in the terminal phase the vehicle is fully committed to a downward trajectory, and is thereby limited to a maximum possible downward acceleration of 1g. The substantial horizontal velocity of the vehicle thereby prohibits reaching the ground near the point where the vehicle enters is descent.

To summarize, simple order-of-magnitude estimates give continental to intercontinental-range glide lengths at altitudes of 50km and higher, and ranges of few hundred kilometers at altitudes of 10 km. The terminal descent phase in a banking trajectory can reach an area of approximately 500 x 1000 km^2, when descending from altitudes of 50km.

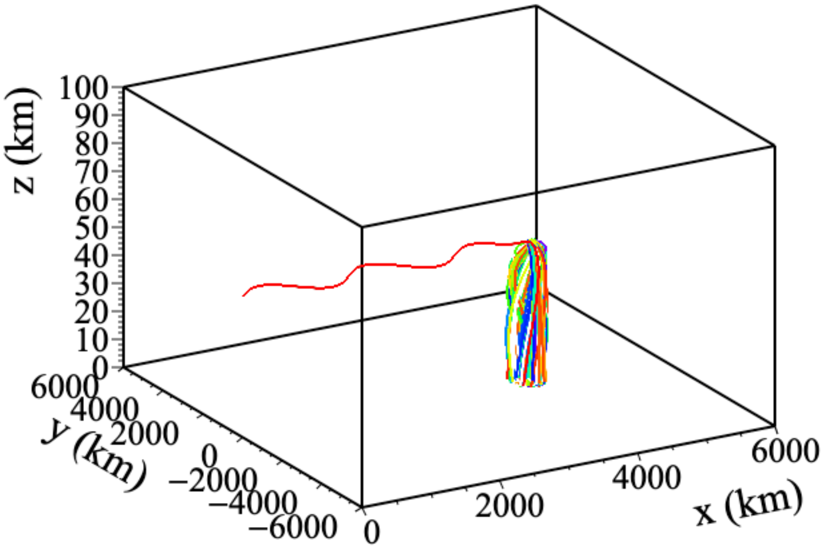


Figure 52. Simulated terminal phase trajectories.

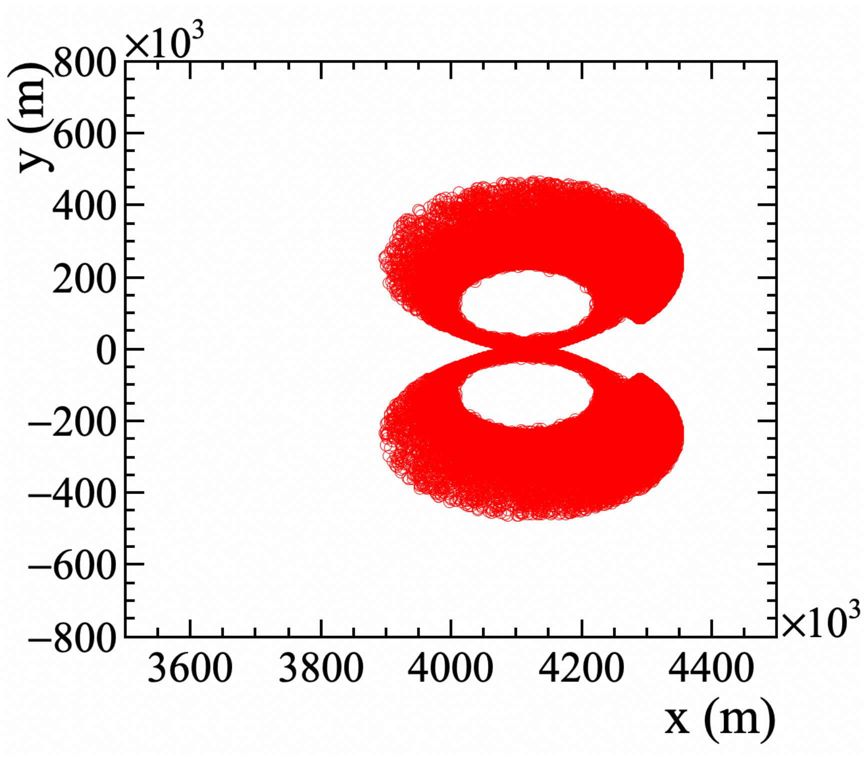


Figure 53. Map of impact points. See text for details.

5.7 References

[16] Rojer R. Labbe Jr. n.d.a. "FilterPy." https://filterpy.readthedocs.io/en/latest/.

[17] ———. n.d.b. "Kalman and Bayesian Filters in Python." *GitHub repository*. https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python; GitHub.

