UNCERTAINTY CHARACTERIZATION OF ORBITAL DEBRIS

by

JOLANTA MATUSEWICZ

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN AEROSPACE ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2006

ACKNOWLEDGEMENTS

First I would like to thank my research advisor, Dr. Kamesh Subbarao for his guidance in my research. He has helped me develop my interest in orbital mechanics by expanding my knowledge of the field. I would also like to thank him for his effort in instilling strong research and study ethics. He has maintained a professional work level for the Aerospace Systems Lab (ASL) research group. Each of his students is expected to give presentations at weekly meetings and submit weekly reports that help improve communication and organizational skills. I have confidence that these will benefit me in my pursuit of a Ph.D in Aerospace Engineering.

I would also like to thank my Graduate Advisor, Dr. Don Wilson for his guidance in my pursuit of the Master's of Science degree in Aerospace Engineering. He has always taken the time to meet with me and to answer my questions that have helped me finish the program in a timely fashion. Dr. Wilson has also guided me in financial support such as teaching assistantships and fellowships that have greatly helped pay for my studies. I would also like to thank Dr. Wilson for asking me to help run the elementary school rocket building competition. In addition to being an enjoyable task, it has helped me improve my communication and teaching skills. I would also like to thank Dr. Atilla Dogan for his guidance as a professor in the flight mechanics course and for agreeing to serve on my thesis defense committee. Dr. Dogan has helped me improve my understanding of spacecraft dynamics and control that has helped me with other classes such as applied nonlinear controls. I would also like to thank the United Space Alliance (USA) for funding my research that has lead to my thesis. Specifically, I would like to thank Joe Frisbee from USA for his helpful suggestions on my research project. I would also like to thank Barbara Sanderson, Janet Gober, Sally Thompson, Donna Woodhead, and Connie Neice for helpfully answering my questions, aiding me with last minute photocopies, faxes, etc… They have helped me while I worked as a teaching assistant and research assistant.

Lastly, I would like to thank my family and friends who have supported me in my studies. I would like to especially thank my parents, Teresa and Jan Matusewicz, for continuing to help fund my studies as I pursue my next level of education in my field of interest. I would also like to thank them for their interest in my research and for always giving me helpful pointers.

July 19, 2006

ABSTRACT

UNCERTAINTY CHARACTERIZATION OF ORBITAL DEBRIS

Publication No. ______

Jolanta Matusewicz, MS

The University of Texas at Arlington, 2006

Supervising Professor: Dr. Kamesh Subbarao

The increase of orbital debris in the low Earth orbit is a concern for the space industry. Spacecraft and satellites are at risk of collision with orbital debris. Collisions can damage sensitive components or result in catastrophic failure. Organizations such as the United States Space Command are responsible for tracking orbital debris using ground based sensor sites located around the world. Orbit determination techniques are used to estimate the position and velocity of the orbit using range, azimuth, and elevation measurements obtained from the sensors.

Nine sensors from the Space Surveillance Network are simulated to track a debris object in an International Space Station orbit. Perturbations due to a 4 X 4 complex gravity model and an exponential atmospheric model are included in the two-body orbital equations of motion force model. The nonlinear batch least squares and the continuous-discrete extended Kalman filter techniques are used to estimate the debris orbit.

TABLE OF CONTENTS

Chapter

Appendix

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

Table Page

LIST OF SYMBOLS

Symbol	Definition	Units
R	Earth's Radius Magnitude	km
r	Debris Object Geocentric Distance	km
m	Debris Object Mass	kg
C_d	Drag Coefficient	
A	Debris Object Area	m^2
az	Azimuth Angle	deg
el	Elevation Angle	deg
h_{alt}	Altitude	km
h_o	Base Altitude	km
H_{alt}	Scale Height	km
$\vec{r}$	Debris Object Radius Vector	km
S_n^m, C_n^m	Harmonic Coefficients of the Geopotential	
P_n^m	Associated Legendre Polynomial	
P	State Error Covariance Matrix	
$\vec{R}$	Sensor Site Radius Vector	km
$\vec{a}$	Debris Object Acceleration Vector	km/s^2
a	Semi-Major Axis	km
F	State Matrix	
H	Sensitivity Matrix	
K	Kalman Gain Matrix	
R_{cov}	Measurement Error Covariance Matrix	
Q	Process Noise Error Covariance Matrix	
μ	Earth's Gravitational Parameter	km^3/s^2
$\vec{h}_s$	Specific Angular Momentum Vector	km^2/s
h_s	Specific Angular Momentum Magnitude	km^2/s

i	Inclination of Orbit	deg
ρ_{atm}	Atmospheric density	kg/m^3
$\vec{\rho}$	Slant Range Vector	km
ρ	Range	km
ρ_o	Reference Density	kg/m^3
ϕ	Latitude of Sensor Site	deg
λ	East Longitude of Sensor Site	deg
γ	Geocentric Longitude of Debris Object	deg
δ	Geocentric Latitude of Debris Object	deg
α	Right Ascension of the Debris Object	deg
θ_g	Right Ascension of Greenwich	deg
θ	Local Sidereal Time of Sensor Site	deg
Φ	State Transition Matrix	
$\vec{v}$	Debris Object Velocity Vector	km/s
$\vec{v}_r$	Velocity Relative to the Atmosphere	km/s
$\vec{\varpi}$	Inertial Rotation Vector of Earth	rad/s
ω_e	Angular Speed of Earth	rad/s
U	Geopotential Function	

Subscripts	Definition
n	Degree
(u,e,n,s)	(Up, East, North, South)
d	Drag
g	Gravity
cov	Covariance
atm	Atmosphere
alt	Altitude

