Summer 8-2023

A Comparative Study of Vinti-Based Orbit Propagation and Estimation for CubeSats in Very Low Earth Orbits

Ethan Michael Senecal
Old Dominion University, ethanmsenecal@gmail.com

A COMPARATIVE STUDY OF VINTI-BASED ORBIT PROPAGATION AND ESTIMATION FOR CUBESATS IN VERY LOW EARTH ORBITS

by

Ethan Michael Senecal
B.S.M.E. May 2022, Old Dominion University

A Thesis Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

AEROSPACE ENGINEERING

OLD DOMINION UNIVERSITY
August 2023

Approved by:

Sharan Asundi (Director)

Sebastian Bawab (Member)

Brett Newman (Member)

ABSTRACT

A COMPARATIVE STUDY OF VINTI-BASED ORBIT PROPAGATION AND ESTIMATION FOR CUBESATS IN VERY LOW EARTH ORBITS

Ethan Michael Senecal
Old Dominion University, 2023
Director: Dr. Sharan Asundi

In recent years, there has been a growing interest in CubeSats and very low Earth orbit (VLEO) space missions. Mission SeaLion, a collaborative CubeSat mission between Old Dominion University, the U.S. Coast Guard Academy, and U.S. Air Force Institute of Technology, planned to launch a 3U CubeSat into VLEO. The VLEO mission is a particularly challenging environment for navigation and orbit propagation because drag introduces a significant perturbation for orbit models such as SGP4. Additionally, mission requirements left no capacity for attitude determination or control, further reducing knowledge of drag behavior of the satellite in flight. This deficiency is a common problem for CubeSats due to their small form factor. The mission requires the onboard computer to maintain ephemerides of the satellite orbit so it can predict communication intervals with the ground station network and downlink its orbital state. The downlinked data enables parallel orbit propagation on the ground. The time interval before initial downlink is thus of critical importance. The satellite design included a GPS receiver to obtain an orbit fix after launch and periodically maintain orbit information throughout the mission lifetime. Due to the short lifetime of the mission, solar panels were not included in the design, resulting in system batteries being the sole power source. Continuous operation of the GPS receiver for ephemerides was therefore not feasible as it would deplete the batteries too quickly.

To address this issue, an orbit propagation tool is developed to be onboard the CubeSat. The tool is based on the Vinti oblate spheroidal method with the addition of a drag routine. The propagator is named drag equipped Vinti oblate spheroidal propagator (DEVS). The tool utilizes the GPS receiver

data to obtain an initial state estimate, then propagates the CubeSat motion using the efficient DEVS routine. The propagator logic then periodically obtains another state fix from GPS, drastically reducing power usage. Analysis of accuracy of the model without GPS, shows a moderate preference toward DEVS compared to SGP4 for VLEO; in the span of 5 hours, SGP4's error is over 2 times that of DEVS's. Case studies are then performed for various GPS pinging frequencies, representing different mission requirements. The resulting effect on power draw, accuracy, and communications with the ground station networks via S-band is examined. Predicted total access time with the ground network differs from the truth model by only 1% for a 4-orbit GPS period and retains accuracy of access start and end times within 11 seconds. Link budget analysis predicts maximum error in link margin to be 0.15 decibels relative to the truth model. A state estimator is implemented to further increase accuracy of the propagator. Positional error, in terms of root mean square, does not exceed: 70 meters when the GPS is used every 5 minutes, and 1.1 kilometers when the GPS is used once per orbit. The resulting tool show cases the applicability of Vinti's solution in the modern space environment.

ACKNOWLEDGMENTS

I would like to thank my advisor and committee chair, Dr. Sharan Asundi, for his many hours teaching, advising, and motivating me throughout my education and research. My sincerest gratitude to my committee members, Dr. Brett Newman and Dr. Sebastian Bawab. I thank Dr. Bawab for his continued support of my education from my undergraduate through my masters' studies. He has always been eager to help me succeed. Dr. Newman, I thank for the instruction and guidance received in and out of classes. His in-depth knowledge pertaining to the fields of aerospace and control systems engineering has been invaluable.

The SeaLion team deserve my many thanks as they have assisted me with my research with their various expertise. Specifically, thanks to Mr. Sean Marquez for the many hours spent setting up and developing the software environment for my research. Thanks to Mr. Cian Branco who helped me get this research off the ground. I'd also like to thank our assistant dean and my supervisor, Dr. Tony Dean, who has looked out for me during my time at ODU.

My gratitude extends to my friends and family, who have supported and encouraged me throughout my life and college career. I thank my parents for their wisdom and support. Finally, I thank my sisters, brothers-in-law, and close friends, for being invested in my success and celebrating my accomplishments with me.

NOMENCLATURE

3U	3-Unit
AFIT	Air Force Institute of Technology
AGI	Ansys Government Initiative
AIAA	American Institute of Aeronautics and Astronautics
a	Semi-Major Axis
C	C Programming Language
CGA	Coast Guard Academy
DEVS	Drag Equipped Vinti Oblate Spheroidal
dB	Decibel
dBm	Decibel Milliwatt
dBW	Decibel Watt
ECI	Earth Centered Inertial Coordinate System
GPS	Global Positioning System
H	Hamiltonian
HPOP	High Precision Orbit Propagator
J_2	Second Zonal Harmonic
J_3	Third Zonal Harmonic
J_4	Fourth Zonal Harmonic
J2000	ECI Reference Frame based on Jan 2000 Reference Epoch
km	Kilometer
MC3	Mobile CubeSat Command and Control Ground Station Network
MSL	Mean Sea Level
m	Meter

Mb	Megabit
min	Minute
mW	Milliwatt
ODU	Old Dominion University
RMS	Root Mean Square
RMSE	Root Mean Square Error
$\boldsymbol{r}$	Cartesian Position Vector
$\dot{\boldsymbol{r}}$	Cartesian Velocity Vector
STK	Systems Tool Kit
s	Second
VLEO	Very Low Earth Orbit
VOSM	Vinti's Oblate Spheroidal Method
V	Gravitational Potential Function
W	Energy Per Unit Mass
X	State Vector
μ	Earth Gravitational Parameter
ρ	Atmospheric Density

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION AND MOTIVATION

1.1 Very Low Earth Orbit Space Missions

Very low Earth orbit (VLEO) is a classification of Earth orbits with altitudes less than 450 kilometers (km) [1]. Interest in VLEO missions has increased in the past few years. As Werner and Roberts discuss, VLEO offers cheaper launches, lower transmission latency and communication power, decreased radiation, and increased resolution for imagery missions [1, 2]. Additionally, Roberts continues, due to the large effects from atmospheric drag, VLEO missions automatically de-orbit, cleaning the crowded orbital space; this however, poses a problem as well if a long orbital life is desired [2]. In such a case, some propulsion method would be required to maintain a VLEO. A future Roberts envisions for VLEO is one where this class of orbits is predominantly used by short duration and unmanned missions clearing space in higher orbital regimes for missions where longevity is a larger concern.

In space missions, it is critical for a satellite to establish connection with a ground station and downlink its orbital parameters or state vector; otherwise, the satellite operators have no situational awareness of the mission. After the initial downlink, the ground station can propagate the orbit and predict the next pass in which it can obtain mission data from the satellite. Various solutions exist to accomplish this. Typically, operators use NORAD tracking data in the form of Two-Line Elements (TLEs). VLEO missions have a unique problem; due to their low altitude, tracking data is not available. Space Track reports that objects must have sufficiently long orbital lifetime in order to maintain TLEs [3]. Research performed by Riesing found that satellites in orbits under 420 km experience exceedingly high drag levels, preventing TLEs from maintaining accuracy for longer than a few hours [4]. Thus, satellites in VLEO are not easily trackable for the following reasons:

- Mission lifetime in VLEO is too short; and

- TLEs degenerate too rapidly.

To solve the issue of initial downlink in VLEO missions, an accurate onboard navigation system is needed to enable prediction of communication intervals between the satellite and ground station. This thesis aimed to develop a low power, accurate navigation system for VLEO missions. The approach pursued couples an onboard GPS receiver for orbit determination, and an orbit propagator. The GPS receiver cannot serve as the sole provider of ephemeris (time and orbital state), as this would draw far too much power. Instead, the GPS receiver acquires a GPS fix from which the propagator predicts future orbital states using an algorithm based on the Vinti oblate spheroidal method. Power usage may be limited by switching the GPS on only a few times per orbit and relying on the orbit propagator in between GPS pings. The proposed navigation system can provide continuous ephemeris data for prediction of ground station passes, without exceeding mission power budget.

1.2 Rideshares and CubeSats

Ridesharing is a common practice in space missions where launch costs are exorbitant. Often a primary payload will not reach the maximum mass or volume capacity of its launch vehicle. A brilliant use of available resources is then to fill in unused capacity with other payloads; often with different mission owners, thereby reducing launch costs [5]. The number of vehicles carried in rideshares has increased in recent years as more powerful rockets are developed. SpaceX is on the forefront of this; in January of 2023, the company launched a Falcon 9 carrying 114 space vehicles, the 6^{th} launch in the Transporter mission [6].

CubeSat is a classification of small satellites that are built from standardized, mass-produced components, thereby resulting in a cheap satellite relative to other more tailored small satellites. A 1U CubeSat, a 10x10x10 cm cube, is the unit form factor [7]. Other common builds are 2U, 3U, and 6U, which have approximately two, three, and six times the volume as a 1U design. The specifications for a 3U design are shown in Figure 1.1 below from CubeSat's spec sheet [8].

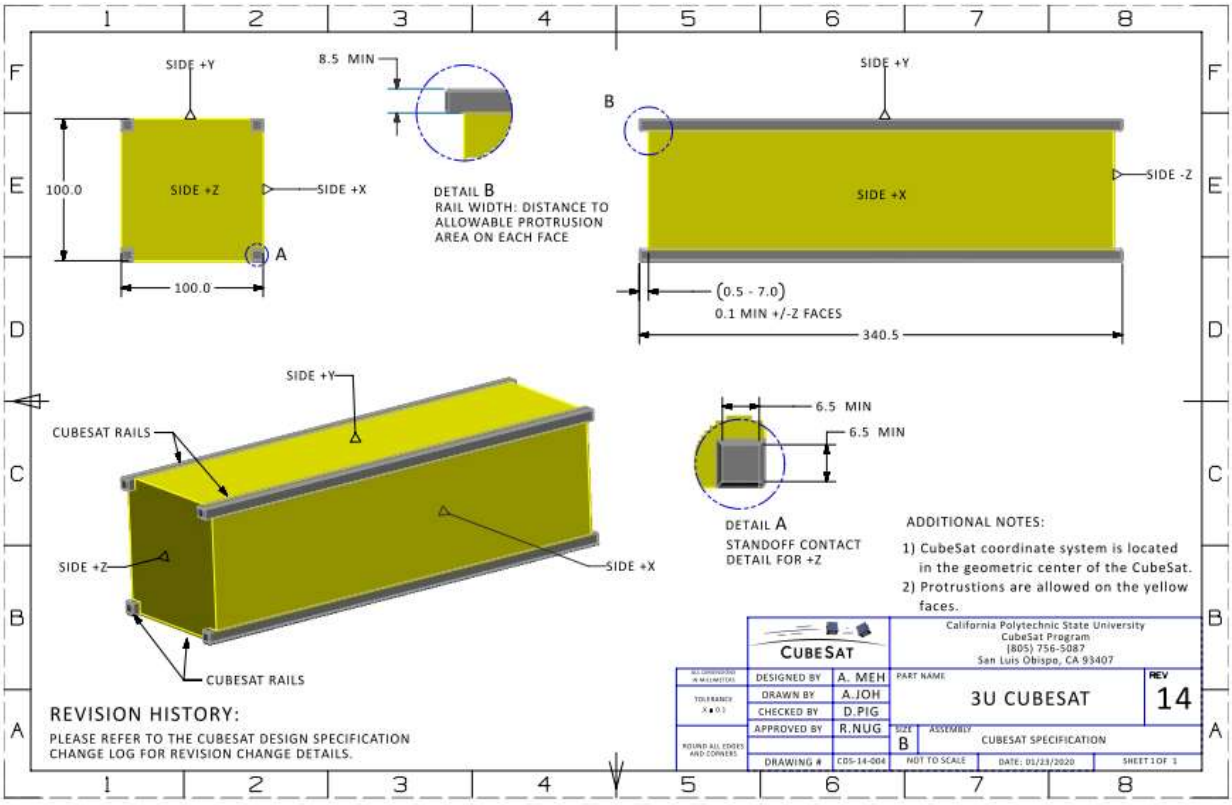

Figure 1.1: 3U CubeSat Drawing [8]

According to the European Space Agency, CubeSat missions, typically launched as a Rideshare, offer a cheap and lightweight access to space for applications ranging from academic, prototyping, and science [9]. These missions are commonly seen at universities and research institutions.

1.3 Mission SeaLion Scenario

The project which motivated this study, Mission SeaLion, is a joint project between Old Dominion University (ODU), the U.S. Coast Guard Academy (CGA), and the U.S. Air Force Institute of Technology (AFIT). The mission employs a 3U CubeSat designed to carry three payloads, one from ODU and two from CGA and AFIT. As initially planned, SeaLion would be launched from Wallops Flight Facility (WFF) in a Rideshare on a Northrup Grumman Antares rocket in March 2023 [10]. The rockets burnout trajectory would send SeaLion into VLEO with an altitude no more than 250 km. The vehicle was to have a mass of between 4 and 5.5 kg, leading to a projected lifetime of less than 10 days. Because of its short

lifetime and limited internal volume, it was determined that neither an attitude determination nor attitude control system was feasible; instead, the mass distribution would be such that drag forces would eventually yield stability [11].

Two ground station networks are available to Mission SeaLion: Mobile CubeSat Command and Control (MC3) Ground Station Network, and Virginia Ground Station Network (VGSN) [10]. The MC3 is a military led network hosting UHF, VHF, and S-band radio signals with stations at AFIT, Army Space & Missile Defense Command (SMDC), CGA, Hawaii Spaceflight Lab (HSFL), Malabar Transmitter Annex (MLB), Naval Postgraduate School (NPS), Space Dynamics Laboratory (SDL), Texas A&M University (TAMU), University of Alaska Fairbanks (UAF), University of New Mexico (UNM), and US Naval Academy (USNA) [12]. The VGSN is a UHF/VHF joint network with stations located at ODU, University of Virginia (UVA), Virginia Tech (VT), and WFF [13]. Both ground station networks aim to extend coverage for CubeSat missions. Together, these networks offer sufficient opportunities to downlink mission data, so long as the satellite is aware of its access intervals. After the satellite has downlinked ephemerides, the ground operators would perform parallel orbit propagation to add redundancy in prediction of future accesses for downlinking mission data.

Even with the coverage provided by the ground station networks, timeframes to communicate with the SeaLion satellite are limited. The portion of the ground that is in range for communication is known as the swath and is determined by the transmitter cone angle and altitude of the orbit. Figure 1.3 pictures the swath of a transmitter with a 30-degree cone half-angle over a ground station in Florida. The image was created in Systems Tool Kit (STK) 12.2, Ansys Government Initiatives (AGI) [14].

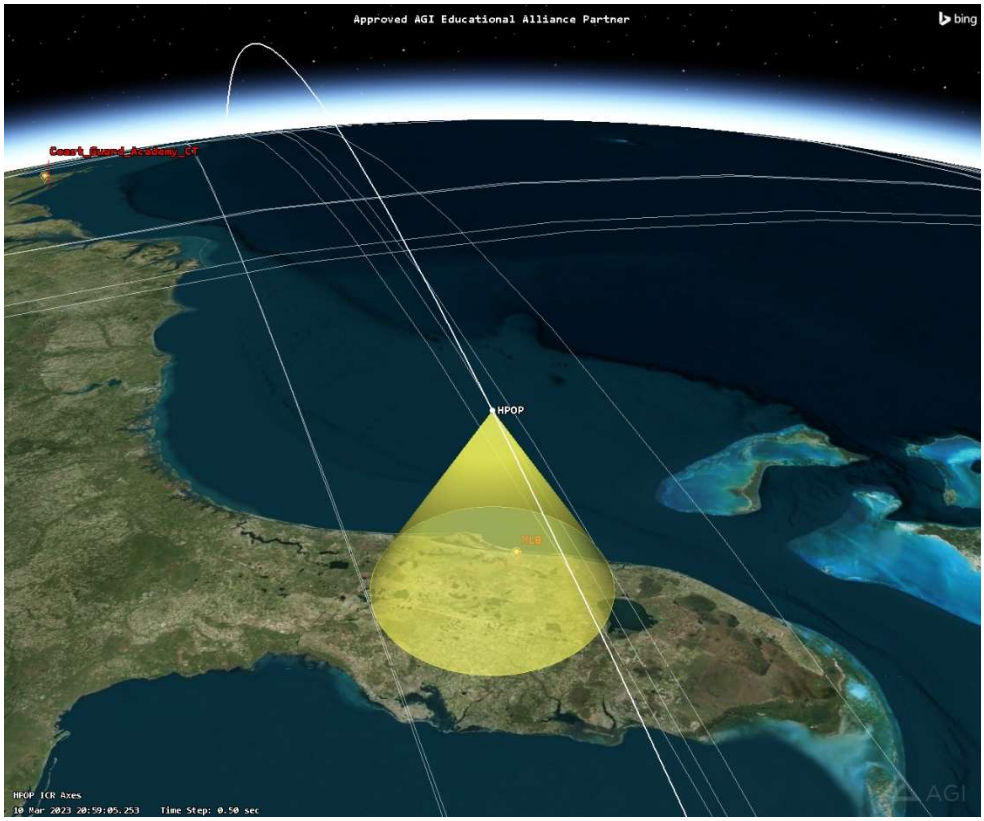

Figure 1.2: Image of Sensor Swath over a Ground Station

In order for the satellite to downlink orbital and mission data, it must accurately predict when a ground station is within its swath. Thus, the satellite must continuously upkeep its ephemeris with the onboard navigation system. Therefore, the orbit propagator must be based on accurate governing equations.

The payloads aboard the SeaLion satellite are an impedance probe, a multispectral sensor, both provided by CGA and AFIT, and a deployable composite boom structure, provided by ODU [15]. The purpose of the impedance probe is to measure plasma temperature and density in the upper atmosphere and the multispectral sensor will allow for a baseline spectroscopy reading for future missions [10]. The deployable boom is a proof of concept design which may be employed in future missions to deploy devices such as solar sails, solar panels, etc. [10]. Design, fabrication, and material

testing of the boom was performed and presented at the AIAA SciTech Conference by Mr. Jimesh Bhagatji [16]. Figure 1.2 shows the SeaLion satellite with the boom deployed.

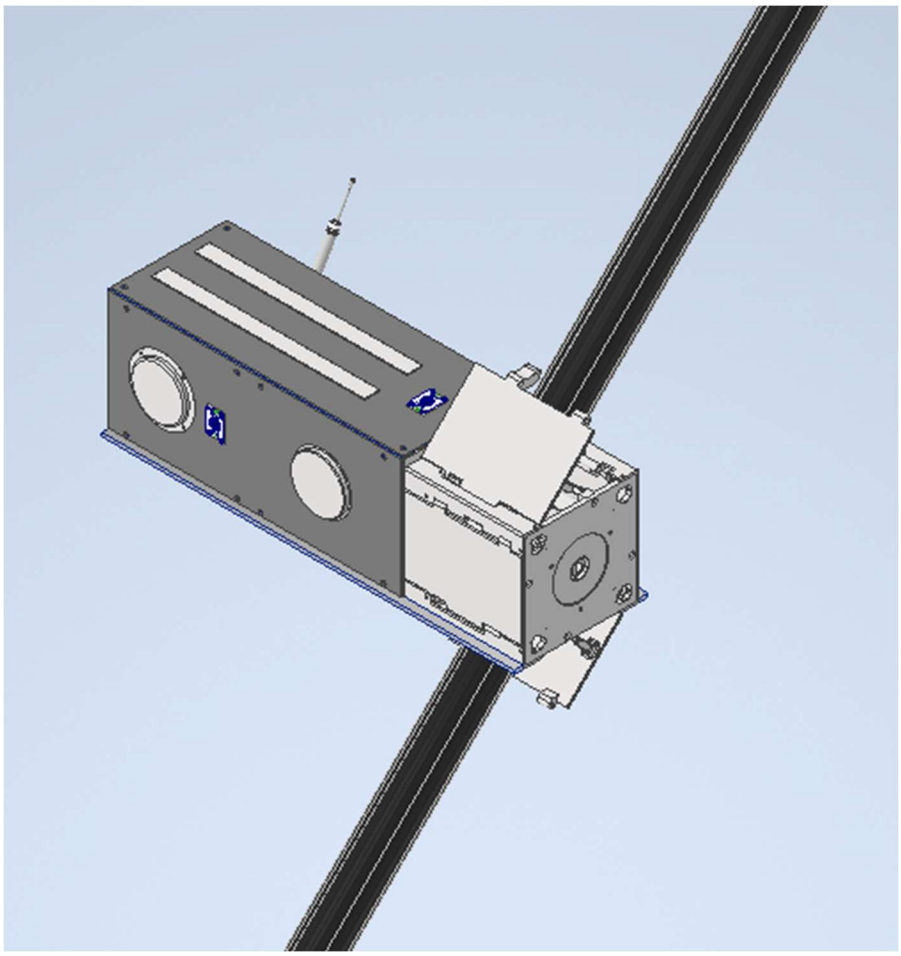

Figure 1.3: SeaLion CubeSat Model - Boom Deployed

Two additional papers pertaining to Mission SeaLion were presented at SciTech. The first was a study by Mr. Sean Marquez and Mr. Kevin Chiu presenting a model-based approach to designing flight software systems and a Docs-as-Code approach to development [17]. This paper was also published as an article in the MDPI journal and as a master's thesis defended by Chiu at ODU [15, 18]. The second conference paper was a comparative study of the Vinti6 and SGP4 propagation routines, the initial iteration of this thesis [11]. Another critical contribution to SeaLion was the thesis written by Mr. Robb

Borowicz on failure modes of CubeSat missions and subsequent corrective actions required in the design and redesign of SeaLion [19]. Finally, the work of Joe Siciliano details the design of electrical and communication systems employed on the SeaLion satellite [20]. Mission SeaLion involves many aspects of engineering, each critical to the success of the mission.