[18] Wikipedia contributors. 2021. "Great Circle — Wikipedia, the Free Encyclopedia." https://en.wikipedia.org/w/index.php?title=Great_circle&oldid=1006306042.

[19] Zhao, Yongsheng, Yongjun Zhao, and Chuang Zhao. 2018. "A Novel Algebraic Solution for Moving Target Localization in Multi-Transmitter Multi-Receiver Passive Radar." *Signal Processing* 143: 303–10. https://doi.org/https://doi.org/10.1016/j.sigpro.2017.09.014.

[20] National Aeronautics and Space Administration, and United States Air Force, U.S. Standard Atmosphere, 1976, NOAA-S/T 76-1562

6 Conclusion

The tracking of an HGV over its flight path is challenging because of their speeds and their altitudes. Compared to traditional ballistic missile systems, they fly at lower altitudes and they can be maneuverable. The combination of their hypersonic speed, maneuverability and flight altitude create a difficult tracking problem. The work summarized in this report documents a tool developed to explore some of the major system design parameters associated with developing a fleet of low earth orbit satellites using radar to track HGVs. In this case, we were focused on the problem of tracking one vehicle in a rather ideal flight trajectory. Further expansion of this work would include looks at more realistic trajectories as well as tackling the problem of tracking multiple vehicles at once. Developing an understanding of what is required will define the number of satellites one needs to accomplish this task including major design characteristics like power and antenna requirements.

Appendix A: Code Repository Primer

The python code and associated data for this project are stored at git.lanl.gov/mphoffmann/vehicle-tracking-di. Contact Mitch Hoffmann at mphoffmann@lanl.gov for access. The authors apologize in advance for the lack of documentation in the code, but hope this report and in particular the following discussion are enough to start you off.

There are 2 folders at the top level of the repository. The dev folder contains the simulation suite as well as a few pieces of extraneous code in varying levels of completion. The directory structure in this sub-folder is not intuitive to how the whole simulation fits together. For those interested in continuing this work, some headway on future investigations has been made in this folder. For example, measurement simulation code which takes into account acceleration error and allows for arbitrary acceleration corrections exists in this folder, but we did not feel it was mature enough to include in the end product.

For those looking to run the simulation as it stands now, we recommend starting in the other top level directory, "production." Here you will find a more clearly laid out simulation structure. A brief outline which should aid in getting started follows here:

- sim_structure
    - atmos.py
      This file contains code for simulating atmospheric pressure, density, and temperature. It incorporates the 1976 standard atmospheric model, but other models could be added easily.
    - measurements.py
      This file contains the logic for simulating mono and bi-static radar measurements using the output of the geometric simulation
    - orbits.py
      This is a large file which contains all the classes and functions for simulating trajectories of satellites and flight vehicles. The "HGV" class allows simulation of an arbitrary path. It will take inputs of waypoints and create a trajectory of several great circles stitched together, with some smoothing to turn the corners into curves. An arbitrary profile function can be passed to this class to simulate the in track and vertical acceleration of the vehicle given the current values of those states, and the cross-track G-force. You may also pass a vehicle class which contains information about the aerodynamics of the vehicle. This is used to calculate attitude during complex maneuvers. It may be advisable to include the profile function as part of this class, but is not required.
    - snr_params.py
      This file contains the class that stores variables that goes into the SNR calculation
    - Notebooks
      Look at the jupyter notebooks in this folder for examples of how to use the code. The "orbit_examples" and "generate_test_scenario" will be especially useful to the new user
- Simulation_scripts
    - This is where the real simulation scenarios should be kept. There are three examples in there now which should help with understanding the workings of the simulation. These scripts output csv files which represent the motion of all the satellites and the target, as well as the measurements received by the constellation. The files also call the geolocation code to track the vehicles and generates its output.

Appendix A: Code Repository Primer

- Geolocation
    - This is the directory that stores all the code needed for geolocation
    - geolocation.py
      This file contains the function that shows how to loop over the data and use the SatKalman class to track the vehicles. It can be run independently or inside the scripts in simulations_scripts. For independent running a configuration file needs to be passed and an example of that can be found in the geolocation directory
    - sat_kalman.py
      This file contains all the code to track the object. Although the code for use the MSRPA is in there, the default is to use the UKF.
    - position_plots.py
      This file contains all the code used to generate plots using the results of the geolocation.
    - cart_kep_conversions.py
      This file contains all the code to convert between Cartesian coordinates and orbital parameters.
- RCS
    - This directory contains all the code to calculate the RCS of a given bistatic measurement by interpolating from RCS calculations for different angles produced in CST. The main file to use is p00_vehicleRCS_v3.py. It contains a class called RCS that takes in the file to parse, and the 4 polarization parameters that are wanted.