Superscripts	Definition
m	Order

Acronyms	Definition
UT	Universal Time
USSPACECOM	United States Space Command
USSTRATCOM	United States Strategic Command
SSN	Space Surveillance Network
LEO	Low Earth Orbit
ESA	European Space Agency
ISS	International Space Station
EGM96	Earth Gravity Model 1996
NIMA	National Imagery and Mapping Agency
AFS	Air Force Station
AFB	Air Force Base
RAF	Royal Air Force
ECI	Earth-Centered Inertial
LST	Local Sidereal Time
GMT	Greenwich Mean Time
GMST	Greenwich Mean Sidereal Time
JD	Julian Date
GEODSS	Ground-based Electro-Optical Deep Space Surveillance
BMEWS	Ballistic Missile Early Warning Radar System
PARCS	Perimeter Acquisition Radar Attack Characterization System
PAWS	Phased Array Warning System
NLS	Nonlinear Least Squares
EKF	Extended Kalman Filter

CHAPTER 1

INTRODUCTION

1.1 Research Objectives

The first research objective is to characterize the size and statistics of the uncertainty associated with an orbit debris object in an International Space Station (ISS) orbit subject to a complex nonlinear gravity model and atmosphere. The second research objective is to compare the performance of different orbit determination techniques. For this research, the nonlinear least squares differential corrections (NLS) and continuous-discrete extended Kalman filter (EKF) techniques are compared. The purpose is to improve the orbit determination accuracy based on the information available from the first part of the research.

1.2 Orbital Debris

Orbital debris are artificial objects in orbit around Earth that have no useful purpose. Debris objects can be launch vehicle upper stages, mission-related objects (i.e. bolts), disabled satellites, explosion or collision components, paint flakes, solid rocket motor slag, nuclear power source coolant (i.e. from RORSAT), and other particles.[1] Radar Ocean Reconnaissance Satellite (RORSAT) is the western term for the Soviet Upravlenniye Sputnik-Aktivny (US-A) satellites. RORSAT satellites were launched between 1967 and 1988 to monitor fleets.[2] Spacecraft and satellites avoid collisions by maneuvering around large debris objects. Large debris objects are greater than 10 cm in diameter and can be tracked and catalogued by United States Space Command (USSPACECOM) sensors. The USSPACECOM analyzes the trajectories of orbital debris to identify possible collisions. For example, if a large debris object is estimated to come within a few kilometers of the space shuttle then a change in orbit maneuver is implemented. Particles less than 1 mm in diameter are not tracked by sensors. Such small particles cause surface abrasions and microscopic holes, but do not cause catastrophic failure. Medium size particles (objects with a diameter between 1 mm to 10 cm) are not easily tracked and can cause significant damage.[3] Large debris objects are the focus of this research.

1.3 Motivation

The orbital debris environment is increasing as more satellites are being launched. High-speed (10 km/s or 22,000 mi/h) collisions between existing debris objects can also increase the amount of debris in orbit. Debris objects remain in orbit for an extended period of time, which is a major concern for the space industry. Disabled satellite explosions and fragmentation can damage functioning satellites and spacecraft.[3] The ISS has an expected lifetime of 30 years, which makes the ISS vulnerable to the orbital debris environment.[4] The ISS is heavily shielded to help protect critical systems from orbit debris collisions. Components such as habitable compartments and high-pressure tanks are effectively shielded against debris as large as 1 cm in diameter.[3] Besides having appropriate shielding, the ISS and other spacecraft can change orbit to avoid impact with debris. When spacecraft or satellite alter their orbits, fuel is expended. In addition, experiments on the ISS need to be shutdown before the orbit can be changed.[4] Both actions waste time and money; hence, it is imperative to reduce unnecessary maneuvers. As such, it is important to validate the assumptions that are currently used for orbital debris collisions.

Figure 1.1 is a graphic showing the debris distribution in a Low Earth Orbit (LEO).[5]

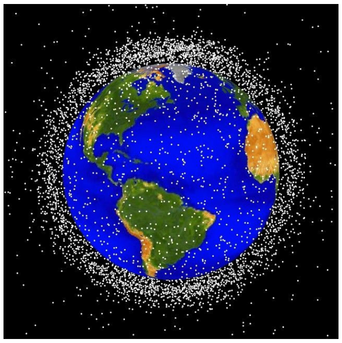


Figure 1.1: Space Debris in Low Earth Orbit

There are approximately 6,000 objects in LEO being tracked by the U.S. Space Surveillance Network (SSN). Of these only about 5 percent are operating spacecraft.[6] There are approximately 11,000 objects in LEO 10 cm or larger and approximately 100,000 fragments between 1 and 10 cm in diameter. There are possibly more than tens of millions of particles smaller than 1 cm in diameter.[7]

1.4 Orbit Determination Research

A major topic of interest concerning the orbit determination of debris objects is analysis of orbital debris collision probability. Maneuvering the ISS, space shuttle or satellites is based on a probabilistic methodology.[4] For example, an imaginary sphere with a specified radius is created around the ISS (ISS at center). The probability that a debris object will pass within the sphere is evaluated. If this probability is above a chosen threshold, the ISS will then be maneuvered to avoid a collision with the debris.[4] This requires accuracy in calculating the probability of collision. The probability that a debris object will be in the sphere (or a given region of space) is determined by the state error covariance. This covariance depicts the accuracy of the state (position and velocity) of the debris object at a certain point in time.[4] Specifically, the diagonal elements of the error covariance matrix provide a measure of the accuracy of the state.[8] The position and velocity of the debris object can be predicted for a future time by propagating the covariance and state.[4] If the covariance is accurate then the predicated collision is more accurate. If the covariance is too small then there is a small chance that the debris object will be in the sphere and an avoidance maneuver will not be performed. Furthermore, if the covariance is too large, then the debris object can be located anywhere and probability analysis is useless.[4]