The initial conception of mission SeaLion was altered due to Antares being over its mass budget. The mission will continue and is now set to launch on a Firefly Alpha rocket in late 2023. However, this thesis, is based on the initial specifications of Mission SeaLion, examining orbit propagation in VLEO as this is a matter of increasing significance to the space sector.

1.4 Orbit Propagation

A satellite in orbit follows mostly a predictable path as governed by the laws of motion. With only the initial orbital state, a vector describing the position ($\boldsymbol{r}$) and velocity ($\dot{\boldsymbol{r}}$) of a satellite, it is possible to predict another orbital state at any time (t) based on the governing equations of motion; this procedure is known as orbit propagation [21]. Of critical importance is then to obtain a reliable initial condition. Several procedures exist to determine the initial state, commonly referred to as the state at orbital epoch. Typically, a collection of observations are made from an onboard GPS receiver or a ground station radar system [21]. This data may then be fused to improve accuracy and decrease noise via a least squares estimation such as implemented by Larson and Wright [22, 23].

Once the orbit has been determined, the initial state can then be propagated forward to the next state; however, the prediction procedure is not always a simple task. Unsurprisingly, this field has been studied for hundreds of years. Newton and Kepler discovered equations effective at predicting many astronomical phenomena such as bounded and unbounded orbits, comet trajectories, and to some degree, artificial satellites [24]. Newton's law of gravitation, a fundamental starting point of orbital mechanics, is presented in standard form in Equation (1) and in Hamiltonian form in Equation (2), where

μ is the gravitational parameter and the terms involving p, are the momenta components. Variable H, is the Hamiltonian, a measure of the total specific energy, Equation (3) [25, 26].

$$\frac{d^2\boldsymbol{r}}{dt^2} = -\frac{\mu}{r^3}\boldsymbol{r} \tag{1}$$

$$H = \frac{1}{2}\left(p_x^2 + p_y^2 + p_z^2\right) - \frac{\mu}{r} \tag{2}$$

$$H = Kinetic\ Energy + Potential\ Energy \tag{3}$$

The models are simple and approachable but lack fidelity to account for the various perturbations present in actual spaceflight. Some of the primary perturbations are due to atmospheric drag and the oblateness of Earth, i.e., Earth is not a perfect homogeneous sphere. The oblateness leads to uneven mass distributions, causing significant error between the actual dynamics and Equation (1). A natural evolution of Equation (1) is to include a disturbance acceleration vector, $\boldsymbol{a}_d$, containing the variation from Newton's equation due to the perturbations as shown in Equation (4) [27].

$$\frac{d^2\boldsymbol{r}}{dt^2} = -\frac{\mu}{r^3}\boldsymbol{r} + \boldsymbol{a}_d \tag{4}$$

Modeling this perturbation vector is complicated and varies by implementation. A myriad of methods and equations have been developed to produce reliable analytical propagations starting with Kepler's equations and extending to the far more robust Simplified General Perturbations (SGP) and Vinti models. While the Keplerian approach is efficient, it lacks desired accuracy; it does not model perturbations such as:

- Drag,
- Variation in the gravitational potential as a result of:
    - Earth's irregular mass distribution, known as harmonics,
    - or other celestial bodies (i.e., multi-body problem).

Harmonic perturbations are classified as either: zonal, denoted by J_i where i is a positive integer, in which the geopotential depends on latitude; sectoral, in which the geopotential depends on longitude; or tesseral, which are the most accurate and the resulting geopotential depends on longitude and latitude [28].

The differential equations governing perturbed orbit propagation are complex and have many terms that while increasing the accuracy of the model, lead to a set of equations that are particularly difficult to solve analytically and in some cases are not separable. A natural solution to this is a numerical integration of the governing equations. Numerical integrators are the most accurate propagation methods because they can include as many terms and factors as required by the mission. The High Precision Orbit Propagator (HPOP) is one such example of a very accurate propagator [29-31]. Numerical integration methods, such as HPOP, are classified as a special perturbation method. As described in Introduction to Orbital Mechanics, special perturbation methods output a solution for the motion of a spacecraft in a specific orbit during a specific time interval, hence the name special [28]. Therefore, special perturbation methods, and by extension numerical integrators, are computationally intensive requiring excess process time or a powerful processor, neither feasible onboard a CubeSat. As a result, analytical solutions with reasonable accuracy have been sought. Analytic solutions fall under the category of general perturbation methods which, as the name implies, output solutions valid for any orbit and time interval [28]. It should be noted that general perturbation methods are developed using a series solution and truncate terms to allow for analytic solutions; thus they are approximate solutions [28]. Two common branches of perturbed analytical propagation theory are Brouwer's, on which SGP4 was originally based, and Vinti's [32].

CHAPTER 2

DESIGN OF ORBIT PROPAGATION SCHEME

2.1 Vinti Oblate Spheroidal Method

A paper by Der and Kalman of Pumpkin, Inc. presents an implementation of Vinti oblate spheroidal method (VOSM) for a CubeSat in low Earth orbit (LEO) at an altitude of 1400 km [33]. They found Vinti7 to offer exceptional performance over SGP4 with 2 to 10 times the accuracy. Der and Kalman's paper inspired the use of VOSM in this work. The VOSM, developed half a century ago by John P. Vinti, is by no means a recent addition to the space industry. Despite its accuracy and efficiency, it is not widely used due to the popularity of SGP4 and other methods. Gim Der and several of his colleagues have long been proponents of Vinti's work, arguing that the lack of utilization and discussion of VOSM is a disservice to the space community [25]. VOSM is unique in that it solves the perturbed Kepler problem via the Hamilton-Jacobi equation – Equation (5) – in oblate spheroidal coordinates (OSC) using the 1966 Vinti gravitational potential model in Equation (6), thereby including zonal harmonics up to J_3 and 75% of J_4 [26].

$$H(q, \frac{\partial W}{\partial q}, t) = \alpha_1 = constant \tag{5}$$

$$V = -\frac{\mu(\xi + \delta\eta)}{\xi^2 + c^2\eta^2} \tag{6}$$

The variables used in Equations (5) and (6) denote the following:

- *H* – Hamiltonian which describes the total specific energy of the orbiting body,
- q – generalized spatial coordinate,
- *W* – energy per unit mass,
- t – time,
- α_1 – the constant value of the Hamiltonian, named the Jacobi constant,

- V – geopotential,
- ξ and η – oblate spheroidal coordinates, and
- c and δ – fitting parameters defined in Equations (7) and (8) based on the planetary parameters defined in Table 2.1.

Table 2.1: Planetary Orbital Parameters

Gravitational Constant (μ)	Equatorial Radius (r_e)	2nd Zonal Harmonic (J_2)	3rd Zonal Harmonic (J_3)
$398600.5\ km^3/s^2$	$6378.137\ km$	$1082.62999 * 10^{-6}$	$-2.53215 * 10^{-6}$

$$c^2 = r_e^2 J_2 \left(1 - \frac{J_3^2}{4J_2^3}\right) \tag{7}$$

$$\delta = -\frac{r_e J_3}{2J_2} \tag{8}$$

Oblate spheroidal coordinates are chosen since Earth is accurately described as an oblate spheroid albeit with some imperfections. Figure 2.1 from Wolfram MathWorld aid in visualizing these coordinates which are derived from elliptic cylindrical coordinates, revolved about the vertical (z) axis [34].

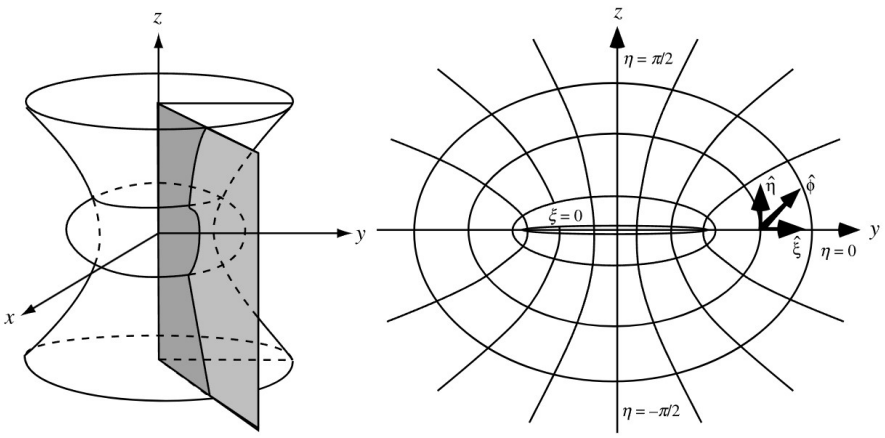

Figure 2.1: Oblate Spheroidal Coordinates [34]

Coordinates x, y, and z are the Cartesian coordinate axes. The vertical plane on the left of Figure 2.1 corresponds to a single elliptical plane like that shown on the right. The azimuthal angle of the plane is described by ϕ, bounded by 0 and 2π. Coordinate ξ is a positive real number describing the size of the spheroidal surface which roughly correlates to the radial distance used in the more familiar spherical coordinates. Coordinate η, bounded between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$ determines the curvature of the hyperbolic surface on the left. The intersection of these three surfaces, the vertical plane, oblate spheroid, and hyperboloid of revolution, determine the location of a point. Unit vectors $\hat{\xi}$, $\hat{\eta}$, and $\hat{\phi}$ correspond to reference values of the surfaces.

There are several implementations of VOSM. While Der and Kalman used Vinti7, Vinti6 was selected for this work based on its public availability. Vinti6 was developed in Fortran by Gim Der and available in the supplemental material of Vinti's book [35]. The version was ported to the C programming language by Herb Reynolds; this is the version used by the author. Three required C files are found in "vinti_source\source\Other source" of the supplemental material folder:

- Mainvinti.c,
- vinti6.h, and

- vinti6.c.

The file vinti6.c was left unaltered other than adding a hot fix for an issue with malformed outputs. A few identified input state vectors were returning nonnumeric (NaN) outputs. The issue originated from near zero negative values not being properly zeroed and subsequently square rooted [36]. Per the suggestion of other SeaLion team members, this issue was remedied by applying the absolute values to these occurrences. The interface file Mainvinti.c was modified to meet the needs of this research, changing the output file to only include numeric values of the Earth centered inertial (ECI) state vector. Note the computer routine requires the <Math.h>, <stdio.h>, and <time.h> header files. Unit tests were performed on the code and outputs were consistent with example outputs provided by Der in the supplemental material [35].

The algorithm of the VOSM is described in Appendix B of Orbital and Celestial Mechanics, and is presented below [26]. Given initial state vector, $\boldsymbol{X}_i$ at time t_i, and time difference, between initial time and final time, t_f, the new state vector, $\boldsymbol{X}_f$ is computed.

1. To avoid numerical issues, use Equation (9) to convert from dimensional units to Earth units (EU), dimensionless units based on Earth's equatorial radius and gravitational parameter, noting that $\boldsymbol{r}$ and $\dot{\boldsymbol{r}}$ are Cartesian position and velocity vectors,

$$\begin{bmatrix} \boldsymbol{r}_{EU} \\ \dot{\boldsymbol{r}}_{EU} \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}\dfrac{1}{r_e} \\ \dot{\boldsymbol{r}}\dfrac{1}{r_e}\sqrt{\dfrac{r_e^3}{\mu}} \end{bmatrix} \tag{9}$$

   The reader should note that after conversion to EU, the r_e terms vanish from c and δ in Equations (7) and (8),

2. Compute initial estimate of $\boldsymbol{X}_f$ using Kepler routine. Obtain universal variable, $\hat{x}$, where the universal variable is a generalized flight angle analogous to the true anomaly [35, 37],

3. Transform ECI coordinates to Oblate Spheroidal coordinates using Equations (10) through (14),

$$\boldsymbol{X}_i = \begin{bmatrix} \xi_i \\ \eta_i \\ \phi_i \\ \dot{\xi}_i \\ \dot{\eta}_i \\ \dot{\phi}_i \end{bmatrix} = \begin{bmatrix} \left(\frac{d}{2} + \frac{1}{2}\sqrt{d^2 + 4c^2(z_i + \delta)^2}\right)^{\frac{1}{2}} \\ \frac{z_i + \delta}{\xi_i} \\ \text{atan}\left(\frac{y_i}{x_i}\right) \\ \frac{\sqrt{F}}{\xi_i^2 + c^2\eta_i^2} \\ \frac{\sqrt{G}}{\xi_i^2 + c^2\eta_i^2} \\ \frac{x_i\dot{y}_i + \dot{x}_i y_i}{D^2} \end{bmatrix} \tag{10}$$

$$d = \left|\left|\boldsymbol{r_i}\right|\right|^2 - c^2 + \delta(2z_i + \delta) \tag{11}$$

$$\sqrt{F} = \xi_i(\boldsymbol{r}_i \cdot \dot{\boldsymbol{r}}_i) - (c^2\eta_i - \delta\xi_i)\dot{z}_i \tag{12}$$

$$\sqrt{G} = \eta_i(\boldsymbol{r}_i \cdot \dot{\boldsymbol{r}}_i) - (\delta\eta_i - \xi_i)\dot{z}_i \tag{13}$$

$$D^2 = \left(\xi_i^2 + c^2\right)\left(1 - \eta_i^2\right) \tag{14}$$

4. Calculate $\alpha = \alpha\left(\boldsymbol{X}(t_i)\right)$ the Jacobi Constants,

5. Factorize the quartics, F and G numerically,

6. Initialize integration coefficients,

7. Calculate remaining Jacobi Constants, β,

8. Solve the generalized Kepler Equation at t_f,

9. Return to ECI coordinates, Equations (15) and (16),

$$X_f = \begin{bmatrix} x_f \\ y_f \\ z_f \\ \dot{x}_f \\ \dot{y}_f \\ \dot{z}_f \end{bmatrix} = \begin{bmatrix} D\cos\phi_f \\ D\sin\phi_f \\ \xi_f\eta_f - \delta \\ \dot{D}\cos\phi_f - \dot{\phi}_f D\sin\phi_f \\ \dot{D}\cos\phi_f + \dot{\phi}_f D\cos\phi_f \\ \dot{\xi}_f\eta_f + \xi_f\dot{\eta}_f \end{bmatrix} \tag{15}$$

$$\dot{D} = \frac{\xi_f \dot{\xi}_f \left(1 - \eta_f^2\right) - \eta_f \dot{\eta}_f \left(\xi_f^2 + c^2\right)}{D} \tag{16}$$

10. Convert back to dimensional units from dimensionless EU.

Thus, while the VOSM procedure is based on elaborate mathematical theory, the resulting procedure is rather straightforward and can easily be performed by a computer program.

2.2 Orbit Propagator Architecture

Now, with the VOSM established, the architecture of the drag equipped Vinti oblate spheroidal propagator (DEVS) can be presented. A flowchart of the propagator logic is given in Figure 2.2.

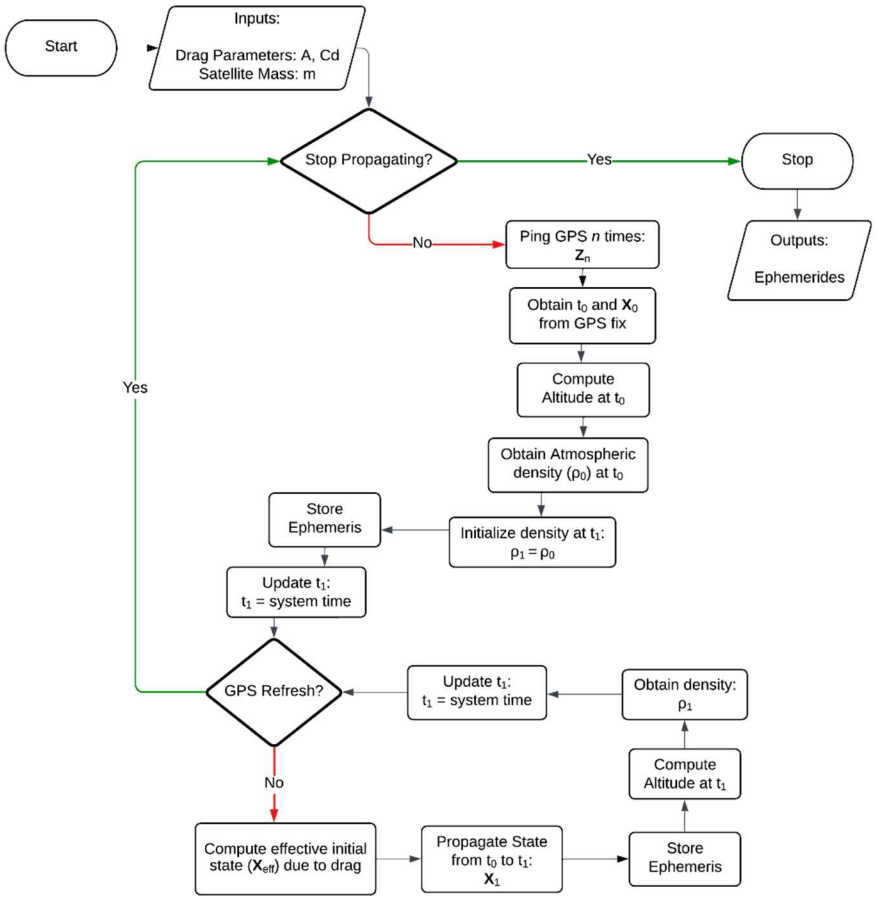

Figure 2.2: Updated Propagator Architecture

The algorithm consists of two loops:

- The outer loop periodically redetermines the satellite orbit through contact with GPS and passes the information to the inner loop.
- The inner loop continuously carries out propagation from the last state estimate at the previous GPS fix.

Drag is incorporated by calculating an effective velocity with Equation (21). Equations (17) through (20) detail the derivation of Equation (21), where m denotes the mass of the satellite. The values for density are found using a lookup table generated by MATLAB (MathWorks) function: Complete 1976 Standard Atmosphere [38].

$$F_D = ma_d = m\frac{d\dot{r}}{dt} \tag{17}$$

$$\frac{d\dot{r}}{dt}m = \left(\frac{C_d S_{Ref}}{2}\right)\rho\dot{r}^2 \Rightarrow \int_{\dot{r}_0}^{\dot{r}_1}\frac{1}{\dot{r}^2}d\dot{r} = \frac{C_d S_{Ref}}{2m}\int_{t_0}^{t_1}\rho(t)dt \tag{18}$$

$$Let\ \rho(t) = \rho_0 + \frac{d\rho}{dt}(t - t_0) = \rho_0 + \frac{\rho_1 - \rho_0}{t_1 - t_0}(t - t_0) \tag{19}$$

$$\int_{t_0}^{t_1}\rho(t)dt = \left(\rho_0 + \frac{\rho_1 - \rho_0}{2}\right)\Delta t\ \sim\ \rho_0\Delta t \tag{20}$$

$$\therefore \left|\left|\dot{\boldsymbol{r}}\right|\right|_{1_{drag}} = \left(\frac{1}{\left|\left|\dot{\boldsymbol{r}}_0\right|\right|} + \ c_d\frac{S_{Re}}{2m}\left(\rho_0 + \frac{\rho_1 - \rho_0}{2}\right)\Delta t\ \right)^{-1} \tag{21}$$

This value is then fed into the initial state vector, $\boldsymbol{X}_0$, as the effective velocity, forming the effective initial state, $\boldsymbol{X}_{0_{eff}}$ in Equation (22) where w is a weighting factor determining how much the effective velocity depends on the computed velocity due to drag. The model was experimentally found to behave best when w was set to 0.2 as other values tended to either underpredict or overshoot drag predictions.

$$\dot{r}_{0_{eff}} = (1 - w)\,\dot{r}_0 + w\,\dot{r}_{1_d} \Rightarrow \boldsymbol{X}_{0_{eff}} = \begin{bmatrix}\boldsymbol{r}_0 \\ \dot{r}_{0_{eff}}\ \hat{\boldsymbol{u}}_{\boldsymbol{0}}\end{bmatrix} \tag{22}$$

Variable $\widehat{\boldsymbol{u}}_{\boldsymbol{0}}$ is the velocity unit vector at t_0. This new effective state vector is then run through the VOSM routine.