The state error covariance of a debris object can be obtained from ground sensor observations such as range, azimuth, and elevation. Batch least squares and the Kalman filter are common techniques that use these sensor measurement to obtain the covariance and state estimate of the debris object. The batch least squares calculates the error between the observations and the estimates obtained using the force model. A weighted sum of the squared errors is evaluated to update the initial guess for the state of the debris object. This process is repeated until a chosen convergence is achieved. The convergence is determined by evaluating the difference between the updated state estimate and the previous state estimate. The Kalman filter is a smoothing technique that tries to remove noise in the observations. The Kalman filter updates the current state continuously using each measurement sequentially.[4] More specifically, the extended Kalman filter uses a weighted statistical average of the difference in state inputs predicted from the force model and the observations. The model is corrected towards more accurate estimates based on the known errors in the state inputs.[9] The extended Kalman filter weights are chosen to minimize the mean squared error of the estimate.[10]

The Kalman filter can account for force model errors that the batch least squares method ignores.[4] Force model errors are a result of improper force models. Errors can result from the truncation of the model, which is often done to improve computer-processing time. For example, there are errors from the inability to accurately model the Earth's gravity. For this research, the process noise (force model) errors due to gravity are related to the gravity field that is above the 4 X 4 complex model.[11] The force model used to model the forces on the debris object is important in determining the orbit accurately. Forces can include complex gravity models, drag, lunar, and solar perturbations.[4] Higher-order integration schemes such as Runge-Kutta Order 4 (RK4) or higher are employed to evaluate the models. Drag is the only perturbation that has not been determined accurately, since drag depends on the atmosphere that is constantly changing.[4]

Orbit determination requires analysis of nonlinear force models. The nonlinear least squares method and the extended Kalman filter approximate nonlinear force models with linear models.[9] Linearization can produce unstable Kalman filters if the assumptions of local linearity are violated.[10] This relates to the uncertainty in the initial state estimate. This can also be applied to the nonlinear least squares method. Furthermore, the derivation of the Jacobian matrices are nontrivial in most applications and can be difficult to implement.[10] The Jacobian matrices for this research have complete solutions. The Kalman filter is often modified to improve the accuracy of the state updates and the covariances. Small perturbations that affect the object cannot be modeled correctly by the technique. Furthermore, there are errors in the method of obtaining the actual position and velocity data from the ground sensors. The Kalman filter also contains errors in the predictions from linearization of the force models.[9] Methods have been developed to reduce the linearizations to improve the performance and orbit prediction of the extended Kalman filter. It has been determined that the modified techniques developed do not provide significant improvement to the extended Kalman filter.[9] Consequently, the unmodified extended Kalman filter is an appropriate technique to use for this research. The extended Kalman filter should also show a significant improvement to the nonlinear batch least squares method.[11] However, the extended Kalman filter is not normally used for real world applications as it has unstable characteristics that can lead to inaccurate orbit estimates. As such, the extended Kalman filter often requires specific tuning for each application to give accurate estimates.[8] Initialized error covariances that can tune the extended Kalman filter used in this research are further explained in chapter 7.

There are not enough sensors to continuously track objects orbiting the Earth. Computer programs estimate the orbit from sensor observations and predict the location of the object to the next available sensor.[8] Orbit determination accuracy depends on the amount of observations available and the computer processing of the data.[12] The covariance is affected by the number of observations used. Dense observational data is considered as hundreds of observations of the object in a single pass over the sensor site.[12] The nonlinear least squares covariance relies on the central limit theorem. This theorem states that the distribution develops into a standard Gaussian distribution if the sample size increases.[12] The batch least squares method minimizes the sum of the square of the residuals and obtains a statistically accurate covariance matrix. Consequently, a sufficient amount of observations is required for the orbit determination technique to work.[12] The extended Kalman filter should also improve if more observations are used.

The orbit determination problem can be summarized into three main points.[11] First, accuracy in the orbit determination techniques depends on how good the initial guess is. Specifically, a closer initial guess to the true state is better. Second, more measurements give a better estimate of the true solution. Third, longer fit spans require better initial guesses. This will increase the iterations and provide longer accuracy in the estimated states. The extended Kalman filter iterations depend on the nonlinearity of the system force model; however, one iteration is usually sufficient.[4] The batch least squares method is iterated until an arbitrary convergence between the current and previous state estimates is achieved.[4]

1.5 Research Specifications

Debris objects are modeled with ballistic coefficients and orbital parameters that could collide with the ISS. In addition, ground-based sensor sites that are part of the Space Surveillance Network are represented.

1.5.1 Orbital Debris Parameters

The average mass density of orbital debris is 4.0 g/cm^3.[13] As mentioned before, large debris objects are typically defined as objects larger than 10 cm in diameter. Such objects are capable of being tracked and cataloged by USSPACECOM. Furthermore, their orbital elements are maintained.[3] Objects smaller than 10 cm in diameter cannot be tracked individually.[14] However, objects as small as 3 mm can be detected by ground sensor sites and used to obtain a statistical estimate of the number of medium sized debris objects in orbit.[7] To assess the risk of particularly dangerous 1 cm to 10 cm debris objects, scientists at ESA and other space organizations use sophisticated probability models and software. Risk of collision is determined using a spacecraft's cross-sectional area, orbital altitude, flight path and other factors.[14] Simulations for this research are made for an 11 cm (in diameter) large debris object with properties given in Table 1.1. Table 1.1 drag coefficient value is an estimate.

Table 1.1: Debris Object Parameters

C_d	0.0001	
m	2.78	kg
A	0.01	m^2

1.5.2 ISS Orbit Parameters

The altitude of the debris object in the ISS orbit is taken as $h_{alt} = 354\,\text{km}$. This is the average height of the ISS orbit.[15] The true initial position and velocity of the debris object in the ISS orbit is taken as the true ISS position and velocity (beginning on orbit 1998).[16]

$$\vec{r}_0 = [4187.27834, 3150.07678, 4204.17024]^{\mathrm{T}}\ \text{(km)}$$
$$\vec{v}_0 = [\text{-}1.396521475, 6.674056120, \text{-}\ 3.589135869]^{\mathrm{T}}\ \text{(km/s)}$$

A MATLAB function is used to check for the inclination and period of the orbit using the given initial position and velocity. The inclination is calculated to be approximately 51.3 degrees. The ISS inclination is referenced at 51.6 degrees.[17] Furthermore; the period is obtained to be approximately 1.52 hours. This generates approximately 15.75 rev/day for the debris object in the chosen orbit. Reference [17] indicates that the ISS orbit on average has 15.65 rev/day. Therefore, the true initial ISS position and velocity chosen for the debris object is acceptable. To obtain the inclination, the orbit specific angular momentum needs to be evaluated using equation (1.1).[18] $\vec{h}_s = \lfloor (h_s)_i \quad (h_s)_j \quad (h_s)_k \rfloor$ are the components of the specific angular momentum. The magnitude of the angular momentum is given as h_s. The inclination, i, is then calculated using equation (1.2).[18] The period is obtained using equations (1.3-1.4).[18] a is the semi-major axis of the orbit. The MATLAB algorithm used to calculate the inclination and period is described in chapter 8.

$$\vec{h}_s = \vec{r}_0 x \vec{v}_0 \tag{1.1}$$

$$i = \cos^{-1}\left(\frac{(h_s)_k}{h_s}\right) \tag{1.2}$$

$$a = \left(\frac{2}{r} - \frac{v^2}{\mu}\right)^{-1} \tag{1.3}$$

$$\mathrm{p} = 2\pi\sqrt{\frac{a^3}{\mu}} \tag{1.4}$$

Important Earth parameters used for this research problem are given in Table 1.2. Table 1.2 parameters were obtained from reference [18]. μ is the Earth's gravitational parameter and ω_e is the Earth's angular speed. R is the radius of the Earth.

Table 1.2: Earth Parameters

μ	398600.4418	km^3/s^2
R	6378.137	km
ω_e	7.2921158553e^{-5}	rad/s

1.5.3 Space Surveillance Network

Trackers need to be modeled based on the type and geographical distribution similar to that of the USSTRATCOM Space Surveillance Network (SSN) as shown in Figure 1.2 (reference [19]).

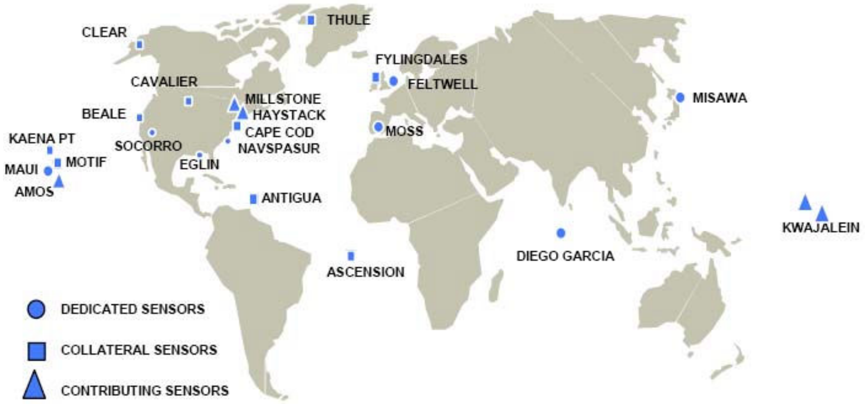


Figure 1.2: Space Surveillance Network

Table 1.3 gives longitude, latitude and local sidereal time (LST) data for ground-based SSN sensors used in this research problem. Table 1.3 LST, θ, data is validated in chapter 3, section 3.3. The local sidereal time equations are provided in chapter 3, section 3.2. The longitude and latitude was taken from reference [18]. North and East are positive.

Table 1.3: SSN Sensor Parameters

Location	**Longitude (deg)**	**Latitude (deg)**	θ **(deg)**
Eglin, FL	-86.21	30.57	58.42
Cavalier AFS, ND	-97.90	48.72	46.73
Clear AFB, AK	-149.19	64.29	-4.56
Maui, HI	-156.26	20.71	-11.63
Socorro, NM	-106.66	33.82	37.97
Cape Cod, MA	-70.54	41.75	74.09
RAF Fylingdales Moor, UK	-0.67	54.37	143.96
Beale AFB, CA	-121.35	39.14	23.28
NAVSPASUR Fence (9 sites)	-87 to -117	≈ 33	57.63 to 27.63

1.6 SSN Classifications

There are three classifications for the SSN sensors. Dedicated sensors have a primary mission that includes the detection, tracking, identification, cataloging, and characterization of orbital debris objects.[20] The following dedicated sensors are used for this research:

- Eglin, Florida
- Maui, Hawaii
- NAVSPASUR Fence
- Socorro, New Mexico

Collateral sensors have a primary mission other than space surveillance. These sensors also provide support to the space surveillance mission in the same way as dedicated sensors.[21] The following collateral sensors are used for this research:

- Clear AFB, Alaska
- Cape Cod, Massachusetts
- Beale AFB, California
- Fylingdales, United Kingdom
- Cavalier AFS, North Dakota

Contributing sensors provide observation data on satellites to USSPACECOM on a contributing basis. These sensors are not operated by USSPACECOM.[22]

1.7 SSN Sensor Background

Table 1.4 provides the system and type information for the SSN sensors used for this research. The maximum range for each sensor was obtained from reference [18]. Sensor sites (1-8) system and type information was obtained from reference [23]. Sensor site 9 system and type information was obtained from reference [18].