To make the simulation simpler, the code running the algorithm was developed in GNU Octave within the SeaLion workspace image for a consistent Linux based environment. GitHub Repository was utilized for version control. The Octave script accesses the VOSM executable file through system commands, calling it as one would a function. The script inputs the state vector, $\boldsymbol{X}_{\boldsymbol{0}}$, and time difference, $t_1 - t_0$, through a text file read into the VOSM executable. The output state vector, $\boldsymbol{X}_0$, is similarly written by the executable to a text file and then retrieved by Octave. The drag calculations and GPS fixes are performed within Octave. Simulated GPS data is generated via an STK numerical integrator and accessed by Octave through another lookup table. This program may be ported to C or another desired language for improved performance and implementation on a CubeSat onboard computer (OBC). The entire package, including all C code and octave scripts, is available as a module of the mission SeaLion Git repository [39].

2.3 Propagation Differences between SGP4 and DEVS

While DEVS inputs and outputs motion data in the ECI coordinate system and performs its computations in OSC, SGP4 uses a form of mean orbital elements that are initialized by Two Line Elements (TLEs) [32]. NASA defines classical (or Keplerian) mean elements as [40]:

- Semi-Major Axis (a): size of orbit, usually in km,
- Eccentricity (e): non-circularity; for bounded orbits, values range from 0 (circular) to 1 (parabolic),
- Inclination (i): angle of orbit relative to equatorial plane,
- Right Ascension of Ascending Node (RAAN, or Ω): rotation of orbital plane,
- Argument of Perigee (ω): angle between ascending node and periapsis,
- True Anomaly (ν): location angle of satellite in its orbital plane.

Figure 2.3 below presents a visualization of orbital elements for the reader. The descending and ascending nodes are the locations in orbit where the orbiting body crosses the horizontal plane. Here, the longitude of ascending node is the RAAN. The semi-major axis, *a*, is half of the ellipses' longer principal length and the semi minor axis, *b*, is half the ellipses' shorter principal length. According to Taylor, the eccentricity is related to the semi-major and semi-minor axis by Equation (23) [24].

$$e = \sqrt{1 - \left(\frac{b}{a}\right)^2} \tag{23}$$

The apoapsis and periapsis, or apogee and perigee if Earth is the central body, are the points at which the orbiting body is furthest from and closest to the central body, respectively.

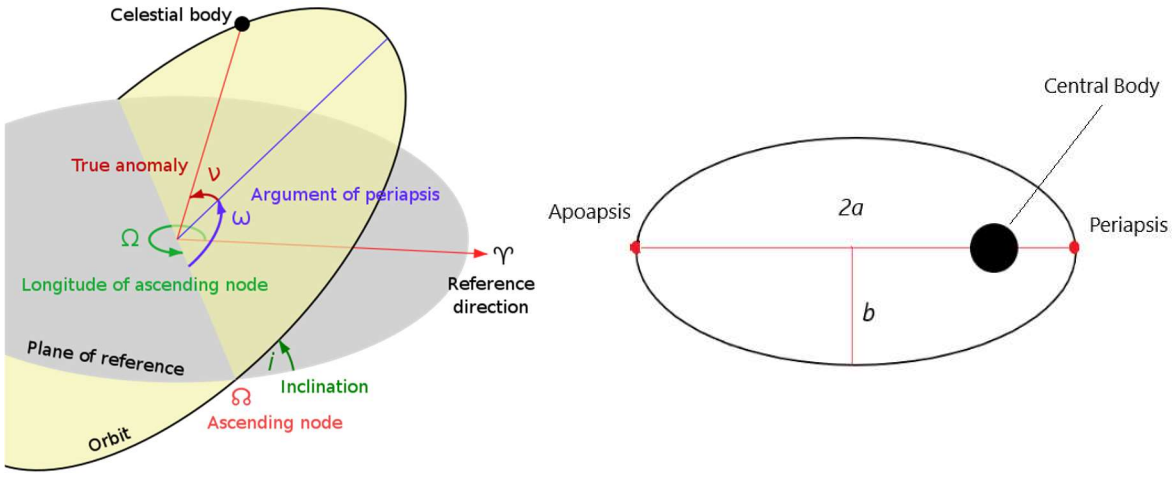

Figure 2.3: Visualization of Orbital Elements [41]

CelesTrack defines SGP4 mean elements similarly, except that SGP4 uses mean anomaly (M) instead of true anomaly and mean motion (n), the rate of change of mean anomaly, rather than semi-major axis [42]. The mean anomaly, utilized to simplify the equations of motion, is a circularized version of the true anomaly which gives the same orbital period [37]. The mean motion indirectly describes the size of the orbit and is related to the semi-major axis by Equation (24) [23].

$$n = \sqrt{\frac{\mu}{a_e^3}} \tag{24}$$

SGP4 and DEVS utilize similar model assumptions. Both propagators account for zonal harmonics, J_2, J_3, and J_4 and drag perturbations, while neither account for 3rd body perturbations, sectoral or tesseral harmonics in VLEO [26, 43]. However, the implementation and order of terms used related to the zonal harmonics differ between methods. SGP4 carries secular (non-periodic) effects of J_2 through order 2 and periodic terms to first order of J_2 [44]. In DEVS, secular terms are accurate to order J_2^3 and periodic terms are accurate to order J_2^2 [45, 46]. The methods used to include drag effects also differ. SGP4's approach uses a power density function for the atmospheric density while DEVS uses a lookup table [44]. Additionally, DEVS predicts the drag effect preemptively by averaging the drag force for a given time interval and applying a corresponding correction factor to the initial velocity. SGP4 on the other hand embeds the drag perturbation in the solution more elegantly, directly accounting for changes in the orbital elements [43]. A summary of the differences between the two propagation methods is given in Table 2.2 below.

Table 2.2: Summary of Model Differences between SGP4 and DEVS in VLEO

Propagator	Secular Terms	Periodic Terms	Drag	Atmospheric Density
SGP4	J_2, J_4, J_2^2	J_2, J_3	Effect embedded in equation	Power density function
DEVS	J_2, J_2^2, J_2^3	J_2, J_3, J_2^2	Precomputes effect	1976 atmosphere lookup table

SGP4, as compared to DEVS, likely has a better drag implementation but a less refined geopotential implementation because SGP4 does not include the zonal harmonics in terms as high order as DEVS. It should also be noted that SGP4 includes more features than DEVS in higher altitude orbits. For example,

in orbits with periods greater than 225 minutes, tesseral terms are utilized, and the effects of solar and lunar gravity are modeled using a point mass approach [43, 44]. DEVS does not include any of this functionality. For the use case investigated, that is VLEO, these features are not necessary, but the distinction is made to clarify the shortcomings of DEVS to the reader if an implementation is sought for missions with longer orbital periods. For VLEO missions, DEVS is theoretically superior due to its higher fidelity incorporation of the zonal harmonics. Analysis on the drag implementation in DEVS must be performed before a conclusion can be made on its effectiveness.

CHAPTER 3

SIMULATION AND RESULTS

3.1 Performance of SGP4 vs DEVS

Before implementing a drag model, it is necessary to compare the original Vinti6 code with a numerical integrator, taken as a truth model, to verify that it performs to the extent Vinti and Der claim. Two numerical integrators were selected: one that computes up to J_2; and another up to J_4. Neither prediction computes the drag as the purpose of this first simulation is only to ensure the accuracy of Vinti's geopotential model. The initial conditions are given in Tables 3.1 and 3.2. In STK the initial condition is generated from TLEs for an SGP4 propagation model and given in Table 3.1.

Table 3.1: Initial Condition TLE

n (deg/sec)	0.067598127943
e	0.00734957
i (deg)	52.80135
Arg Per (deg)	330.4165
RAAN (deg)	19.16215
M (deg)	174.3266

VOSM requires ECI input. Thus, the equivalent initial condition in the J2000 reference frame is also required. The ECI state vector is presented in Table 3.2 and is acquired from the SGP4 satellite object in STK via the Cartesian position and Cartesian velocity data providers as described in STK Help [47].

Table 3.2: Initial Condition ECI

r_1 (km)	-5877.600
r_2 (km)	428.240
r_3 (km)	3051.400
$\dot{r_1}$ (km/s)	-2.9910
$\dot{r_2}$ (km/s)	-5.0497
$\dot{r_3}$ (km/s)	-5.0231

The results are shown in Figure 3.1 below. The orange line represents Vinti6's error relative to the J_2 model and the blue line, the error relative to the J_4 model.

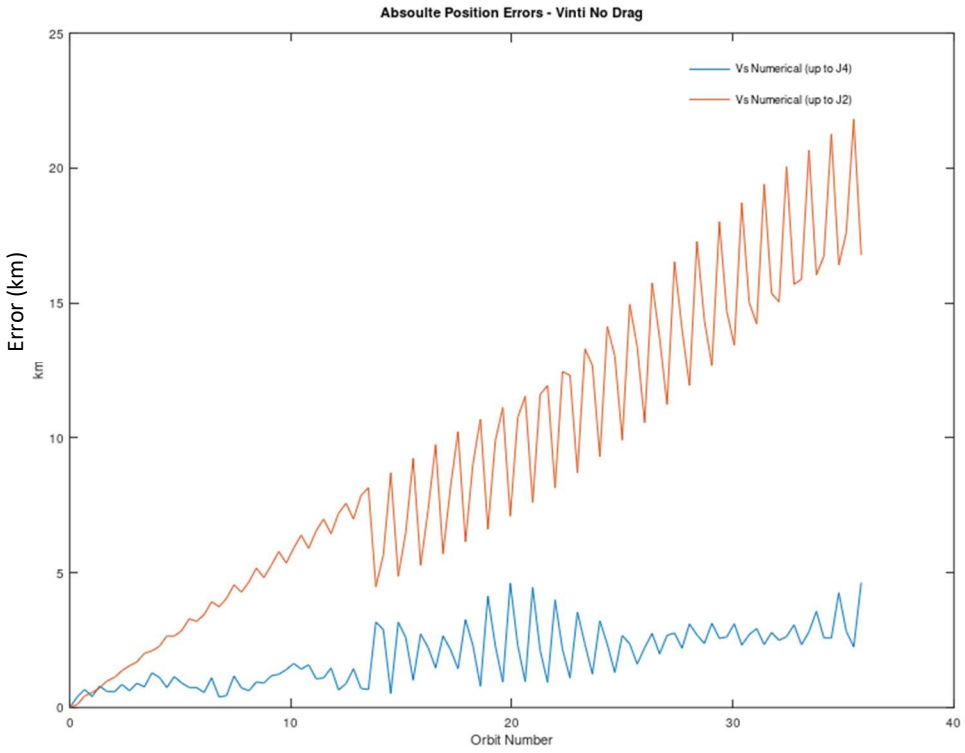

Figure 0.1: VOSM Compared to Only Geopotential Model

As expected, VOSM fits more closely with the J_4 geopotential model but not perfectly; as Vinti acknowledges, VOSM captures only about 75% of J_4 [26]. Note the violent oscillations beginning at 13 orbits. The exact cause of these oscillations is unknown but are likely due to differences in the highest order terms for which the J_2 and J_4 perturbations are accounted, leading to further decreased accuracy as the error increases. An analysis on the oscillations is beyond the scope of this thesis as the behavior in the first few orbits are of primary concern for initial downlink. The takeaway from Figure 3.1 is that VOSM is accurate to the J_4 geopotential model for up to 13 orbits. The next step is to add drag to the model. Note that other effects such as 3rd body, radiation, and tidal perturbations may be added but the researcher leaves that for future work.

To confirm the legitimacy of DEVS, a comparative study was performed by Branco and Senecal comparing DEVS without GPS assistance with the robust Lifetime analysis tool found in STK [11]. The same initial conditions were utilized as in Tables (1) and (2). Drag and satellite parameters are given in Table (3.3). The satellite was assumed to be tumbling to provide a worst-case scenario. Drag area is calculated as an average projected area for a 3U CubeSat. Carná and Bevilacqua detail a method of deriving the drag coefficient for CubeSats based on the Knudsen number, a description of the flow regime [48]. It is derived to be about 2.2 for the tumbling case in VLEO which is the standard practice in the literature [49]. While simulations can be made more accurate by refining this value further, this value gives a good estimation and allow for comparison of DEVS and other propagation methods. The mass is selected based on mission estimates.

Table 3.3: Drag Parameters

C_d	Reference Drag Area (S_{Ref})	Mass (m)
2.2	0.031 m^2	5.5 kg

The drag information can be combined into an SGP4 ballistic coefficient, B^*, given by Equation (25), where $\rho_0$is the reference atmospheric density at one Earth radius [50].

$$B^* = \frac{C_d S_{Ref} R_e \rho_0}{2m} \tag{25}$$

Using the drag parameters in Table 3.3, B^* is equal to $1.6434 * 10^{-7}$.

The resulting comparison in Figure 3.2 shows a promising result in terms of the apogee and perigee heights above mean sea level (MSL). DEVS outputs results that closely match the STK Lifetime tool.

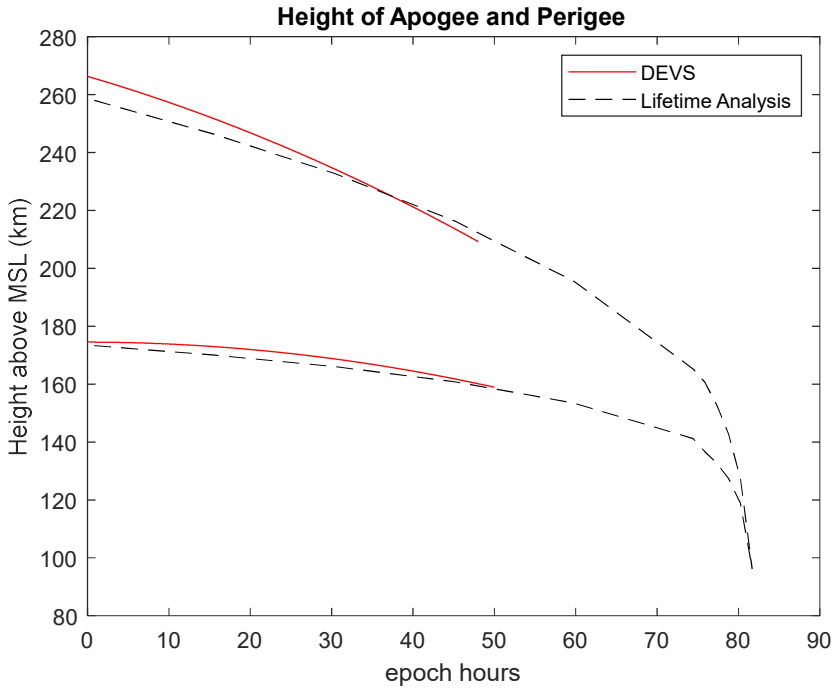

Figure 3.2: DEVS vs STK Lifetime Tool

Further analysis was performed to compare DEVS with SGP4 and to verify the drag implemented model. To do this, the accumulated position errors were calculated relative to truth model. This simulated truth model is computed in STK using the HPOP propagator. In this case, it was desired to analyze the true

accuracy of DEVS, thus the best model was chosen. Gravitational parameters included the full EGM2008 gravitation model, Sun and Moon 3rd body perturbations, solar effects, and the DTM2012 drag model. Figure 3.3 give the propagator settings, namely the initial conditions and drag characteristics, respectively.

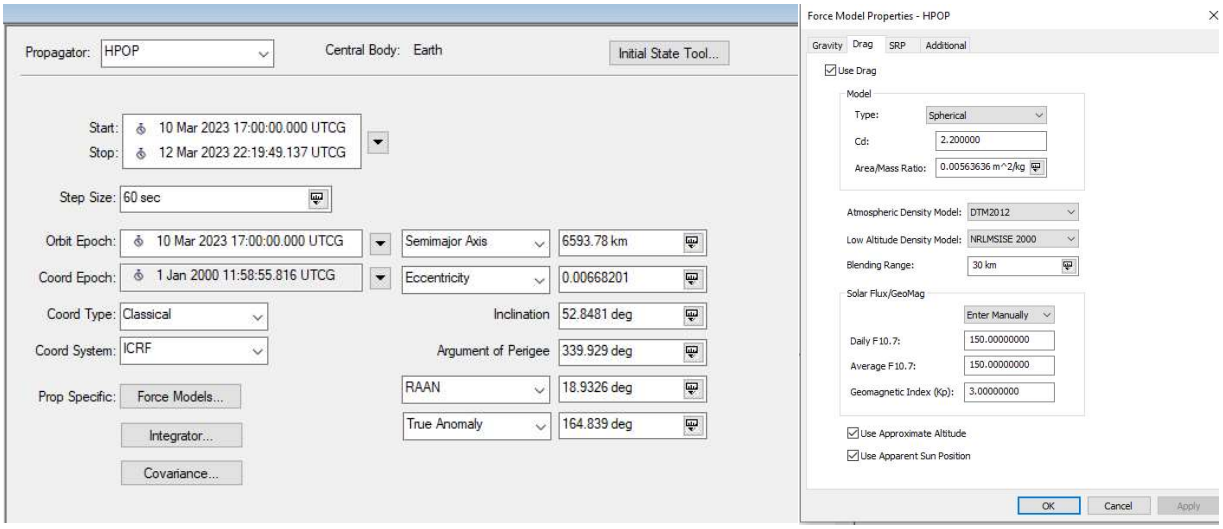

Figure 3.3: STK HPOP Configuration

The position errors, vector ($\boldsymbol{\delta r}$) and Root Mean Square (δr_{RMS}), are calculated by Equations (26) and (27) where subscripts N and V denote numerical and DEVS, respectively.

$$\boldsymbol{\delta r} = \boldsymbol{r}_v - \boldsymbol{r}_N \tag{26}$$

$$\delta r_{RMS} = \sqrt{\frac{\sum_{i=1}^{3}(\delta r)_i^2}{3}} \tag{27}$$

The period of the orbit is determined by Equation (28) where a ~ 6594 km, and μ = 398600.4 $\frac{km^3}{s^2}$

$$T = 2\pi\sqrt{\frac{a^3}{\mu}} \tag{28}$$

Thus, the orbital period is approximately 1.5 hours.

Figure 3.4 relates the performance of DEVS with SGP4 in terms of position error magnitude, $||\boldsymbol{\delta r}||$. DEVS and SGP4 yield similar performance but the advantage of DEVS is measurable. Error growth is nonlinear, and the error rates increase with time for both models. However, the first 5 hours of flight after a position fix, the orbit epoch, is a good indication of performance for the purpose of VLEO missions. At 5 epoch hours, the error of DEVS is 15 km while the error of SGP4 is 35 km; DEVS offers a 57% improvement over SGP4.

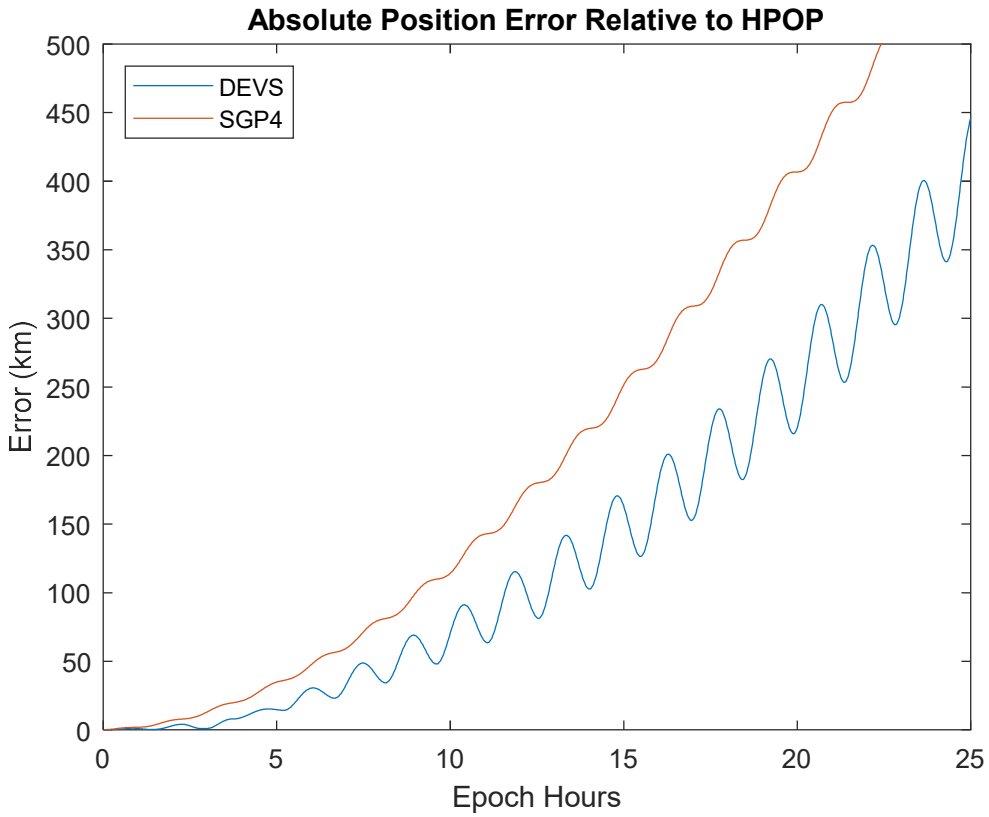

Figure 3.4: DEVS vs SGP4

Further confirmation of the drag implementation is desired. Thus, Figure 3.5 compares the RMS position errors of Vinti6 without drag (VOSM) and Vinti6 with drag (DEVS) relative to the most accurate

HPOP model. Figure 3.6 does likewise but comparing to the J_4 geopotential model with 1976 drag model. As clearly indicated, the drag implementation, albeit simple and not tightly integrated with VOSM, performs exceptionally well and greatly improves performance in VLEO.