Table 1.4: SSN Sensor System/Type and Maximum Range

Sensor No.	Location	System/Type	Maximum Range (km)
1	Eglin, FL	Phased Array Radar	13,210
2	Cavalier, ND	PARCS/Phased Array Radar	3300
3	Clear, AK	BMEWS/Phased Array Radar	4910
4	Maui, HI	GEODSS	---
5	Socorro, NM	GEODSS	---
6	Cape Cod, MA	PAVE PAWS/Phased Array Radar	5555
7	Fylingdales, UK	BMEWS/Phased Array Radar	4820
8	Beale, CA	PAVE PAWS/Phased Array Radar	5555
9	NAVSPASUR Fence (9 sites)	Interferometer	---

Phased array radar sites obtain range, azimuth, and elevation measurements from the tracked orbital debris. A phased array consists of a group of antennas. The relative phases of the signals applied to the antennas are varied so that the radiation patter of the antenna array is reinforced in a desired direction and suppressed in an undesired direction.[24] Figure 1.3 is an image of the phased array radar site located in Cavalier.[19]

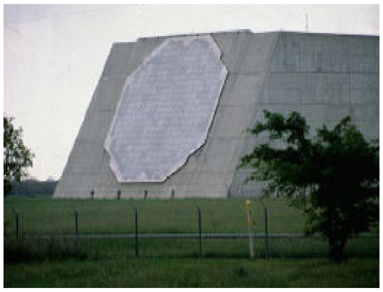


Figure 1.3: Cavalier Phased Array Radar

The NAVSPASUR (Naval Space Surveillance System) fence is composed of nine stations (3 transmitters and 6 receivers). The nine stations are all located on approximately the 33rd parallel of the United States.[18] The fence can detect near-Earth space objects that have orbital inclinations of 33 degrees or greater.[25] The three transmitter sites emit a planar and continuous-wave vertical fan of radar energy. When a debris object passes through the fan, the energy is reflected back towards the Earth.[18] The six receiver sites collect the transmitted energy reflected from satellites or debris objects as they pass through the fence.[26] Each receiver site has individual antennas located at defined intervals. The longest antenna at each site is the "alert" antenna. This antenna is very sensitive and is able to detect a signal before the other antennas. The "alert" antenna is used to electronically alert the system controller to the presence of an orbital object. The controller then tunes the receiver to the correct frequency of the energy that is reflected from the debris object. Two receiver sites, Elephant Butte and Hawkinsville, are "high-altitude" sites. These receivers have higher gain and sensitive electronic systems that enable the detection of reflected energy from objects in higher altitudes.[26] Figure 1.4 is an image of the NAVSPASUR fence showing the fan of radar energy emitted from the three transmitter sites.[27] A data processing center located in Dahlgren, Virginia is shown in Figure 1.4. Figure 1.5 is an image of the NAVSPASUR fence showing the locations of the six receiver sites.[26] The measurement data processed from the NAVSPASUR fence is direction cosines that are converted to azimuth and elevation angles. Further processing (triangulation) can provide rough estimates of the range. However, the range is usually two to three times less accurate than the azimuth and elevation measurements.[18] Range measurements are not included for the NAVSPASUR fence. Table 1.4 reflects this limitation.

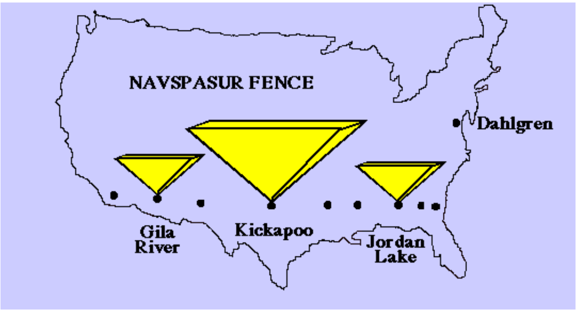


Figure 1.4: NAVSPASUR Fence

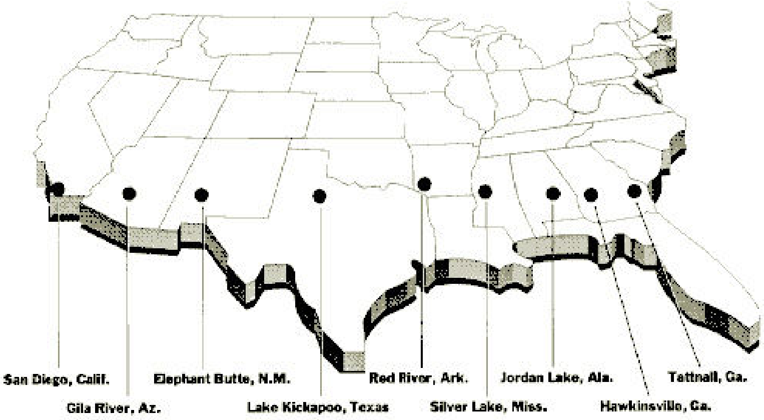


Figure 1.5: NAVSPASUR Fence Receiver Sites

GEODSS sites are optical systems that use charge-coupled device cameras to enable digital image processing.[18] Debris objects are detected by the system close to real time. The GEODSS sites use sidereal drive telescopes to remove the visible movement of stars across the night sky that result from the rotation of the Earth. This movement causes debris objects to appear as streaks of light against a background of point light sources. Removing successive frames of video in the camera processor can eliminate the stationary star field. The result is the debris object without the star field. This improves the detectability of faint objects in dense star backgrounds.[28] The outputs of GEODSS sites are direction measurements consisting of right ascension and declination. The azimuth and elevation can be obtained from the right ascension and declination data.[18] Range measurements are not included for the GEODSS sites. Table 1.4 reflects this limitation. In general, the accuracy of data from the phased array radar sites exceeds that of the optical systems; enabling more accurate orbit determination.[28] The results for this research should confirm this. Figure 1.6 is an image of a GEODSS site.[19]