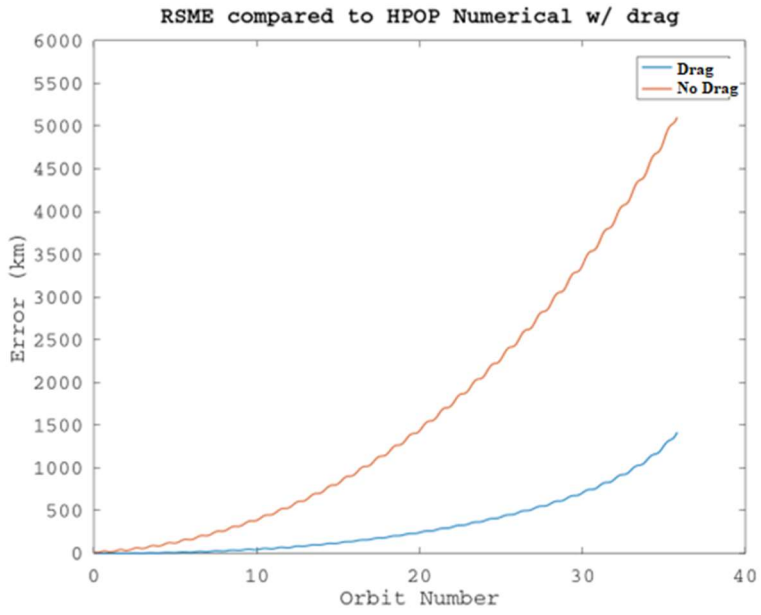

Figure 3.5: Position RMSE relative to HPOP – with and without Drag

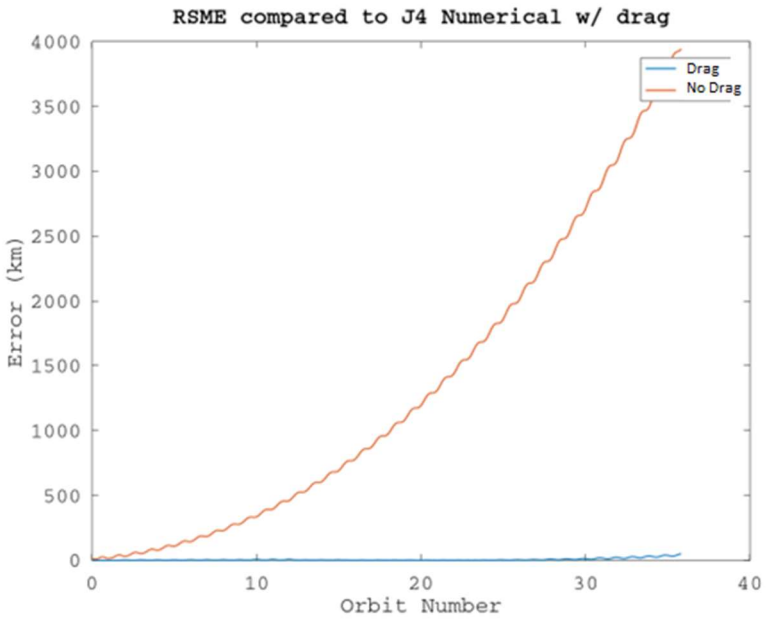

Figure 3.6: Position RMSE relative to J_4 – with and without Drag

Figures 3.7 and 3.8 compare the position and velocity magnitudes of HPOP and DEVS where once every 2 orbits, the state is reinitialized based on the HPOP state. This further demonstrates the ability of DEVS to match output comparable ephemerides as the truth model.

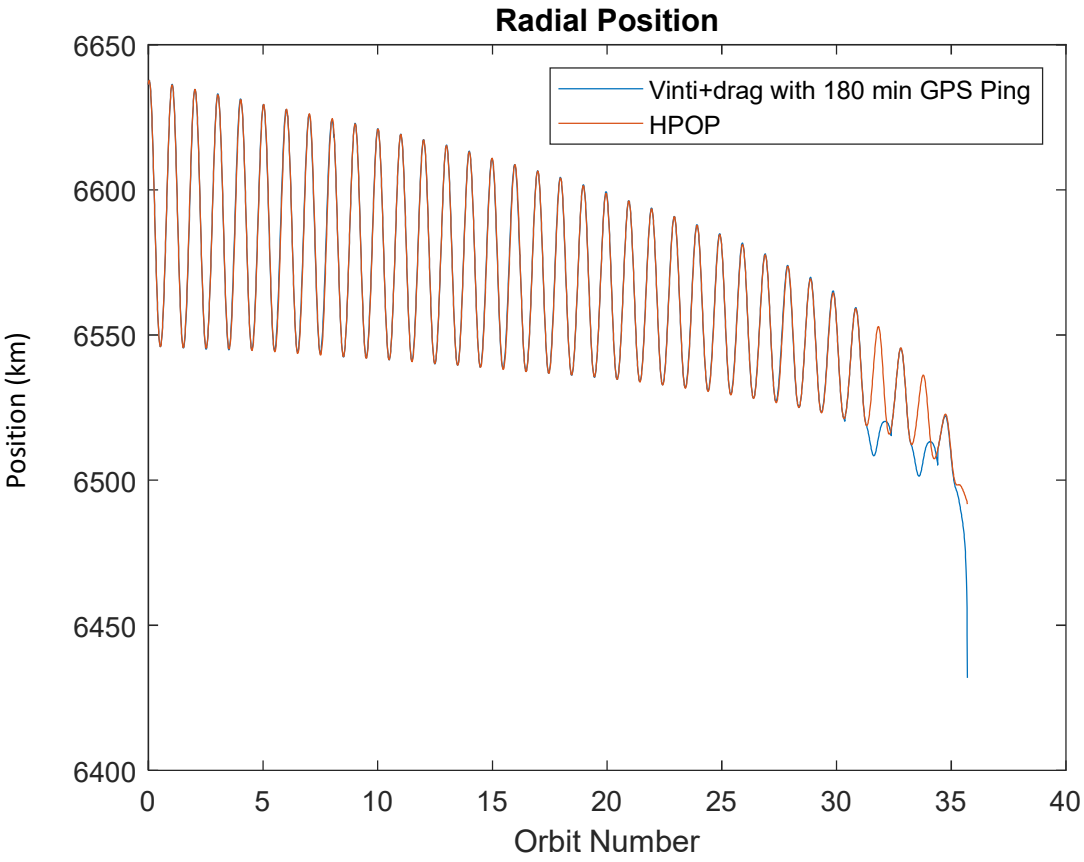

Figure 3.7: HPOP and DEVS Magnitude of Radial Position

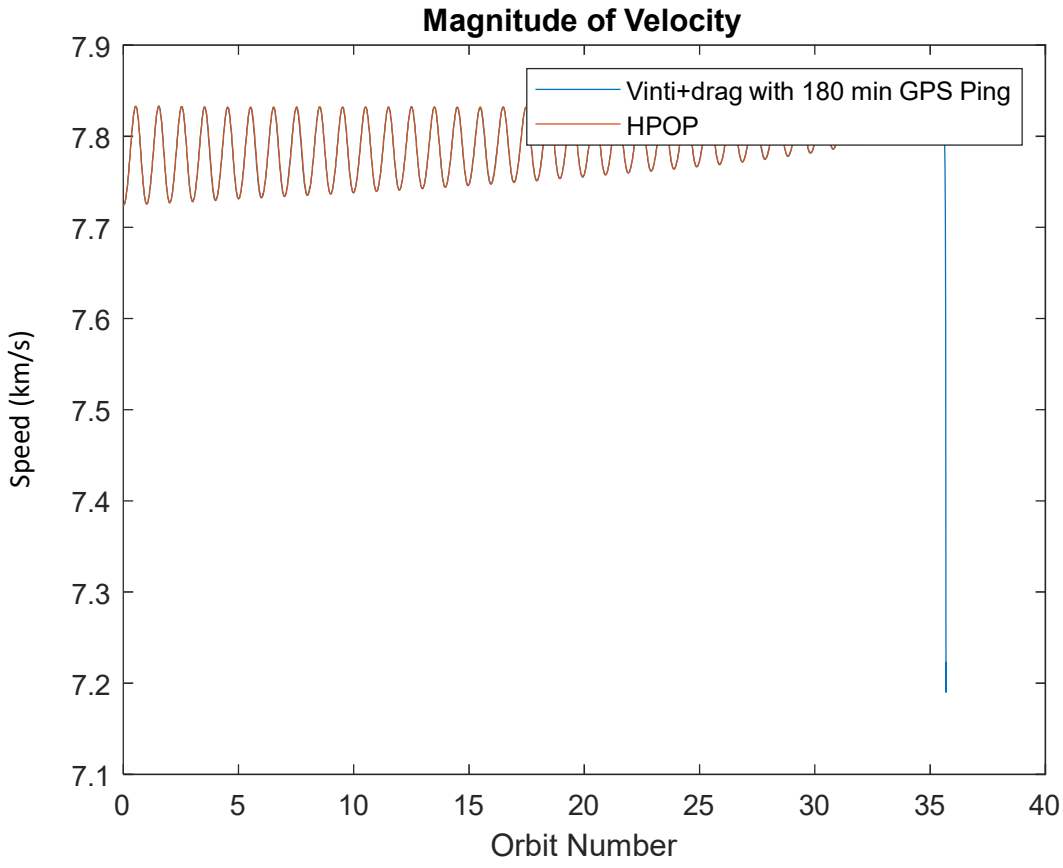

Figure 3.8: HPOP and DEVS Magnitude of Velocity

While DEVS is certainly a capable orbit predictor and superior to SGP4, especially in VLEO, this is not the end of the story. A satellite relying on only the DEVS algorithm to compute its next ground station pass for downlinking critical mission data will only have accurate flight information for about 5 hours. Additionally, despite the accuracy of the model, there are numerous uncertainties in actual flight conditions, namely ballistic coefficient, and solar effects. Thus, a GPS is implemented on the model satellite to refresh the orbital state of the satellite thereby reducing the in-orbit errors. The benefit of DEVS compared to SGP4, is that the GPS may be pinged less due to the higher accuracy of DEVS, thereby consuming less power – an imperative for a CubeSat mission.

3.2 Power and Accuracy

Several case studies are performed to investigate the prospective power savings DEVS offers a GPS equipped satellite. Figure 3.7 through 3.16 compare the propagated ephemeris data to that calculated by HPOP. The upper subplot gives the position error for each cartesian component of the position vector, $\boldsymbol{r}$, while the lower subplot shows the RMSE of position. Figure 3.7 gives a baseline error analysis with the satellite only obtaining an initial GPS fix (from HPOP) and then propagating. An expected megaphone pattern is seen as the error model becomes less certain as it predicts states further away from the determined initial state. This can be compared to the growth of the covariance matrix in a Kalman filter model.

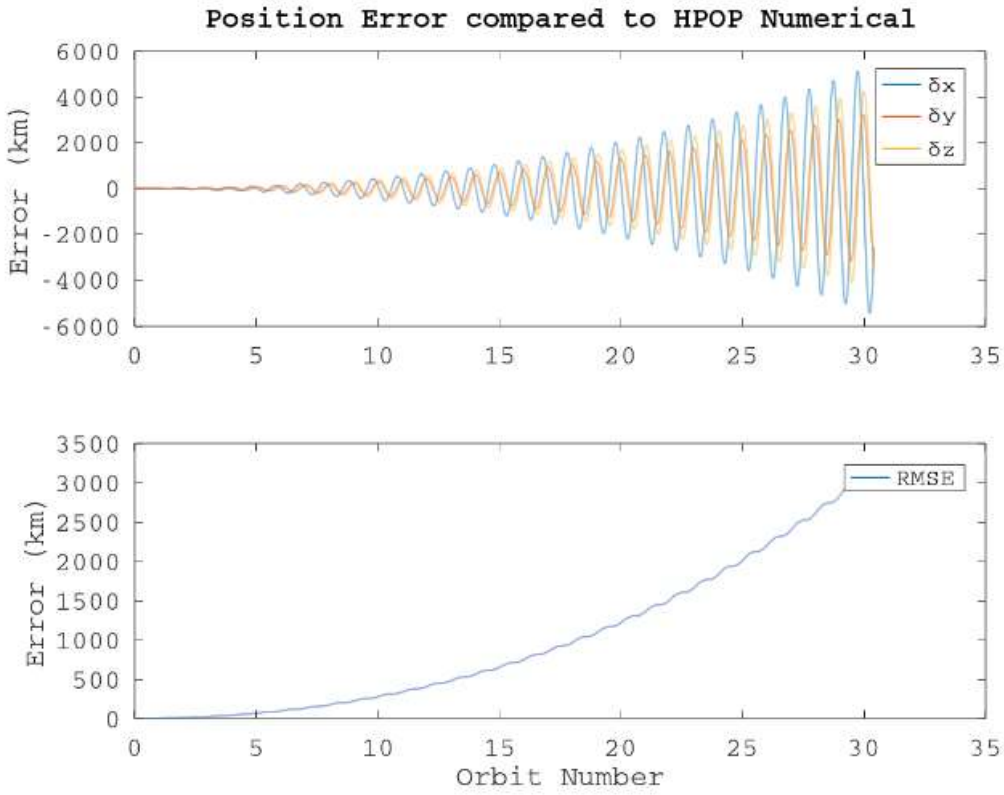

Figure 3.9: No GPS Ping

Figures 3.10 through 3.18 show how maximum position errors decrease with increasing GPS polling rate. As shown in Figure 3.10, if the GPS is pinged once every 360 min, which is 4 orbits, the accuracy of the propagator is greatly improved. However, maximum errors still reach about 50 km RMS. With a refresh rate of twice an orbit as seen in Figure 3.11, the propagator beings to enter the realm of acceptable errors at 12 km RMS. While too high for many practical purposes, the routine seems to behave well with no abnormalities. Expected behavior is seen of peaks and troughs with periods matching the GPS rate. The drag model seems to be over or underpredicting at times leading to some short-term oscillations.

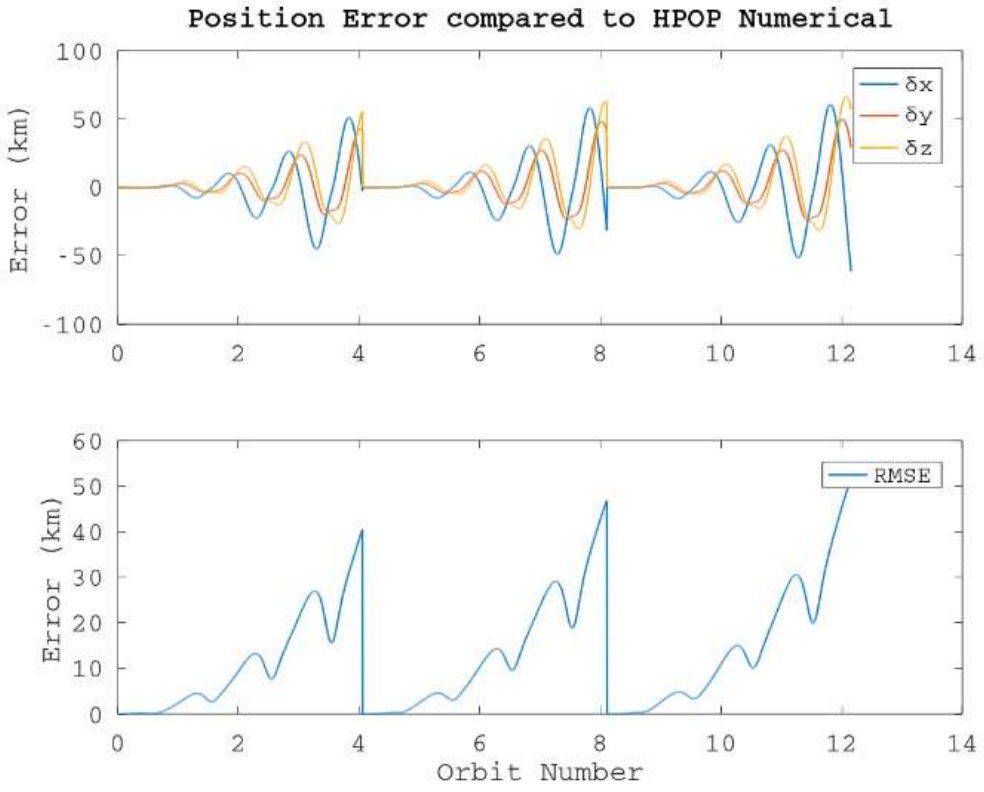

Figure 3.10: Ping GPS Every 360 Mins

Figure 3.11: GPS Pinged Every 180 Mins

Figures 3.11 and 3.12 show promising results for pinging the GPS only once an orbit, and once every hour, respectively. RMSE for these cases are between 1.3 and 2 km which may be low enough for many missions as shall be examined in section 3.3.

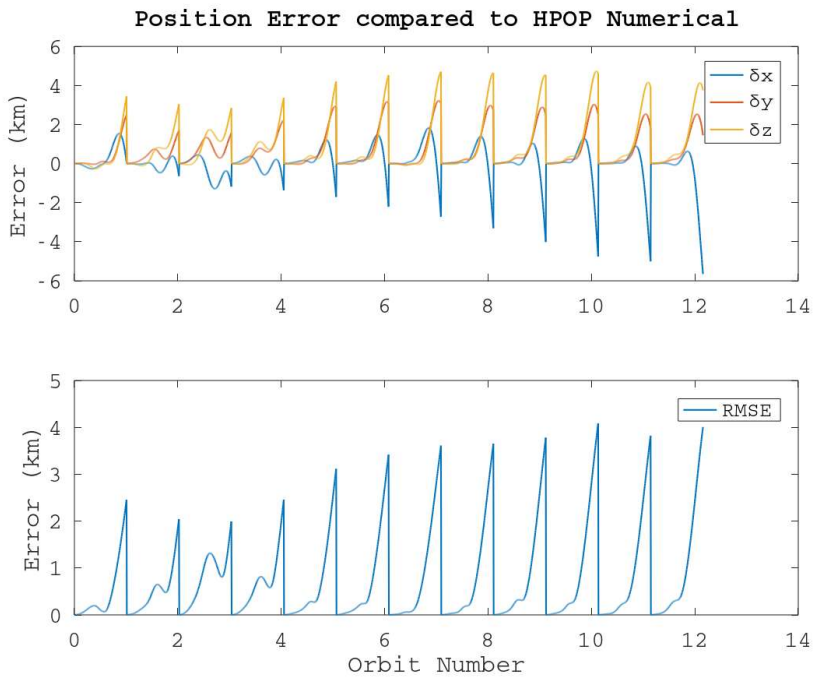

Figure 3.12: GPS Pinged Every 90 Mins

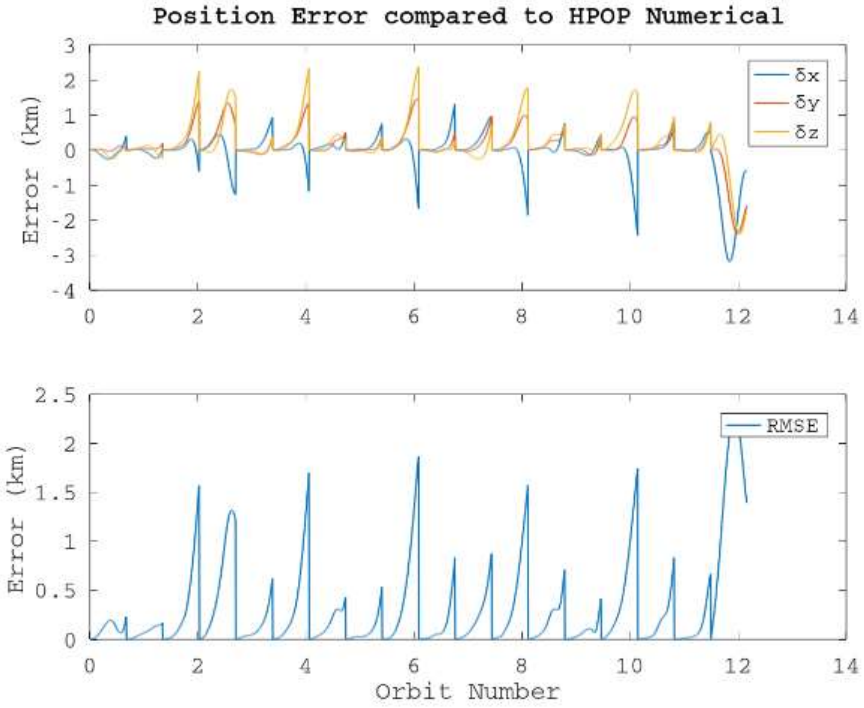

Figure 3.13: GPS Pinged Every 60 Mins

When the GPS period is 45 mins, the performance is once again improved with errors not exceeding 400 m for the first 10 orbits. However, large error spikes occur at around 12 orbits as seen in Figure 3.14. This behavior continues for increased GPS polling frequencies, Figures 3.15 through 3.18 although it occurs at different times. Also, it does decrease in magnitude; with a GPS period of 10 and 5 mins, the error spikes do not exceed 1 km RMSE. The issue of error spikes will be reexamined in Chapter 4.

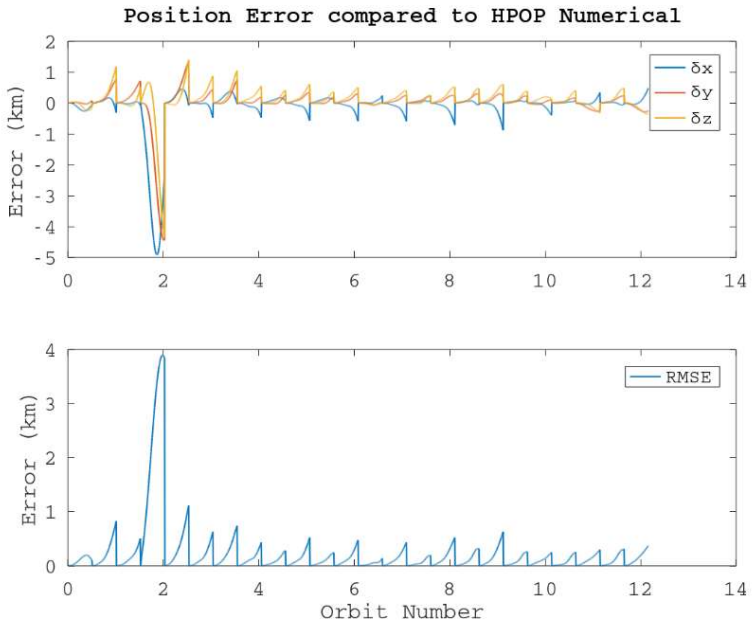

Figure 3.14: GPS Pinged Every 45 Mins

Figure 3.15: GPS Pinged Every 30 Mins

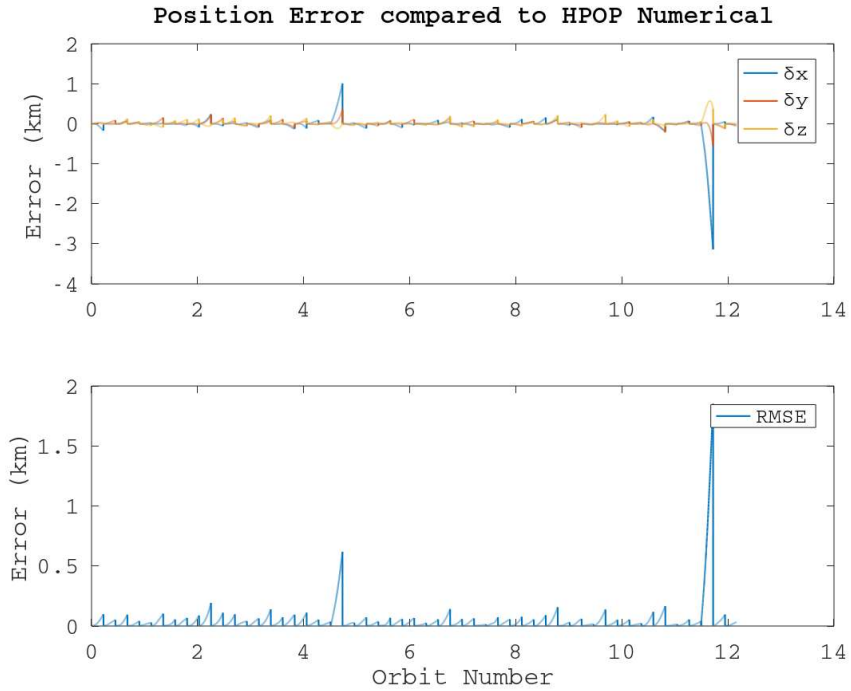

Figure 3.16: GPS Pinged Every 20 Mins

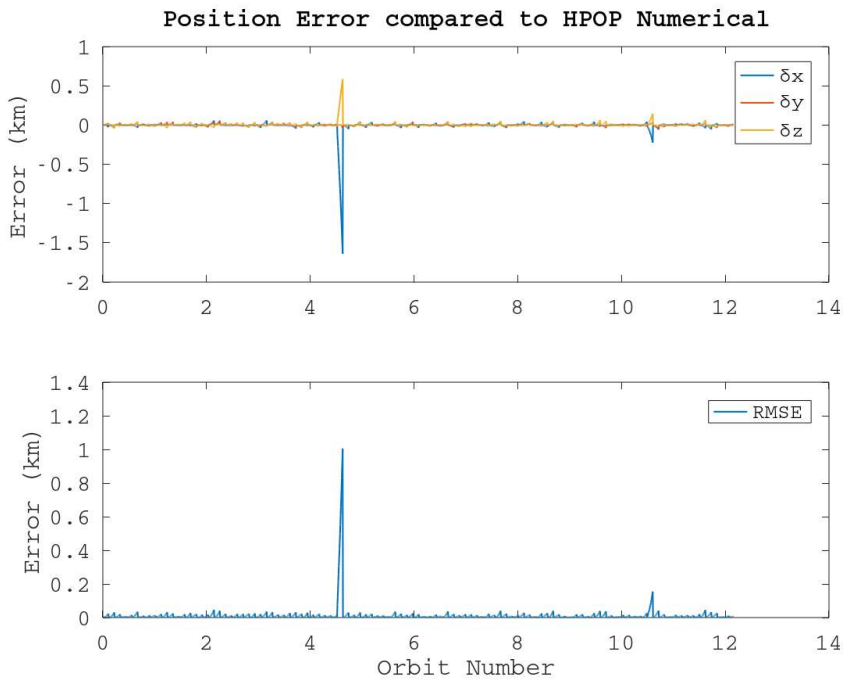

Figure 3.17: GPS Pinged Every 10 Mins

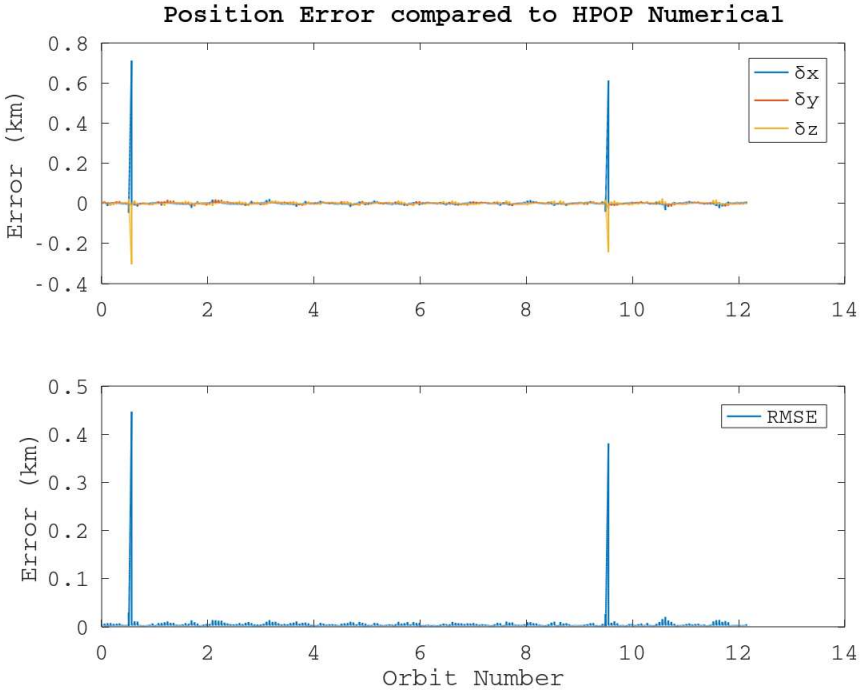

Figure 3.18: GPS Pinged Every 5 Mins

As shown in this error analysis, DEVS exhibits favorable performance with reasonable max errors. For comparison, the selected GPS receiver for SeaLion is the Venus838FLPx, a small, low power, CubeSat GPS antenna-receiver package with specifications given in Table 3.4 [51]. Circular Error Probable (CEP) accuracy denotes that the measured GPS location will be within 2.5 m of the true location 50% of the time, equating to a 3σ value of 6.37 m [52, 53]. Now comparing this value to Figures 3.17 and 3.18, GPS polling periods of 10 mins and 5 mins led to max errors of 40 m and 15 m, respectively. Thus, DEVS does well in propagating between pings for these cases with only slightly more errors than the selected GPS receiver.

Table 3.4: Venus838FLPx Specifications

Power Draw (mW)	Cold Start TTFF (s)	Hot Start (s)	CEP Accuracy (m)
110	29	1	2.5

Figures for cold, and hot starts are presented. For each case study the average GPS power draw is reported as given in Equation (29). Corresponding maximum errors are presented, ignoring brief outlier spikes. Table 3.5 reports this data. The first data column gives the power draw of using only GPS and no propagator, running the receiver continuously. Drastic power savings are observed; assuming a cold start for each ping as a worst-case scenario, values range from 10.6 mW to 0.1 mW for 5 minute and 4 orbit GPS refresh rates, respectively. Corresponding RMS errors range from 15 m to 50 km.

$$P_{avg}\left(\frac{mWhr}{hr}\right) = n_{\frac{pings}{hr}} * P_{GPS}\,(mW) * \Delta T_{GPS}\,(hr) \tag{29}$$

Table 3.5: Summary of Power Draw and Associated RMSE

GPS fix Period	Continuous	5 mins	10 mins	45 mins	1 orbit	2 orbits	4 orbits
$P_{avg_{cold}}\ (mW)$	110	10.63	5.31	1.18	0.591	0.295	0.148
$P_{avg_{Hot}}\ (mW)$	110	0.367	0.183	NA	0.0204	NA	NA
$RMSE\ (km)$	0.006	0.015	0.04	0.25	2	12	50

Note that a hot start is not available after 2 and 4 orbits (3 and 6 hours) since the criteria dictate that the last GPS fix may not be older than 2 hours [54]. As GPS refresh rate increases, hot start power draws more accurately describe true power draw from the GPS. Thus, for the case of 5-minute and 10-minute GPS polling rates, expected power usage is 0.367 mW and 0.183 mW. When the GPS is refreshed twice an orbit, a hot start is again impossible because the satellite is not within 100 km of the previous fix [54]. Therefore, of the cases presented, there are four candidate GPS periods: 5 minutes, 10 minutes, 1 orbit, and 4 orbits.

Power draw for the propagator itself was not simulated but can be done based on satellite hardware. The simulation was run on a 6 core/12 thread intel i5-10600kf processor. Computer Processor Unit (CPU) time for the propagator routine averaged 1.5 milliseconds (ms) for the deterministic model. CPU energy usage may be estimated by Equation (30), where f is a factor giving the performance of a given CPU relative to that used in this simulation.

$$E_{cpu} = P_{cpu} * \Delta t_{cpu} * f \tag{30}$$

In the Pumpkin Sat study, running the Vinti7 routine was determined to draw 1% of the power compared to obtaining a location fix from the satellite's Pumpkin GPSRM 1 [33]. Accounting for the difference in GPS power draws for the GPSRM 1 and the Venus838FLPx, the DEVS routine still uses only about 10% of power draw compared to the Venus838FLPx receiver.

3.3 Communications

It is of interest to examine how errors associated with increased reliance on DEVS will affect the communications between a satellite and the ground station network. If the satellite predicts that it is within the swath of a ground station when it is not, a false positive, it will attempt to transmit but the ground will receive nothing. On the other hand, if the propagator incorrectly computes that it is outside the swath, a false negative, the satellite will miss its chance to downlink data to the ground station. Both situations are to be avoided because a false positive results in increased power usage by needless use of the transmitter while a false negative results in a failed mission as communication with the satellite has not been established.

A study in STK 12.2 is performed using the S-band capabilities of the MC3 Ground Station Network, comparing access times and link budgets for state data predicted by DEVS and HPOP. Access times refer to time intervals during orbit that a satellite has line of site to a ground station. Link budgets are a measure of the quality of signal, allowing satellite operators to predict when a ground station will detect and receive a satellite signal, or vice versa. The aforementioned HPOP satellite was utilized in this scenario. Satellite ephemeris data, propagated using the presented DEVS algorithm, was imported into STK using the external propagator option. Analysis was performed using simulated S-Band communications. The transmitter on board the CubeSat was the EnduroSat S-Band Transmitter with 15 mega bit per second (mbps) transmit rate, 5 megabaud (MBd) symbol rate, and 32 dBm output power [55]. Effective antenna gain was set to 8 dB based on analysis by Amine and Minaoui of typical S-band antenna performance for CubeSats in Leo [56]. The transmission frequency was set to 2.2 gigahertz (GHz) and the ground station gains were set to 33 dBi per specifications given for several MC3 S-band receivers by Alcaide [57]. An image from the STK scenario is given in Figure 3.19 showing the orbital path as predicted by HPOP in white and DEVS in red. Lines of access are shown in green for HPOP and gold for DEVS. The DEVS satellite data was that for the 5-minute case. All MC3 ground station receivers were

placed in the scenario file at a height above ground of 18 ft as an estimate of antenna placement. The STK terrain server was utilized to calculate interruptions in satellite-to-ground station line of site due to terrain.

Figure 3.19: STK Scenario Image – Comparing Orbital Path and Accesses

Complete Chain Accesses (access times for the MC3 network as a whole) and Link Budget graphs and reports were generated for several interest cases using STK. A 12-hour time interval was selected as it is more than adequate to determine whether an initial data downlink to the ground station of location data can be performed. As a baseline of comparison, the access intervals predicted by the numerical integrator, HPOP are first presented in Figure 3.20 and Table 3.6. An access graph for 5-minute refresh rate is presented in Figure 3.21. Longer Graphs for GPS polling rates are not shown because they would look identical to Figures 3.20 and 3.21. However, access reports for DEVS refreshed every 5 minutes, 30 mins, and 4 orbits are given in tables, 3.7, 3.8, and 3.9. The access durations and access start and end times are compared. The x-axis shows the time in Coordinated Universal Time (UTC) and the green lines

denote access intervals. As shown, with increasing GPS frequencies, predicted access become closer to true accesses, at the price of higher power usage as seen in Section 3.2. Additionally, with a 5-minute GPS polling period, the access intervals appear identical to those of HPOP, compare Figures 3.20 and 3.21. Taking a closer look by comparing access reports, Tables 3.6 and 3.7, predicted total duration of access availability is less than 2 seconds off. Start and stop times are within a second for each access interval.

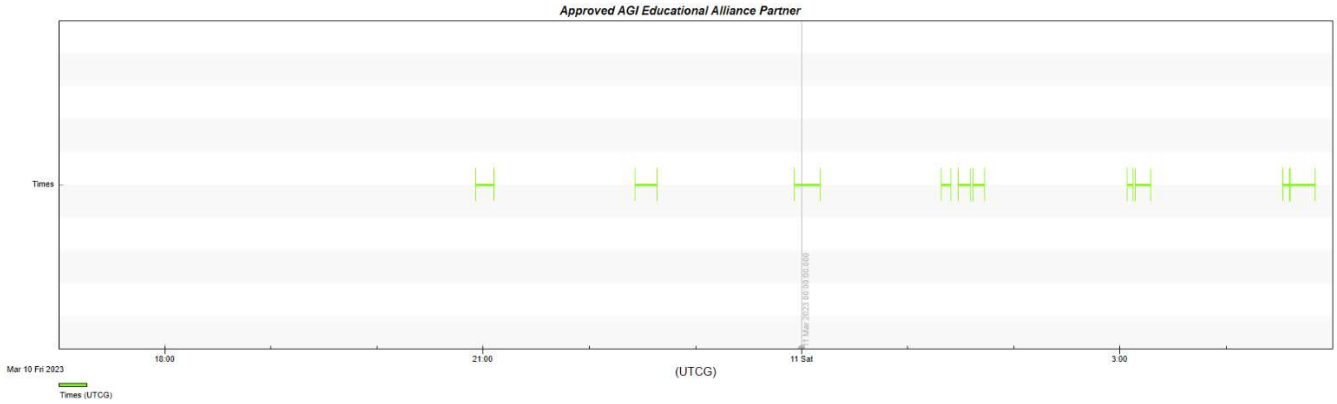

Figure 3.20: Complete Chain Access Intervals – HPOP

Table 3.6: Complete Chain Accesses – HPOP

Access	Start Time (UTCG)	Stop Time (UTCG)	Duration (sec)
1	10 Mar 2023 20:55:45.869	10 Mar 2023 21:06:03.598	617.729
2	10 Mar 2023 22:25:54.017	10 Mar 2023 22:38:20.156	746.139
3	10 Mar 2023 23:55:54.785	11 Mar 2023 00:10:22.124	867.339
4	11 Mar 2023 01:19:00.179	11 Mar 2023 01:24:23.722	323.543
5	11 Mar 2023 01:28:30.751	11 Mar 2023 01:35:22.007	411.256
6	11 Mar 2023 01:36:36.164	11 Mar 2023 01:43:33.070	416.906
7	11 Mar 2023 03:03:52.139	11 Mar 2023 03:07:19.744	207.604
8	11 Mar 2023 03:08:34.826	11 Mar 2023 03:17:24.813	529.987
9	11 Mar 2023 04:31:48.972	11 Mar 2023 04:35:40.228	231.255
10	11 Mar 2023 04:36:07.811	11 Mar 2023 04:50:13.176	845.365
Global Statistics			
Min Duration	7 11 Mar 2023 03:03:52.139	11 Mar 2023 03:07:19.744	207.604
Max Duration	3 10 Mar 2023 23:55:54.785	11 Mar 2023 00:10:22.124	867.339
Mean Duration			519.712
Total Duration			5197.123

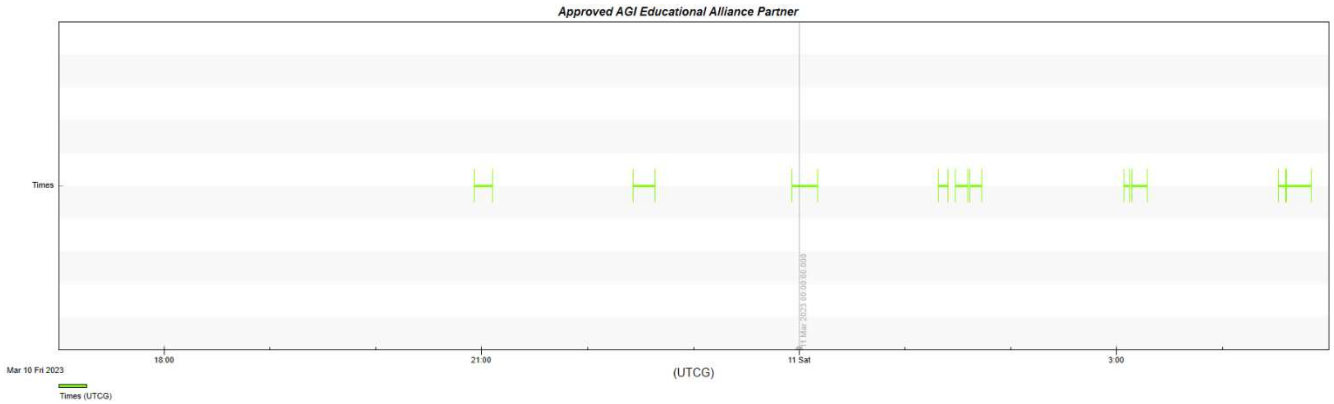

Figure 3.21: Complete Chain Access Intervals – DEVS, 5 min GPS Period

Table 3.7: Complete Chain Accesses – DEVS, 5 min GPS Period

Access	Start Time (UTCG)	Stop Time (UTCG)	Duration (sec)
1	10 Mar 2023 20:55:45.516	10 Mar 2023 21:06:03.289	617.773
2	10 Mar 2023 22:25:54.214	10 Mar 2023 22:38:20.112	745.898
3	10 Mar 2023 23:55:54.595	11 Mar 2023 00:10:22.329	867.734
4	11 Mar 2023 01:19:00.636	11 Mar 2023 01:24:23.184	322.548
5	11 Mar 2023 01:28:31.181	11 Mar 2023 01:35:21.997	410.816
6	11 Mar 2023 01:36:36.311	11 Mar 2023 01:43:32.746	416.435
7	11 Mar 2023 03:03:52.179	11 Mar 2023 03:07:19.695	207.516
8	11 Mar 2023 03:08:34.704	11 Mar 2023 03:17:24.677	529.973
9	11 Mar 2023 04:31:45.582	11 Mar 2023 04:35:40.250	234.668
10	11 Mar 2023 04:36:07.865	11 Mar 2023 04:50:13.255	845.390
Global Statistics			
Min Duration	7	11 Mar 2023 03:03:52.179	11 Mar 2023 03:07:19.695	207.516
Max Duration	3	10 Mar 2023 23:55:54.595	11 Mar 2023 00:10:22.329	867.734
Mean Duration				519.875
Total Duration				5198.750

The changes in access intervals between the 30 and 5 minute GPS cases are imperceptible. Table 3.8 presents the data with 1 second higher total predicted access duration and near identical access times. This indicates that communcation performance is most likely not hindered by infrequently refreshing the orbit via GPS.