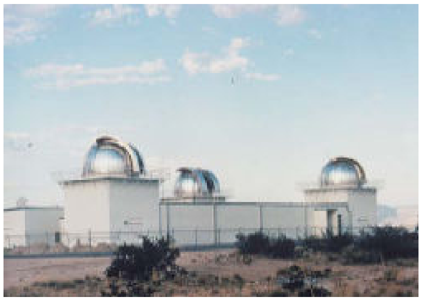


Figure 1.6: GEODSS Site

1.8 Reference System Background

To understand the information presented in this paper a brief discussion of the Earth-Centered Inertial reference frame, topocentric reference frame, and Universal Time is required.

1.8.1 ECI Reference System

The Earth-Centered Inertial (ECI) reference system is also called the geocentric equatorial coordinate system.[18] $\hat{i}_3$ is along the Earth's rotational axis pointing North. $\hat{i}_1$ points in the direction of the vernal equinox. The vernal equinox is a line representing the intersection of the Earth's equatorial plane and the plane of the Earth's orbit around the Sun (ecliptic). $\hat{i}_2$ completes the right-handed orthogonal system for the ECI coordinate system.[29] Longitude (geocentric) is measured positive East of Greenwich. Greenwich is also referred to as the prime meridian (0 degrees longitude).[18] A meridian is an imaginary line on the Earth's surface that connects locations using longitude from the North to South Pole.[30] Latitude (geocentric) is positive above (North) Earth's equator and negative below. Right ascension is measured positive East of the vernal equinox. Declination is measured positive from the equator to the object's location.[18] Diagrams showing these parameters are provided in chapters 2 and 3. Figure 1.7 is a diagram showing the vernal equinox.[31] Figure 1.8 is a diagram showing the prime meridian.[32]

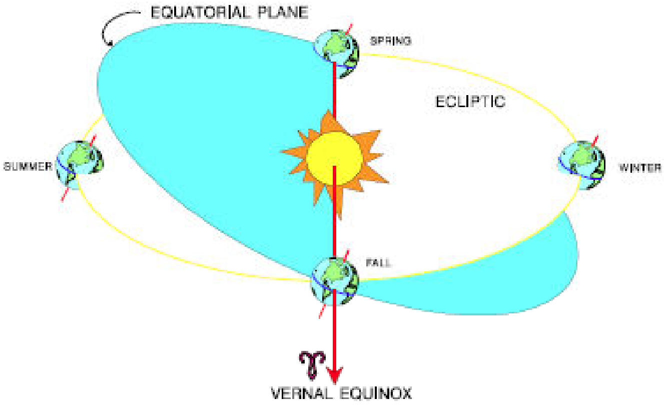


Figure 1.7: Vernal Equinox

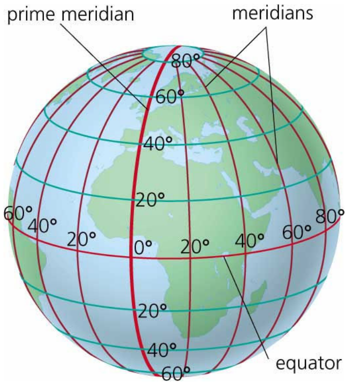


Figure 1.8: Prime Meridian

1.8.2 Topocentric Coordinate System

The topocentric coordinate system is commonly used with sensor sites.[18] The origin is translated from the Earth's center to the origin of the observer's local horizon.[18] The orientation of the axes is the same as the ECI coordinate system. The topocentric system has the $\hat{i}_3$ axis pointing toward the zenith, the $\hat{i}_1$ axis pointing South, and the $\hat{i}_2$ axis pointing East.[33] To transform to the topocentric-horizon system, a rotation is required through an angle θ (local sidereal time) about the $\hat{i}_3$ axis (Earth rotation axis) and then through an angle $(90^0 - \phi)$ (ϕ is the observer's latitude) about the $\hat{i}_2$ axis.[18] The topocentric reference frame has coordinates (ρ_s, ρ_e, ρ_u). Equations (1.5-1.6) provide the two elementary rotation matrices required for the conversion.[18] The elementary rotation about the $\hat{i}_2$ axis is given by equation (1.5). The elementary rotation about the $\hat{i}_3$ axis is given by equation (1.6). The elementary rotations provided do not account for topocentric angle rotation requirements. This is further discussed in chapter 4. Figure 1.9 provides a diagram of the ECI and topocentric coordinate system.[33] Range, azimuth, and elevation are obtained in the topocentric coordinate system. The conversions are developed in chapter 4.

$$C_2(\phi) = \begin{bmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{bmatrix} \tag{1.5}$$

$$C_3(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.6}$$

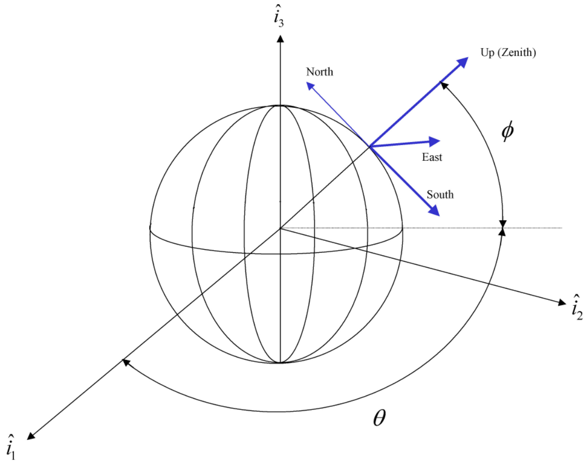


Figure 1.9: Topocentric-Horizon Coordinate System

1.8.3 Universal Time

There are three forms of Universal Time (UT). The differences between UT0, UT1, and UT2 are small.[18] Universal Time (UT) is a recent version of the Greenwich Mean Time (GMT) that is based on the rotation of the Earth.[34] GMT is the mean solar time on the prime meridian. Solar time is the interval between successive transits of the Sun over a local meridian. Alternatively, sidereal time is the time between successive transits of the stars over a local meridian.[18] Sidereal time is further explained in chapter 3.