Table 3.8: Complete Chain Accesses – DEVS, 30 min GPS Period

Access	Start Time (UTCG)	Stop Time (UTCG)	Duration (sec)
1	10 Mar 2023 20:55:45.667	10 Mar 2023 21:06:03.488	617.821
2	10 Mar 2023 22:25:54.099	10 Mar 2023 22:38:20.189	746.090
3	10 Mar 2023 23:55:54.845	11 Mar 2023 00:10:22.138	867.293
4	11 Mar 2023 01:19:00.657	11 Mar 2023 01:24:23.919	323.262
5	11 Mar 2023 01:28:31.035	11 Mar 2023 01:35:22.072	411.036
6	11 Mar 2023 01:36:36.465	11 Mar 2023 01:43:32.700	416.234
7	11 Mar 2023 03:03:50.858	11 Mar 2023 03:07:20.338	209.481
8	11 Mar 2023 03:08:35.099	11 Mar 2023 03:17:24.842	529.742
9	11 Mar 2023 04:31:45.724	11 Mar 2023 04:35:40.159	234.435
10	11 Mar 2023 04:36:07.884	11 Mar 2023 04:50:13.210	845.326
Global Statistics			
Min Duration	7	11 Mar 2023 03:03:50.858	11 Mar 2023 03:07:20.338	209.481
Max Duration	3	10 Mar 2023 23:55:54.845	11 Mar 2023 00:10:22.138	867.293
Mean Duration			520.072
Total Duration			5200.720

When the GPS period reaches 4 orbits, a degradation in access predictions is finally observed. DEVS in this case overpredicts the total access duration for the 12-hour interval by 53 seconds (Table 3.9), 1% of the total access time. It also predicts that the accesses occur later than they actually do. This could pose a problem. in particular looking at the shortest, access 7, the predicted access begins 3 seconds after and end 11 seconds after the true access. For the mission SeaLion scenario, this is not of much concern since because the data rate for the EnduroSat S-Band transmitter, at 15 Mbps, is sufficient to downlink ephemeris data in the 200 second interval. A sample ephemeris text file, containing 5 ephemerides with double precision and a header indicating other parameters is only 609 bytes (B) or 4.872 kilobits (kb). Thus, at the given data rate, it can be uplinked in a fraction of a second. A scenario could be envisioned where this would be an issue, such as a satellite equipped with a low data rate transceiver which with a lower altitude orbit than that of SeaLion. In any case, the DEVS tool proves reasonable access prediction.

Table 3.9: Complete Chain Accesses – DEVS, 4 Orbit GPS Period

Access	Start Time (UTCG)	Stop Time (UTCG)	Duration (sec)
1	10 Mar 2023 20:55:47.850	10 Mar 2023 21:06:08.065	620.216
2	10 Mar 2023 22:25:58.312	10 Mar 2023 22:38:29.627	751.315
3	10 Mar 2023 23:55:56.495	11 Mar 2023 00:10:25.862	869.366
4	11 Mar 2023 01:19:03.813	11 Mar 2023 01:24:28.260	324.447
5	11 Mar 2023 01:28:34.177	11 Mar 2023 01:35:27.351	413.174
6	11 Mar 2023 01:36:37.577	11 Mar 2023 01:43:39.989	422.412
7	11 Mar 2023 03:03:54.972	11 Mar 2023 03:07:30.367	215.395
8	11 Mar 2023 03:08:40.441	11 Mar 2023 03:17:37.141	536.700
9	11 Mar 2023 04:31:55.609	11 Mar 2023 04:35:58.265	242.656
10	11 Mar 2023 04:36:18.812	11 Mar 2023 04:50:33.326	854.514
Global Statistics			
Min Duration	7 11 Mar 2023 03:03:54.972	11 Mar 2023 03:07:30.367	215.395
Max Duration	3 10 Mar 2023 23:55:56.495	11 Mar 2023 00:10:25.862	869.366
Mean Duration			525.019
Total Duration			5250.195

Link margins for the ground receiver at the Naval Post Graduate School (NPS) are given in Table 3.10 for HPOP, and Table 3.11 for DEVS. The GPS polling period was 4 orbits in this case. This was chosen to test the limits of the propagator. Minelli, Magallanes, et. al., state that for an MC3 S-band receiver at NPS, the required signal carrier power is -94.5 decibel milliwatts (dBm) to obtain a target bit error rate of 10^{-5} [12]. Below this threshold, the receiver will not detect a useful transmission. Link times are given on the left column in UTC. Received signal power is given (in dBW) as well as the link margin in dB which describes if the signal will be detected; a negative margin implies that the signal strength is below the threshold. The report is once again pertaining to the first 12 hours of flight. The DEVS accurately predicts link times, received power, and link margins. The maximum difference in link margin is observed to be 0.15 dB for the last link in Tables 3.10 and 3.11. Thus, despite only pinging the GPS once per four orbits with max position errors of 50 km, DEVS predicts similar results in terms of communication availability as does HPOP, the truth model. This implies that the size of errors seen in section 3.2 will not adversely affect data downlink.

Table 3.10: Link Budget Report, HPOP to NPS

Time (UTCG)	Carrier Power at Rcvr Input (dBW)	Link Margin (dB)
10 Mar 2023 23:55:54.785	-123.001	1.4993
10 Mar 2023 23:56:24.000	-121.235	3.2651
10 Mar 2023 23:56:54.000	-119.255	5.2451
10 Mar 2023 23:57:24.000	-116.991	7.5094
10 Mar 2023 23:57:54.000	-114.527	9.9731
10 Mar 2023 23:58:24.000	-112.675	11.8248
10 Mar 2023 23:58:54.000	-113.046	11.4545
10 Mar 2023 23:59:24.000	-115.229	9.2712
10 Mar 2023 23:59:54.000	-117.663	6.8368
11 Mar 2023 00:00:24.000	-119.829	4.6712
11 Mar 2023 00:00:54.000	-121.717	2.7827
11 Mar 2023 00:01:24.000	-123.449	1.0509
11 Mar 2023 00:01:54.000	-125.299	-0.7991
11 Mar 2023 00:01:57.491	-125.552	-1.0520

Time (UTCG)	Carrier Power at Rcvr Input (dBW)	Link Margin (dB)
11 Mar 2023 01:28:30.751	-124.134	0.3665
11 Mar 2023 01:29:00.000	-122.921	1.5790
11 Mar 2023 01:29:30.000	-121.878	2.6216
11 Mar 2023 01:30:00.000	-121.068	3.4321
11 Mar 2023 01:30:30.000	-120.598	3.9016
11 Mar 2023 01:31:00.000	-120.568	3.9325
11 Mar 2023 01:31:30.000	-120.980	3.5197
11 Mar 2023 01:32:00.000	-121.743	2.7569
11 Mar 2023 01:32:30.000	-122.736	1.7643
11 Mar 2023 01:33:00.000	-123.890	0.6102
11 Mar 2023 01:33:30.000	-125.269	-0.7691
11 Mar 2023 01:33:48.947	-126.425	-1.9248

Table 3.11: Link Budget Report, DEVS for 4 Orbit GPS Period

Time (UTCG)	Carrier Power at Rcvr Input (dBW)	Link Margin (dB)
10 Mar 2023 23:55:56.495	-123.015	1.4845
10 Mar 2023 23:56:26.000	-121.235	3.2654
10 Mar 2023 23:56:56.000	-119.256	5.2441
10 Mar 2023 23:57:26.000	-116.993	7.5070
10 Mar 2023 23:57:56.000	-114.531	9.9692
10 Mar 2023 23:58:26.000	-112.679	11.8207
10 Mar 2023 23:58:56.000	-113.047	11.4535
10 Mar 2023 23:59:26.000	-115.227	9.2730
10 Mar 2023 23:59:56.000	-117.660	6.8399
11 Mar 2023 00:00:26.000	-119.825	4.6754
11 Mar 2023 00:00:56.000	-121.712	2.7884
11 Mar 2023 00:01:26.000	-123.439	1.0607
11 Mar 2023 00:01:56.000	-125.275	-0.7750
11 Mar 2023 00:02:00.324	-125.586	-1.0860

Time (UTCG)	Carrier Power at Rcvr Input (dBW)	Link Margin (dB)
11 Mar 2023 01:28:34.177	-124.157	0.3433
11 Mar 2023 01:29:04.000	-122.926	1.5740
11 Mar 2023 01:29:34.000	-121.887	2.6126
11 Mar 2023 01:30:04.000	-121.079	3.4213
11 Mar 2023 01:30:34.000	-120.608	3.8915
11 Mar 2023 01:31:04.000	-120.574	3.9257
11 Mar 2023 01:31:34.000	-120.982	3.5178
11 Mar 2023 01:32:04.000	-121.739	2.7608
11 Mar 2023 01:32:34.000	-122.725	1.7751
11 Mar 2023 01:33:04.000	-123.869	0.6313
11 Mar 2023 01:33:34.000	-125.224	-0.7242
11 Mar 2023 01:33:56.138	-126.570	-2.0695

CHAPTER 4

DISCUSSION AND IMPROVEMENT

4.1 Issues and Error Propagation

Prior to establishing the algorithm presented in section 2.2, the algorithm given in Figure 4.1 was developed. The difference between the two is that this original architecture updated the previous state and time, $\dot{\boldsymbol{X}}_0$ and t_0, after every prediction and propagated from it to the next state. Contrast that to Figure 2.2 in which the most recent GPS fix is used as the initial state for all propagated states until the next GPS fix. This one change drastically affects program performance.

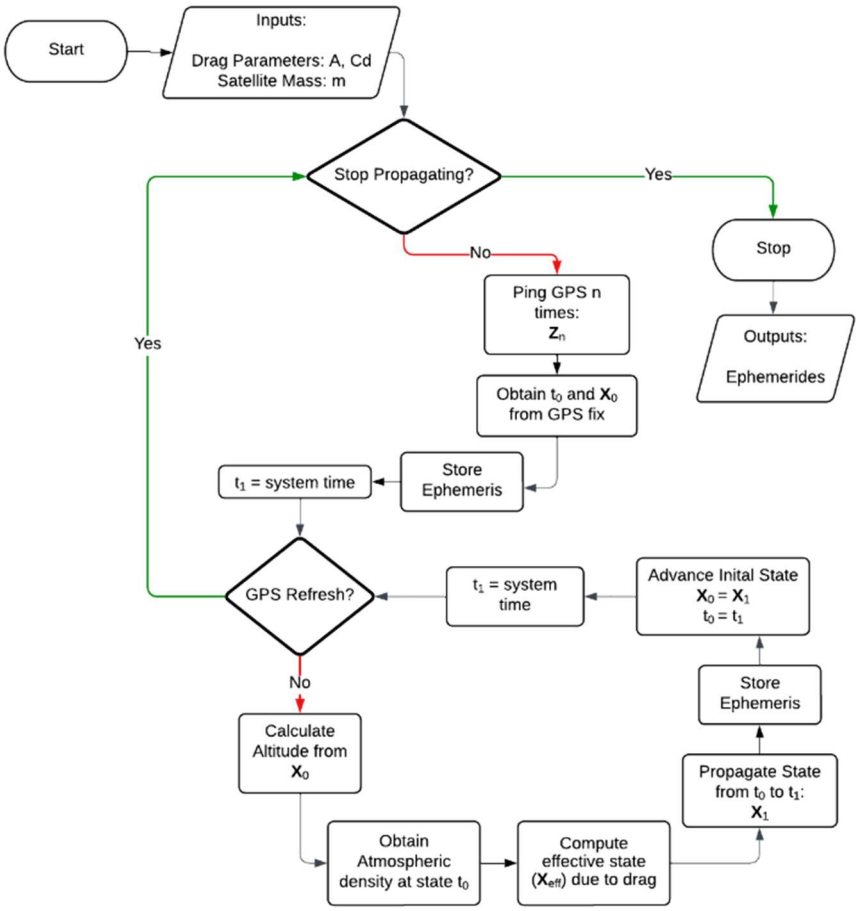

Figure 4.1: Original Orbit Propagator Architecture – High Error

The resulting error growth, due to propagated model errors and possibly truncation errors, is shown in Figures 4.2 and 4.3. The error is much higher than results from chapter 3 using the updated algorithm. Figure 4.2 shows a rapid climb in error for the unrefreshed case while Figure 4.3 shows that even when obtaining a GPS fix every 30 mins, the error quickly accumulates, periodically reaching unacceptable levels. The cause of this behavior lies in updating the state with a prediction value rather than a more accurate state estimate. The assumption inherently made via this approach, is that each state prediction has equal error to the original state obtained from the GPS. This is not the case because process noise and bias in the model cause error of a prediction to be larger than the initial state. With each prediction

dependent on the prior prediction, error will then accumulate rapidly. In any case, this initial version of the algorithm clearly had poor performance and was thus revised to the version presented in Figure 2.2.

Figure 4.2: Error Accumulation - No GPS

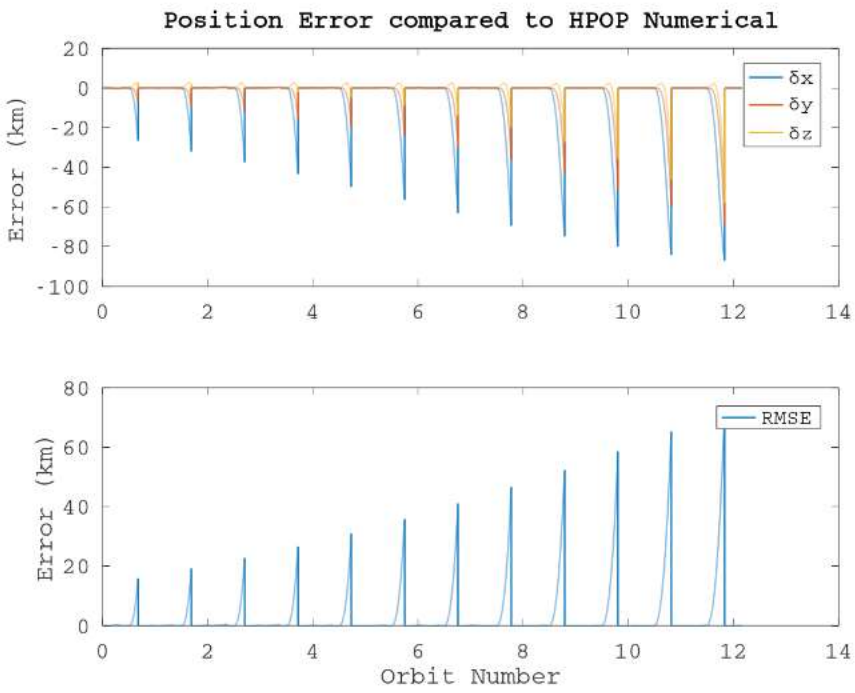

Figure 4.3: Error Accumulation - GPS Pinged Every 30 mins

4.2 Refinement by State Estimation

The investigation of error growth in section 4.1 uncovered another problem: model errors were not being accounted for or corrected. While the DEVS routine as given in Figure 2.2 is quite accurate, it does have errors in the model which can reach significant levels, spiking rapidly to values well over the mean. This phenomenon was ignored in the previous chapter for the sake of power analysis. Here, it is investigated. Some way to predict and compensate for model errors is desired, i.e., a state estimator. In the realistic case of noisy or biased GPS measurements, a Kalman filter is often used to reduce error in the state vector [58]. Such a filter can be implemented based on the typical algorithm as shown in Figure 4.4 below as detailed by Becker [58].

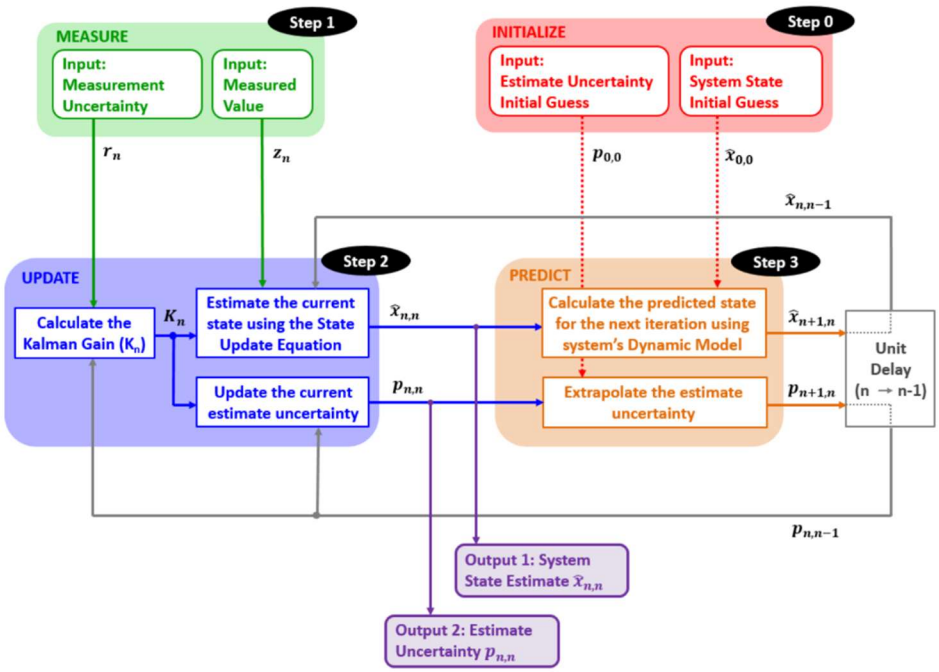

Figure 4.4: Kalman Filter Flow Chart [58]

In the case of this study however, errors in the model are found to be of much higher order of magnitude than the GPS errors: 2 km versus 6 m. Thus, a state estimator is employed that will determine and compensate for the model bias and remove outlying predictions.

The state estimator used here is not quite a Kalman filter, but it was inspired by Asundi's work on SwampSat in which he implemented an extended Kalman filter (EKF) to reduce bias in an attitude determination system [59]. While the DEVS algorithm is analytical and efficient, it is non-linear and involves several coordinate transformations. The system model is given in Equation (31) where f is the nonlinear function that obtains the velocity from forces, representing in effect, the DEVS procedure.

$$\frac{d}{dt}\begin{bmatrix}\boldsymbol{r}\\ \dot{\boldsymbol{r}}\end{bmatrix} = \begin{bmatrix}\dot{\boldsymbol{r}}\\ f(\boldsymbol{r},\dot{\boldsymbol{r}},t)\end{bmatrix} \tag{31}$$

Linearizing the model in accordance with an extended Kalman filter (EKF) is made difficult by the numerous transformations. Thus, implementing an analytic state transition matrix, ϕ, is not trivial and perhaps not worth the effort. Indeed, according to Der, a simple procedure can be followed to create a

state transition matrix using finite differencing which has low error compared to a numerically integrated transition matrix [27]. A similar approach was performed by Wright and was the impetus for pursuing this method [23]. Given state $\boldsymbol{X}_0$ at time t_0, the matrix is constructed as follows:

- Using the DEVS analytical solution, propagate the state by some time t_1 to state $\boldsymbol{X_1}$.
- Offset $\boldsymbol{X}_0$ by a small value, h_j for each element of state to obtain nearby states, X_{0_j} where $j = 1\ to\ 6$.
- Propagate nearby states with DEVS, obtaining $\boldsymbol{X}_{1_j}$.
- Compute normalized difference and insert into matrix, Φ, a 6x6 matrix with Equation (32).

$$\Phi = \begin{bmatrix} | \\ (\boldsymbol{X}_{1_j} - \boldsymbol{X}_1)/h \\ | \end{bmatrix} \tag{32}$$

This matrix then gives a way to predict how changes in one state change another state. Equation (33) therefore predicts how error changes between states.

$$\delta \boldsymbol{X}_1 = \Phi_1 \delta \boldsymbol{X}_0 \tag{33}$$

Therefore, the state estimate then is given in Equation (34).

$$\boldsymbol{X}_1^+ = \boldsymbol{X}_1^- - \delta \boldsymbol{X}_1^+ \tag{34}$$

Figure 4.5 presents the propagator algorithm with the state estimator. The routine is consistent with the updated propagator architecture in Section 2.2 with the addition of an error predictor that corrects the state in accordance with Equations (32) through (34). The routine first pings the GPS at least twice. Using DEVS, the first ping is propagated forward to the time of second ping. The residual is then computed between the prediction and GPS measurement. During the propagation phase where the satellite shuts off the GPS receiver, the error residual is predicted using the state transition matrix and the corresponding state corrected.