The old version of GMT is split into Coordinated Universal Time (UTC) and UT1. UTC is based on atomic time and it approximates UTI within $\pm 0.9^{s}\ (\Delta UT1 = UT1 - UTC)$. UTC is the foundation for civil time systems and it is used for ordinary clocks. UT0 is Universal Time determined at an observatory by observing the diurnal (repeating daily) motion of stars.[18] UT0 does not account for polar motion. The Earth's axis of symmetry does not align with the axis of rotation and the Earth's movement about the center of mass causes polar motion.[34] UT1 corrects UT0 by including the polar motion effect. UT1 is also independent of the observatory location.[18] UT2 corrects UT1 for observed seasonal variations in Earth's rotational speed.[34] UT1 is employed for this research, since UT2 is rarely used and most problems evaluate UT1.[18]

CHAPTER 2

ORBITAL DEBRIS DYNAMIC MODEL

2.1 Dynamic Model

The basic two-body orbital equation of motion is used as the dynamic model for this research problem.[35] (x, y, x) are the ECI position coordinates of the debris object.

$$\vec{a} = -\frac{\mu}{r^3}\vec{r} \tag{2.1}$$

Where,

$$\begin{aligned} \vec{r} &= x\vec{i} + y\vec{j} + z\vec{k} \\ r &= \|\vec{r}\| = \sqrt{x^2 + y^2 + z^2} \end{aligned} \tag{2.2}$$

Figure 2.1 is a MATLAB plot of the basic two-body orbital equation of motion using the initial position and velocity of the ISS. The orbit is propagated for 10 hours.

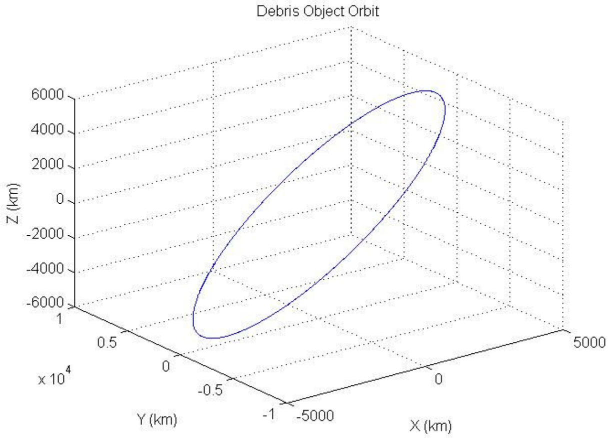


Figure 2.1: Debris Object Orbit (without Perturbations)

Perturbations such as acceleration due to atmospheric drag, $\vec{a}_d$, and higher-order gravity, $\vec{a}_g$, are incorporated into the dynamic model given in equation (2.1). The perturbations are further discussed in subsequent sections in this chapter.

$$\vec{a} = -\frac{\mu}{r^3}\vec{r} + \vec{a}_d\left(\vec{r},\vec{v},t\right) + \vec{a}_g\left(\vec{r},t\right) \tag{2.3}$$

Where,

$$\begin{aligned}
&\vec{a}_d\left(\vec{r},\vec{v},t\right) = a_{dx}\vec{i} + a_{dy}\vec{j} + a_{dz}\vec{k} \\
&\vec{a}_g\left(\vec{r},t\right) = a_{gx}\vec{i} + a_{gy}\vec{j} + a_{gz}\vec{k} \\
&\vec{v} = \dot{x}\vec{i} + \dot{y}\vec{j} + \dot{z}\vec{k}
\end{aligned} \tag{2.4}$$

Figure 2.2 is a MATLAB plot of the basic two-body orbital equation of motion with perturbations using the initial position and velocity of the ISS. The orbit is propagated for 10 hours.

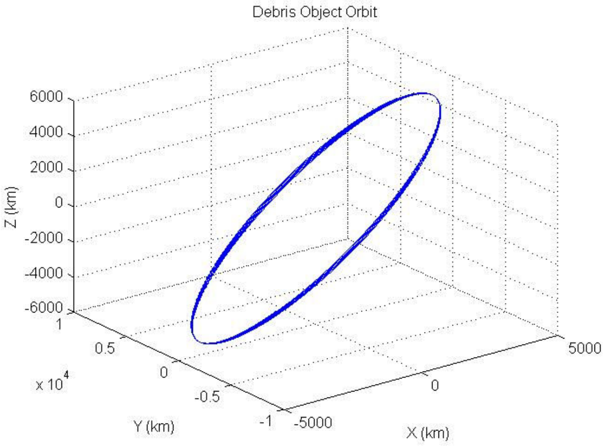


Figure 2.2: Debris Object Orbit (with Perturbations)

2.1.1 Perturbation Requirements

The dynamic model given by equation (2.3) includes a 4 X 4 complex gravity model and an atmosphere model. Figure 2.3 is a diagram showing important parameters used in the orbit determination problem. Figure 2.4 shows the spherical coordinates (r,δ,γ). The parameters are explained in subsequent sections of this report. Figures (2.3-2.4) were adapted from reference [35].