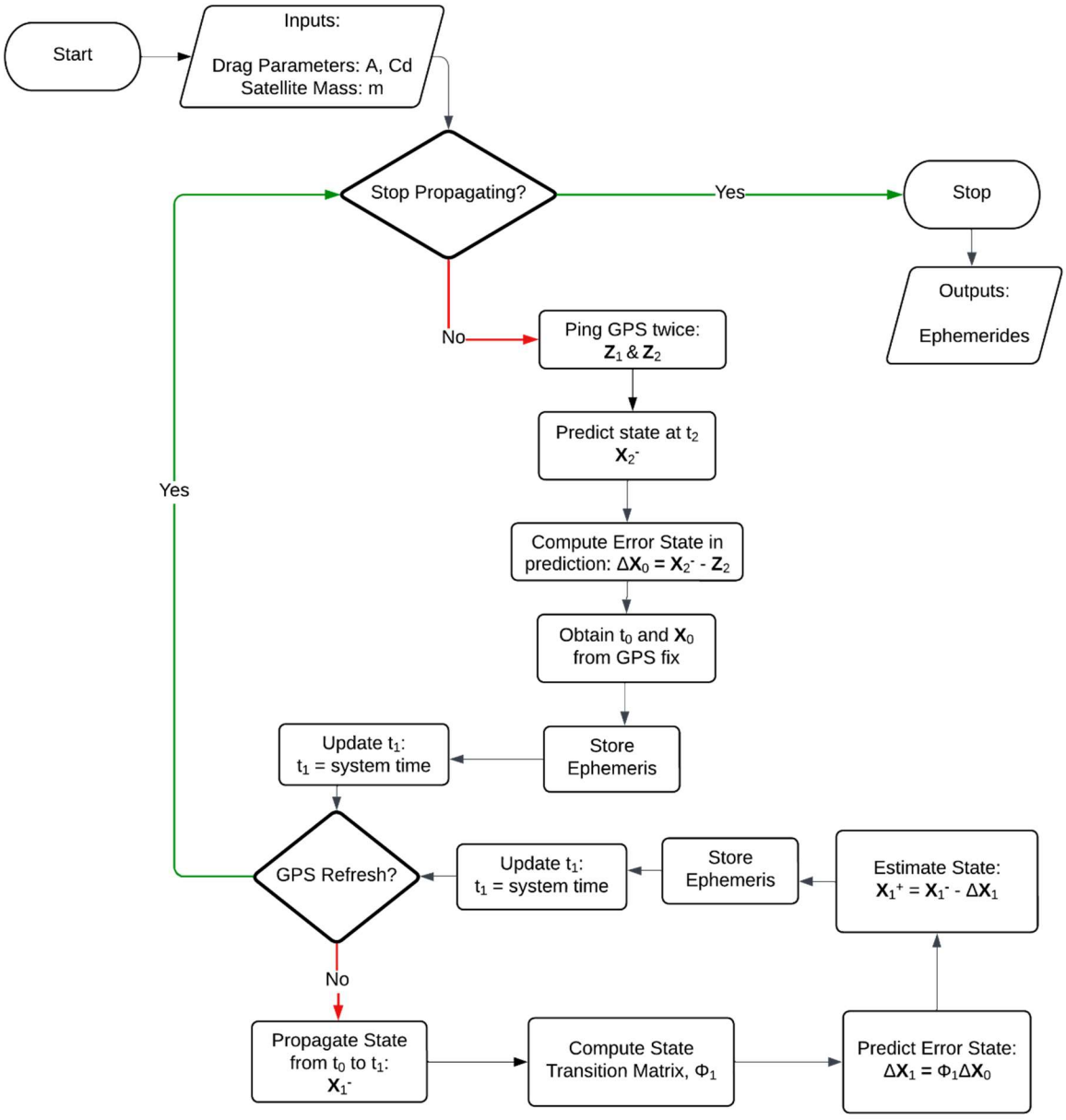

Figure 4.5: State Estimator Algorithm

4.3 Results with State Estimator

The performance of the state estimator is quite favorable. The estimator does not decrease long period errors, but it does reduce spikes and dampens short term oscillation. Figures 4.6 through 4.11 present the position error state and RMSE. Minor improvements can be seen when the GPS is

reacquired every hour – Figure 4.6 – with no peaks over 2 km. Significant reductions in outliers are seen as the GPS frequency increases. Figure 4.7 shows that for a 45-minute GPS period, the error maximum is reduced from 4 km to 1.1 km.

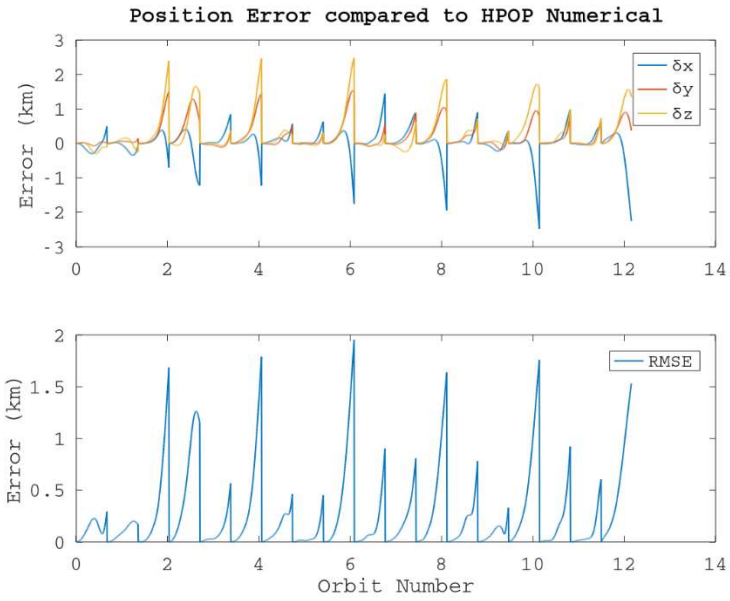

Figure 4.6: GPS Pinged Every 60 Mins

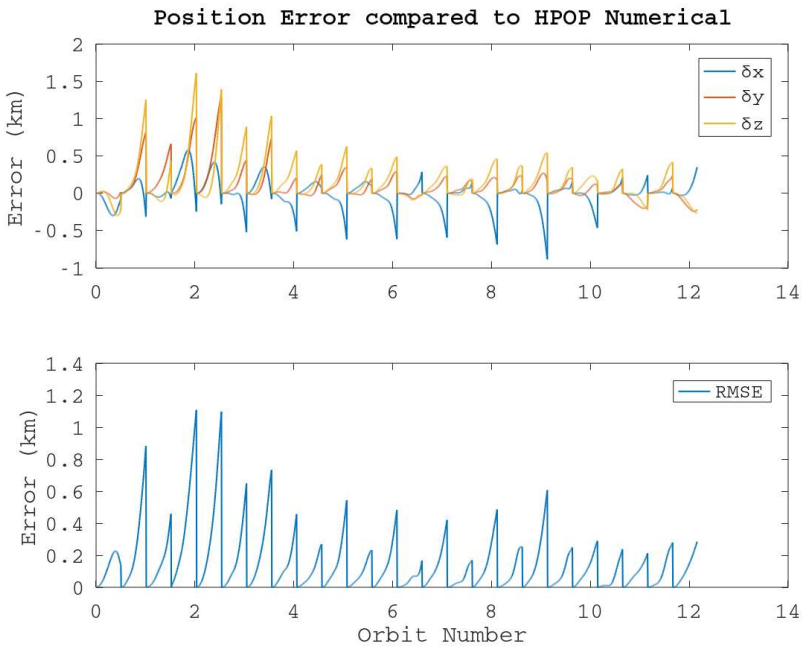

Figure 4.7: GPS Pinged Every 45 Mins

Figures 4.8 and 4.9 continue this trend where maximum errors for these GPS frequencies are decreased by an order of magnitude compared to their values without a state estimator. These cases are of interest because the errors have fallen well below a kilometer, on the order of 300 meters.

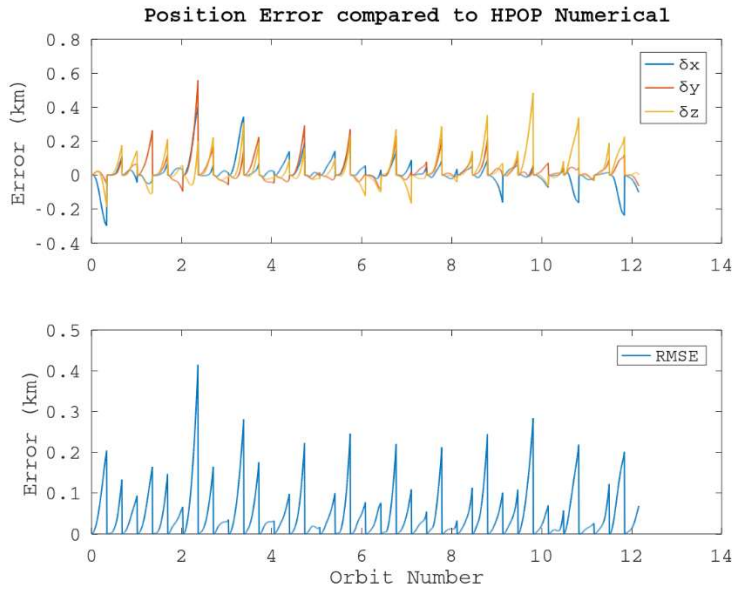

Figure 4.8: GPS Pinged Every 30 Mins

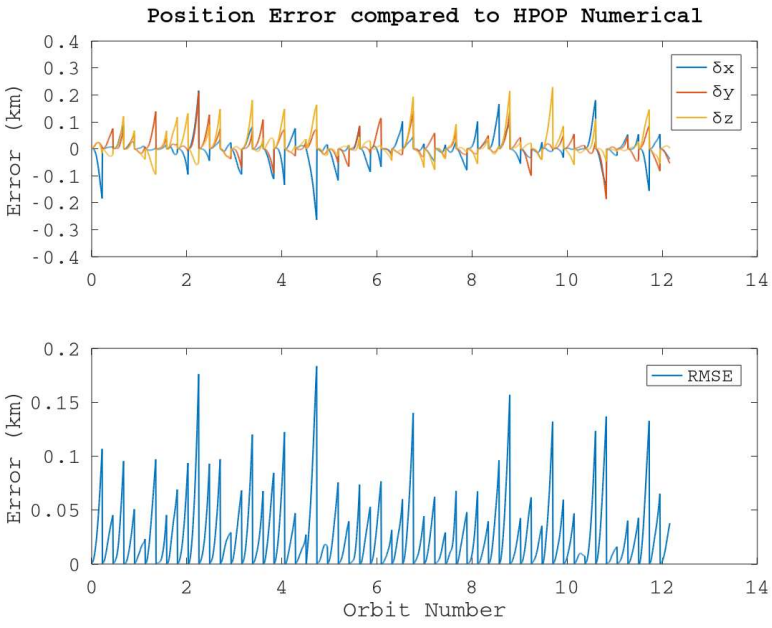

Figure 4.9: GPS Pinged Every 20 Mins

Finally, in Figures 4.10 and 4.11, the maximum errors arrive to nearly the same value as the GPS uncertainty of 6 m. With a 10-minute GPS period, there are minimal outliers with a maximum of 70 meters, very nearly the average error value. The 5-minute GPS case retains one large outlier of 70 meters. Both cases improved by an order of magnitude or more as compared to with no state estimator.

Figure 4.10: GPS Pinged Every 10 Mins

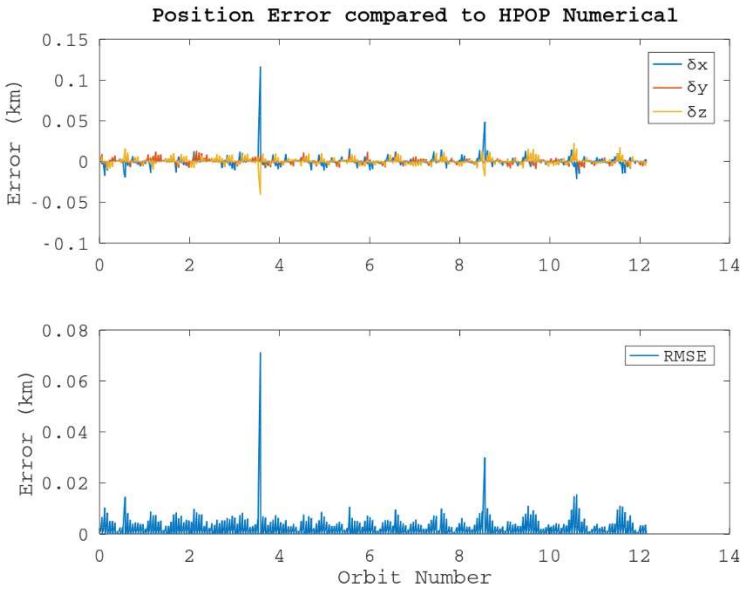

Figure 4.11: GPS Pinged Every 5 Mins

Table 4.1 summarizes the state estimator's improvements in outlier error compared to the uncorrected states in chapter 3. Maximum RMSE is given for each case. Across the board, errors spikes have been drastically reduced. An interesting observation is that the maximum outliers are equal between the 5-minute and 10-minute GPS cases. While not ideal, it will not pose much of an issue because: the outliers are small and brief, barely impacting communications; the satellite computer can include a routine to ignore large spikes; and finally, parallel orbit propagation on the ground will be performed once the satellite has downlinked ephemerides, adding resiliency in orbit prediction.

Table 4.1: Comparison of Maximum RMSE with and without State Estimator

GPS fix Period	5 mins	10 mins	20 min	30 mins	45 mins	60 mins
$RMSE\ (km)$ – No State Estimator	0.45	1	1.75	1.75	4	2.1
$RMSE\ (km)$ - State Estimator	0.07	0.07	0.18	0.42	1.1	1.95

The propagation subroutine was again timed giving a 13 ms runtime for every cycle. While a significant increase in runtime, it is to be expected since the state transition matrix must be solved every iteration. This can be easily improved by computing the transition matrix only every 10 or 30 iterations since it should stay relevant for some time prior to decay.

CHAPTER 5

CONCLUSION AND FUTURE WORK

An onboard orbit propagation tool has been developed for GPS equipped CubeSats in VLEO based on the Vinti Oblate Spheroidal Method (VOSM). A drag model was implemented to account for the high drag regime of VLEO. The analysis examined four components:

- Accuracy of the model, including comparison to SGP4, using the High Precision Orbit Propagator (HPOP) as truth model,
- Sparse utilization of GPS, i.e., accuracy vs power,
- Effects of decreased GPS utilization on communications, and
- Effects of a state estimation on accuracy.

The DEVS propagator proved more accurate than SGP4 in such environments, with errors only 43% of SGP4 errors five hours after orbit epoch as seen in Section 3.1. The higher accuracy of DEVS is due to higher fidelity modeling of zonal harmonics found in Vinti6 compared to SGP4. DEVS includes higher order terms of the second zonal harmonic than SGP4.

The improvement in accuracy allows for further decreased power draw by reducing GPS utilization with minimal impact on ephemeris accuracy. Table 3.5 in Section 3.3 summarizes accuracy and power draw for various GPS frequencies and operation modes. Large power savings may be observed even with a 5-minute GPS refresh period: average power draw of 0.61 mW using a hot start draw. This use case only increases the maximum error to 15 meters for the GPS device used in the scenario. Communications are not noticeably affected by these minor errors and this performance holds even when errors reach 50 km. The implication of these results is that the drag equipped Vinti spheroidal method (DEVS) is a capable orbit propagator for use in VLEO resulting in a highly accurate, energy efficient alternative to other propagators or the sole use of GPS. Access intervals can be

accurately predicted by the satellite allowing it to downlink critical mission data to the ground station promoting mission success.

A potential area to investigate further is the modification of the state estimator to account for noise and bias in GPS observations. While the author determined that GPS errors were of less concern than model bias, errors in GPS could become significant for some missions. Furthermore, DEVS was highly accurate for high frequency GPS reacquisition rates. If for example, the GPS is pinged every 2 or 3 minutes, the errors in GPS measurements could be greater than model errors. Thus, expected satellite operation must be assessed to determine an appropriate extension of the state estimator.

Due to changes in the SeaLion mission, the presented method was not implemented on an actual satellite. In future work, integration of the tool on a CubeSat OBC would allow confirmation of the method with real empirical data supporting its effectiveness. Such a mission would offer an invaluable prototyping of the tool for larger scale missions in the future. Conversion of the algorithm to OBC compatible software is relatively straightforward. An addition of a coordinate converter to transform GPS coordinates, Longitude, Latitude, and Altitude (LLA), to non-rotating ECI coordinates is required. A simple solution to this is by use of C code freely provided by the Standards of Fundamental Astronomy (SOFA). SOFA provides a library of tools for astronomical and Earth attitude calculations [60]. Prior to the mission change, use of SOFA for coordinate transformation was examined, using a C function written based on the procedure presented by Petit and Luzum [61]. The function is also available in the SeaLion Git repository along with the DEVS code [39]. With this groundwork laid, the presented DEVS routine may be implemented on a future CubeSat mission.

REFERENCES

[1] D. Werner, "How low can satellites go? VLEO entrepreneurs plan to find out," *SpaceNews*. [Online]. Available: https://spacenews.com/how-low-can-satellites-go-vleo-entrepreneurs-plan-to-find-out/

[2] P. C. E. Roberts, "1st Symposium of Very Low Earth Orbit Missions and Technologies," *CEAS Space Journal,* vol. 14, no. 4, pp. 605-608, 2022/10/01 2022, doi: 10.1007/s12567-022-00466-9.

[3] SpaceTrack. "Help Documentation: Frequently Asked Questions (FAQ) What is Alpha-5?" https://www.space-track.org/documentation#/faq (accessed July 20, 2023).

[4] K. Riesing, "Orbit Determination from Two Line Element Sets of ISS-Deployed CubeSats," presented at the AIAA/USU Conference on Small Satellites, Utah State University, 2015. [Online]. Available: https://digitalcommons.usu.edu/smallsat/2015/all2015/58/.

[5] M. A. Swartwout, "A brief history of rideshares (and attack of the CubeSats)," in *2011 Aerospace Conference*, 5-12 March 2011, pp. 1-15, doi: 10.1109/AERO.2011.5747233.

[6] "SpaceX Kicks Off 2023 With Transporter 6 Rideshare Mission," (in English), *Satellite Today,* 4 Jan 2023. [Online]. Available: http://proxy.lib.odu.edu/login?url=https://www.proquest.com/trade-journals/spacex-kicks-off-2023-with-transporter-6/docview/2760644444/se-2.

[7] R. C. Jamie Chin, et al., "CubeSat 101: Basic Concepts and Processes for First-Time CubeSat Developers," NASA CubeSat Launch Initiative, 2017. [Online]. Available: https://www.nasa.gov/sites/default/files/atoms/files/nasa_csli_cubesat_101_508.pdf

[8] *CubeSat Design Specification Rev 14.1*, C. P. S. The Cubesat Program, San Luis Obispo, CA, 2022.

[9] "CubeSats," The European Space Agency (ESA), 2022. [Online]. Available: https://www.esa.int/Enabling_Support/Preparing_for_the_Future/Discovery_and_Preparation/CubeSats#:~:text=These%20little%20satellites%20have%20a,studies%2C%20and%20even%20commercial%20purposes.

[10] Old Dominion University & United States Coast Guard Academy, "Critical Design Review: Mission SeaLion - ODU/CGA 3U CubeSat," 2022.

[11] C. A. Branco, E. Senecal, and S. Asundi, "SGP4 versus Vinti6: A Comparative Study of Orbit Propagators for Very Low Altitude CubeSat Orbits," presented at the AIAA SCITECH Forum, 2023. [Online]. Available: https://doi.org/10.2514/6.2023-0934.

[12] L. M. Giovanni Minelli, Noah Weitz and R. P. David Rigmaiden, James Horning, James Newman, "The Mobile CubeSat Command and Control (MC3) Ground Station Network," Naval Postgraduate School, 2020. [Online]. Available: https://www.nasa.gov/sites/default/files/atoms/files/2020-05-20_mobile_cubesat_command_and_control_mc3_-_nasa_smallsat_virtual_institute_webinar.pdf

[13] Z. Leffke *et al.*, "A Prototype Virginia Ground Station Network," presented at the Small Satellite Conference, Utah State University, Logan, UT, 2020. [Online]. Available: https://digitalcommons.usu.edu/smallsat/2020/all2020/159/.

[14] AGI, "Ansys STK: Software for Digital Mission Engineering and Systems Analysis," *Ansys Government Initiatives Products,* 2023. [Online]. Available: https://www.ansys.com/products/missions/ansys-stk.

[15] K. Y.-T. Chiu, "SeaLion CubeSat Mission Architecture Using Model Based Systems Engineering with a Docs as Code Approach," Master of Science (MS), Mechanical & Aerospace Engineering, Old Dominion University, Old Dominion University, 2023. [Online]. Available: https://digitalcommons.odu.edu/mae_etds/360

[16] J. D. Bhagatji, O. Kravchenko, and S. Asundi, "Large Deformation Bending of Ultralight Deployable Structure For Nano-Micro-Class Satellites," in *AIAA SCITECH 2023 Forum*, (AIAA SciTech Forum: American Institute of Aeronautics and Astronautics, 2023.

[17] S. Marquez, S. Asundi, and K. Chiu, "Model-Based CubeSat Flight-Software Architecture using a Docs-as-Code approach," in *AIAA SCITECH 2023 Forum*, (AIAA SciTech Forum: American Institute of Aeronautics and Astronautics, 2023.

[18] K. Chiu, S. Marquez, and S. Asundi, "Model Based Systems Engineering with a Docs-as-Code Approach for the SeaLion CubeSat Project," *Systems,* vol. 11, no. 7, p. 320, 2023. [Online]. Available: https://www.mdpi.com/2079-8954/11/7/320.

[19] R. C. Borowicz, "Failure Mode, Effects and Criticality Analysis of a Very Low Earth Orbit CubeSat Mission," Master of Science (MS), Mechanical & Aerospace Engineering, Old Dominion University, Old Dominion University, 2023. [Online]. Available: https://digitalcommons.odu.edu/mae_etds/348

[20] J. D. Siciliano, "Design and Analysis of Electrical Power and Communication Systems for 3U SeaLion CubeSat Mission," M.S., Old Dominion University, United States -- Virginia, 29206022, 2022. [Online]. Available: https://digitalcommons.odu.edu/ece_etds/238/

[21] W. J. Larson and J. R. Wertz, *Space Mission Analysis and Design* (no. DOE/NE/32145-T1; ISBN: 1-881883-01-9; 0-7923-2998-2). United States: Torrance, CA (United States); Microcosm, Inc. (in English), 1992, p. Medium: X; Size: Pages: (871 p).

[22] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems, Second Edition (Chapman & Hall/CRC Applied Mathematics & Nonlinear Science)*. Chapman & Hall/CRC, 2011.

[23] S. P. Wright, "Orbit Determination Using Vinti's Solution," PhD Dissertation, Department of Aeronautics and Astronautics, Air Force Institute of Technology (AFIT), 2016. [Online]. Available: https://scholar.afit.edu/etd/276

[24] J. R. Taylor, *Classical Mechanics*. University Science Books, 2005.

[25] G. J. Der. DerAstrodynamics. (2013). Classical and Advanced Kepler Algorithms. Available: https://www.academia.edu/44213056/Classical_and_Advanced_Kepler_Algorithms

[26] J. P. Vinti, G. J. Der, and N. L. Bonavito, *Orbital and Celestial Mechanics*, P. Zarchan, ed.: AIAA Progress in Astronautics and Aeronautics, 1998.

[27] G. Der and R. Danchick, "Analytic and numerical error covariance matrix propagation," in *Astrodynamics Conference*, (Guidance, Navigation, and Control and Co-located Conferences: American Institute of Aeronautics and Astronautics, 1996.

[28] L. George, *Introduction to Orbital Mechanics*. Pressbooks.

[29] AGI, "How Does the High-Precision Orbit Propagator (HPOP) Work?," *STK Knowledge Base*. [Online]. Available: https://analyticalgraphics.my.site.com/faqs/articles/Knowledge/How-does-the-High-Precision-Orbit-Propagator-HPOP-work.

[30] AGI, "High-Precision Orbit Propagator (HPOP)," *STK Help,* July 2023. [Online]. Available: https://help.agi.com/stk/#hpop/hpop.htm.

[31] AGI, "Technical Notes for HPOP," *STK Help,* July 2023. [Online]. Available: https://help.agi.com/stk/index.htm#hpop/hpopTechNotes.htm.

[32] Z. Y.-C. Liu, S. Tarlow, M. Akbar, Q. Donnellan, and D. Senkow, "Improved Orbital Propagator Integrated with SGP4 and Machine Learning," presented at the 35th Annual Small Satellite Conference, Utah State University, 2021, SSC21-IX-02. [Online]. Available: https://digitalcommons.usu.edu/smallsat/2021/all2021/193/.

[33] G. Der and A. E. Kalman, "Autonomous Orbit Propagation for GPS Equipped Cubesats," presented at the Advanced Maui Optical and Space Surveillance (AMOS) Technologies Conference, January 01, 2017, 2017. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2017amos.confE..79D.

[34] E. W. Weisstein, "Oblate Spheroidal Coordinates," *From MathWorld--A Wolfram Web Resource*. [Online]. Available: https://mathworld.wolfram.com/OblateSpheroidalCoordinates.html

[35] J. P. Vinti, G. J. Der, and N. L. Bonavito, "Supplemental Material," *vol. 177*, *Orbital and Celestial Mechanics*, P. Zarchan, Ed.: AIAA Progress in Astronautics and Aeronautics, 1998. [Online]. Available: https://arc.aiaa.org/doi/suppl/10.2514/4.866487

[36] J. Abedrabbo, S. Marquez, and E. Senecal, "VintiCode producing NaNs with at least one output," *GitHub Repository,* vol. ODU-CGA-CubeSat/orbit-propagator, 2022. [Online]. Available: https://github.com/ODU-CGA-CubeSat/orbit-propagator/issues/5.

[37] B. Weber. "Orbital Mechanics & Astrodynamics." https://orbital-mechanics.space/intro.html (accessed May 24, 2023).

[38] *Complete 1976 Standard Atmosphere*. (2023). MATLAB Central File Exchange. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/13635-complete-1976-standard-atmosphere

[39] E. Senecal and S. Marquez, "orbit-propagator," *GitHub Repository,* vol. ODU-CGA-CubeSat, 2023. [Online]. Available: https://github.com/ODU-CGA-CubeSat/orbit-propagator.

[40] T. Benson, "Glossary," *NASA Glen Research Center,* 2021. [Online]. Available: https://www.grc.nasa.gov/www/k-12/TRC/laefs/laefs_k.html.

[41] L. Snyder, "Diagram illustrating and explaining various terms in relation to Orbits of Celestial bodies," ed, 2007.

[42] D. T. S. Kelso, "NORAD Two-Line Element Set Format," CelesTrak, 2022. [Online]. Available: https://celestrak.org/NORAD/documentation/tle-fmt.php

[43] F. R. Hoots, P. W. Schumacher, and R. A. Glover, "History of Analytical Orbit Modeling in the U. S. Space Surveillance System," *Journal of Guidance, Control, and Dynamics,* vol. 27, no. 2, pp. 174-185, 2004/03/01 2004, doi: 10.2514/1.9161.

[44] B. N. Daniel J Fonte, Chris Sabol, D. A. Danielson, Major W. R. Dyar, "Comparison of Orbit Propagators in the Research and Development Goddard Trajectory Determination System (R & D

GTDS). Part I: Simulated Data," *Naval Postgraduate School,* 20 August 1995. [Online]. Available: http://faculty.nps.edu/dad/orbital/halif.pdf.

[45] N. L. BONAVITO, J. S. WATSON, and H. WALDEN, "AN ACCURACY AND SPEED COMPARISON OF THE VINTI AND BROUWER ORBIT PREDICTION METHODS " *NASA GODDARD SPACE FLIGHT CENTER,* 1966. [Online]. Available: https://ntrs.nasa.gov/api/citations/19670028911/downloads/19670028911.pdf.

[46] J. P. Vinti, "Inclusion of the Third Zonal Harmonic in an Accurate Reference Orbit of an Artificial Satellite," *JOURNAL O F RESEARCH of the Notional Bureau of Standards - B. Mathematics and Mathematical Physics,* vol. 70B, no. 1, 1966. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/jres/70B/jresv70Bn1p17_A1b.pdf.

[47] AGI, "Satellite: Available Data Providers," *STK Help,* 2023. [Online]. Available: https://help.agi.com/stk/#../Subsystems/dataProviders/Content/html/Satellite.htm.

[48] S. F. Rafano Carná and R. Bevilacqua, "High fidelity model for the atmospheric re-entry of CubeSats equipped with the Drag De-Orbit Device," *Acta Astronautica,* vol. 156, pp. 134-156, 2019/03/01/ 2019, doi: https://doi.org/10.1016/j.actaastro.2018.05.049.

[49] M. M. Moe, S. D. Wallace, and K. Moe, "Refinements in determining satellite drag coefficients - Method for resolving density discrepancies," *Journal of Guidance, Control, and Dynamics,* vol. 16, no. 3, pp. 441-445, 1993, doi: 10.2514/3.21029.

[50] D. Vallado, P. Crawford, R. Hujsak, and T. S. Kelso, "Revisiting Spacetrack Report #3," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, (Guidance, Navigation, and Control and Co-located Conferences: American Institute of Aeronautics and Astronautics, 2006.

[51] I. SkyTraq Technology, "Venus838FLPx-L / Venus838FLPx-D ", ed.

[52] "GPS Position Accuracy Measures " NovAtel, 2023. [Online]. Available: https://www.gnss.ca/app_notes/APN-029_GPS_Position_Accuracy_Measures_Application_Note.html

[53] D. W. Webb, "Circular Probable Error for Circular and Noncircular Gaussian Impacts," Army Research Laboratory, 2012. [Online]. Available: https://apps.dtic.mil/sti/pdfs/AD1043284.pdf

[54] M. S. Ltd. *GPS TTFF and startup modes* [Online]. Available: https://www.measurementsystems.co.uk/docs/TTFFstartup.pdf.

[55] *DataSheet: S-BAND TRANSMITTER*. (2021). [Online]. Available: https://www.endurosat.com/cubesat-store/cubesat-communication-modules/s-band-transmitter/#request-step-modal

[56] M. A. El Moukalafe and K. Minaoui, "Communication Optimization Approach for S-Band LEO CubeSat Link Budget," in *WITS 2020*, Singapore, S. Bennani, Y. Lakhrissi, G. Khaissidi, A. Mansouri, and Y. Khamlichi, Eds., 2022// 2022: Springer Singapore, pp. 1001-1011.

[57] S. M. Alcaide and S. M. Alcaide, "Mobile CubeSat Command and Control (MC3) 3-meter dish calibration and capabilities," Thesis, Monterey, California: Naval Postgraduate School, 2014. [Online]. Available: https://hdl.handle.net/10945/42572

[58] A. Becker, *Kalman Filter from the Ground Up*, 1st ed. Alex Becker, 2023.

[59] S. Asundi, N. Fitz-Coy, and H. Latchman, "Evaluation of Murrell's EKF-Based Attitude Estimation Algorithm for Exploiting Multiple Attitude Sensor Configurations," (in eng), *Sensors (Basel),* vol. 21, no. 19, Sep 27 2021, doi: 10.3390/s21196450.

[60] "SOFA Tools for Earth Attitude," *International Astronomical Union, Standards Of Fundamental Astronomy,* vol. Software version 18. Document revision 1.64. Version for C programming language, 2021. [Online]. Available: http://www.iausofa.org/.

[61] G. Petit and B. Luzum, "IERS conventions (2010)," *Tech. Rep. DTIC Document,* vol. 36, p. 180, 01/01 2010.

APPENDICES

A. Code for Orbit Propagator

```
%% On Board Orbit Propagator Simulation

clear
%% Inputs %%
GPSFileName = "HPOP_J2000_State_Vector_1s.csv";
## GPSFileName = "HPOP_1976_J4_State_Vector_1s.csv";
t_end = 12*90*60; % s
c_d = 2.2;
S_Ref = 0.031; % m^s
SatMass = 5.5; % kg
GPS_period = 90*60; % s
termination_alt = 65; % km
%% End Inputs %%

GPS = importdata (GPSFileName,",",1); % Load GPS [Position, Velocity] data (ECI)
GPS.data(:,1) = GPS.data(:,1)*3600;
format long g
load('atmosDensity.mat')
DensityAltIncr = AtmosDensity(2,1)-AtmosDensity(1,1); %km
r_MSL = 6.371*10^3; % km

altitude = nan(1,1);
Veloc = nan(1,3);
velocUnitVector = nan(1,3);
alt_cond = 0;

i = 0;
## t_start = time();
## t1 = time() - t_start;
t1 = 0; dt = 30; % For simulation only, time delay in system in seconds

cd build

while (t1 < t_end)

## if (alt_cond == 1)
## break;
## endif

## round(t1-t_start)+1
% Ping GPS %
z_gps0 = GPS.data(round(t1)+1,:);
## pause(2); % Simlulates GPS pinging delay
z_gps1 = GPS.data(round(t1+2)+1,:);
% Average Result
GPS_avg = mean([z_gps0;z_gps1],1);
t0 = GPS_avg(1,1); X0 = GPS_avg(1,2:7);
```

```
t_last_gps = t0;

i = i+1;
Eph(i,:) = GPS_avg;

## t1 = time() - t_start;
t1 = t0+dt;
## t1-t_last_gps

altitude = ( norm(X0(1:3)) - r_MSL );
rho_0 = AtmosDensity(round((altitude-AtmosDensity(1,1))/DensityAltIncr+1),2); %
kg/m^3
rho_1 = rho_0;
Veloc(1,:) = X0(4:6)*1000; V0 = norm(Veloc(1,:)); %m/s
velocUnitVector(1,:) = Veloc(1,:)./V0;
X0_eff = X0(1:6);

while ( (t1-t_last_gps) < GPS_period)
% Propagate between GPS Pings
## t_propStart = time();

## disp('propagate')

## if (altitude < termination_alt)
## disp('altitude condition')
## alt_cond = 1;
## break;
## cd ..
## endif

V0_effective = 0.8*V0 + 0.2*( 1/V0 + ( (c_d*S_Ref/(2*SatMass)) * (rho_0 +
        (rho_1-rho_0)/2)*(t1-t0) ) ) ^ (-1);
X0_eff(4:6) = V0_effective*velocUnitVector(1,:)/1000;
csvwrite("inputStateVect.txt",transpose([ X0_eff, (t1-t0) ]))

% Call C code Vinti6 Executable
system('./orbit-propagator');

%Get Data from Output of Vinti6 program
VintiOutput = csvread("outputStateVect.txt");
% Compute New State, X1 at t1, calling it the new X0 and t0. Store data
X1(1,:) = VintiOutput(1:6);
## t_compute(i) = time()-t_propStart;

altitude = ( norm(X1(1:3)) - r_MSL );
if altitude > max(AtmosDensity (:,1))
rho_1 = 0;
else
rho_1 =
        AtmosDensity(round((altitude-AtmosDensity(1,1))/DensityAltIncr+1),2); % kg/m^3
endif

i = i+1;
```

```
98 Eph(i,:) = [t1, X1];
99
100 fprintf("\t time elapsed (orbits): %f\n",t1/(1.5*3600))
101 ## t1 = time() - t_start;
102 t1 = t1+dt;
103 if (t1 > t_end) break; endif
104 endwhile
105 endwhile
106
107 cd ..
108 outputFileName =
          ['VintiEphemeris_cd',num2str(c_d),'_S_ref',num2str(S_Ref),'_GPS_period',num2str(GPS_
period),'.csv'];
109 csvwrite(outputFileName,Eph)
110 ## computeTime = mean(t_compute)
```

B. Code for Orbit Propagator with State Estimator

```
%% On Board Orbit Propagator Simulation

clear
%% Inputs %%
GPSFileName = "HPOP_J2000_State_Vector_1s.csv";
## GPSFileName = "HPOP_1976_J4_State_Vector_1s.csv";
t_end = 12*90*60; % s
c_d = 2.2;
S_Ref = 0.031; % m^s
SatMass = 5.5; % kg
GPS_period = 90*60; % s
termination_alt = 65; % km
## sigPos = (2.5/1000) / sqrt(2*log(2));
## sigVel = (0.1/1000) / sqrt(2*log(2));
%% End Inputs %%

## sigm_gps = diag([sigPos^2 sigPos^2 sigPos^2 sigVel^2 sigVel^2 sigVel^2]); %
        Error-covariance of GPS
Bstar = (c_d*S_Ref/(2*SatMass));

GPS = importdata (GPSFileName,",",1); % Load GPS [Position, Velocity] data (ECI)
GPS.data(:,1) = GPS.data(:,1)*3600;
format long g
load('atmosDensity.mat')
DensityAltIncr = AtmosDensity(2,1)-AtmosDensity(1,1); %km
r_MSL = 6.371*10^3; % km

altitude = nan(1,1);
Veloc = nan(1,3);
velocUnitVector = nan(1,3);
alt_cond = 0;

i = 0;
## t_start = time();
## t1 = time() - t_start;
t1 = 0; dt = 30; % For simulation only, time delay in system in seconds

cd build

while (t1 < t_end)

## if (alt_cond == 1)
## break;
## endif

## round(t1-t_start)+1

%%%%%%%%%%%%% Ping GPS %%%%%%%%%%%%%%%
z_gps0 = GPS.data(round(t1)+1,:);
t0 = z_gps0(1,1); X0 = z_gps0(1,2:7);
```

```
50 ## pause(2); % Simlulates GPS pinging delay
51 z_gps1 = GPS.data(round(t1+2)+1,:);
52 t1 = z_gps1(1,1); X1 = z_gps1(1,2:7);
53 % Propagate from ping 1 to ping 2 and Compute Error State
54 [~, x1_pred(1:6)] = StateTransMatrix(transpose(X0), ( t1-t0 ) );
55 DeltX0 = transpose(x1_pred - X1);
56
57 % Use last GPS ping as start
58 t0 = t1; X0 = X1; X0_eff = X0(1:6);
59 ## P0 = sigm_gps;
60 ## % Average Result
61 ## GPS_avg = mean([z_gps0;z_gps1],1);
62 ## t0 = GPS_avg(1,1); X0 = GPS_avg(1,2:7);
63 t_last_gps = t0;
64
65 i = i+1;
66 Eph(i,:) = z_gps1;
67
68 ## t1 = time() - t_start;
69 t1 = t0+dt;
70 ## t1-t_last_gps
71 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
72
73 altitude = ( norm(X0(1:3)) - r_MSL );
74 rho_0 = AtmosDensity(round((altitude-AtmosDensity(1,1))/DensityAltIncr+1),2); %
        kg/m^3
75 rho_1 = rho_0;
76 Veloc(1,:) = X0(4:6)*1000; V0 = norm(Veloc(1,:)); %m/s
77 velocUnitVector(1,:) = Veloc(1,:)./V0;
78
79 while ( (t1-t_last_gps) < GPS_period)
80 % Propagate between GPS Pings
81 ## t_propStart = time();
82 ## altitude = ( norm(X0(1:3)) - r_MSL );
83 ## if altitude > max(AtmosDensity (:,1))
84 ## rho_0 = 0;
85 ## else
86 ## rho_0 =
        AtmosDensity(round((altitude-AtmosDensity(1,1))/DensityAltIncr+1),2); % kg/m^3
87 ## endif
88 ## disp('propagate')
89
90 V0_effective = 0.8*V0 + 0.2*( 1/V0 + ( Bstar * (rho_0 +
        (rho_1-rho_0)/2)*(t1-t0) ) ) ^ (-1);
91 X0_eff(4:6) = V0_effective*velocUnitVector(1,:)/1000;
92
93 % Call C code Vinti6 Executable
94 % Compute New State, X1 at t1, calling it the new X0 and t0. Store data
95 [PHI, X1] = StateTransMatrix(transpose(X0_eff), (t1-t0));
96 DeltX = PHI*DeltX0;% * (1+beta*(t1-t0));
97 ## P1 = PHI*P0/PHI; K1 = P1/(P1 + sigm_gps);
98 X1 = transpose(X1 - DeltX);
99
```

```
100 altitude = ( norm(X1(1:3)) - r_MSL );
101 if altitude > max(AtmosDensity (:,1))
102 rho_1 = 0;
103 else
104 rho_1 =
        AtmosDensity(round((altitude-AtmosDensity(1,1))/DensityAltIncr+1),2); % kg/m^3
105 endif
106 ## t_compute(i) = time()-t_propStart;
107
108 i = i+1;
109 Eph(i,:) = [t1, X1];
110
111 fprintf("\ttime elapsed (orbits): %f\n",t1/(60*90))
112 ## t1 = time() - t_start;
113 t1 = t1+dt;
114 if (t1 > t_end) break; endif
115 endwhile
116 endwhile
117
118 cd ..
119 outputFileName = ['ErrorCorrectedEphemeris_GPS_period',num2str(GPS_period),'.csv'];
120 csvwrite(outputFileName,Eph)
121 ## computeTime = mean(t_compute)
```

```
1 %%%%% State Transition Function %%%%%
2 function [PHI, xn1] = StateTransMatrix(x0, dt)
3 % Update Input vector or Vinti6 Program
4 csvwrite("inputStateVect.txt",[x0; dt])
5 % Call C code Vinti6 Executable
6 system('./orbit-propagator');
7 %Get Data from Output of Vinti6 program
8 xn1 = csvread("outputStateVect.txt");
9
10 PHI = zeros(6,6);
11 h_pos = norm(x0(1:3)) * 10^(-3);
12 h_vel = norm(x0(4:6)) * 10^(-3);
13 for i = 1:3
14 dxi0 = zeros(6,1); dxi0(i) = h_pos;
15 xi0 = x0 + dxi0;
16 csvwrite("inputStateVect.txt",[xi0; dt])
17 % Call C code Vinti6 Executable
18 system('./orbit-propagator');
19 %Get Data from Output of Vinti6 program
20 xi1 = csvread("outputStateVect.txt");
21 PHI(:,i) = (xi1 - xn1)/h_pos;
22
23 dxi0 = zeros(6,1); dxi0(i+3) = h_vel;
24 xi0 = x0 + dxi0;
25 csvwrite("inputStateVect.txt",[xi0; dt])
26 % Call C code Vinti6 Executable
27 system('./orbit-propagator');
28 %Get Data from Output of Vinti6 program
```

```
29 xi1 = csvread("outputStateVect.txt");
30 PHI(:,i+3) = (xi1 - xn1)/h_vel;
31 endfor
32 endfunction
```

VITA

Ethan Senecal was born in Washington state. As the son of a Coast Guard engineer, he grew up moving frequently as tours of duty commanded. Along with his parents and two sisters, he has lived in Connecticut, Virginia, and New Jersey. Having been fascinated with aviation and physics from a young age, mechanical and aerospace engineering was a natural choice of study. In 2022, he received a Bachelor of Science in Mechanical Engineering from Old Dominion University, Department of Mechanical and Aerospace Engineering (238 Kaufman Hall, Norfolk, VA 23529). During his time there, he has worked as a tutor for mathematics and engineering courses and as a Graduate Lab Assistant in the Additive Manufacturing Lab, supporting projects for the U.S. Navy and academic research by 3D printing critical parts. He interned as a mechanical engineer at General Dynamics Electric Boat, supporting production and delivery of Columbia Class Subsystems. He will go on to work at Jefferson Lab in September 2023, after graduating with his Master of Science in Aerospace Engineering. Responsibilities there will involve designing mechanical and cryogenic systems for the electron-ion collider (EIC). He is excited to be part of a team on the forefront of scientific discovery and cutting-edge R&D. He hopes to build a career in which he will collaborate on projects with lasting significance that will develop innovative solutions to modern environmental and energy crises.