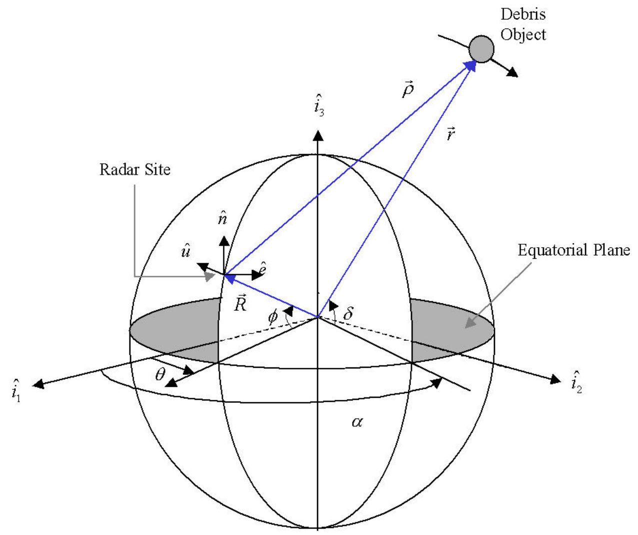


Figure 2.3: Geometry of Earth Observations of Debris Object Motion

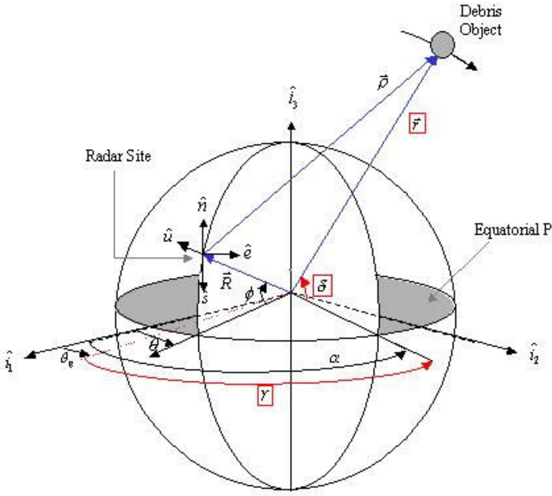


Figure 2.4: Geometry of Earth Observations of Debris Object Motion Specifying Spherical Coordinates

2.2 Complex Gravity Model

Acceleration due to gravity is based on the spherical harmonic representation of Earth's geopotential function. Spherical harmonics are often used to approximate the shape of the geoid. The geoid, which is defined by the Earth's gravity field, is a surface of equal gravitational potential. It follows a hypothetical ocean surface at rest (in the absence of tides and currents).[36] The gravity potential is perpendicular and equal at all points along the geoid.[18] Currently, the best set of spherical harmonic coefficients is EGM96. This model was determined in an international collaborative project led by NIMA (US agency).[36] EGM96 is further described in section 2.2.1. The mathematical description of geopotential model is given by the following series expansion in spherical harmonics: [37]

$$U(r,\delta,\gamma)=\frac{\mu}{r}+\frac{\mu}{r}\sum_{n=1}^{\infty}C_n^0\left(\frac{R}{r}\right)^n P_n^0(\sin\delta)+\frac{\mu}{r}\sum_{n=1}^{\infty}\sum_{m=1}^{n}\left(\frac{R}{r}\right)^n P_n^m(\sin\delta)\left[S_n^m\sin m\gamma+C_n^m\cos m\gamma\right] \tag{2.4}$$

where, $R=\|\vec{R}\|$ is the radius of the Earth. The higher-order terms given by equation (2.4) are modeled using Legendre functions. The geocentric latitude of the debris object can be obtained from the following equation: [37]

$$\delta=\sin^{-1}\left(\frac{z}{r}\right) \tag{2.5}$$

r is the geocentric distance to the debris object. The right ascension of the debris object can be obtained using the following equation: [37]

$$\alpha=\tan^{-1}\left(\frac{y}{x}\right) \tag{2.6}$$

The geocentric longitude of the debris object can be obtained from the following equation: [37]

$$\gamma=\alpha-\theta_g \tag{2.7}$$

θ_g is the right ascension of Greenwich. θ_g is also known as the Greenwich Mean Sidereal Time (GMST). GMST is described in chapter 3. Acceleration due to gravity is obtained by taking the gradient of the potential function: [38]

$$\vec{a}_g(\bar{r},t)=\nabla U(\bar{r},t) \tag{2.8}$$

This acceleration vector is a combination of pure two-body (point mass) gravity acceleration and the gravitational acceleration due to higher order non-spherical terms in the Earth's geopotential.[38] The Legendre functions are often differentiated in spherical coordinates (r,δ,γ). To obtain the acceleration in cartesian coordinates the chain rules needs to be applied as shown in equation (2.9).[37]

$$\vec{a}_g(\vec{r},t)=\frac{\partial U}{\partial r}\left(\frac{\partial r}{\partial \vec{r}}\right)^T+\frac{\partial U}{\partial \delta}\left(\frac{\partial \delta}{\partial \vec{r}}\right)^T+\frac{\partial U}{\partial \gamma}\left(\frac{\partial \gamma}{\partial \vec{r}}\right)^T \tag{2.9}$$

The three partial derivatives of the geopotential, U, with respect to (r,δ,γ) are given by the following equations: [37]

$$\begin{aligned}
\frac{\partial U}{\partial r}&=-\frac{1}{r}\left(\frac{\mu}{r}\right)\sum_{n=2}^{\infty}\left(\frac{R}{r}\right)^n(n+1)\sum_{m=0}^{n}\left(C_n^m\cos m\gamma+S_n^m\sin m\gamma\right)P_n^m(\sin\delta)\\
\frac{\partial U}{\partial \delta}&=\left(\frac{\mu}{r}\right)\sum_{n=2}^{\infty}\left(\frac{R}{r}\right)^n\sum_{m=0}^{n}\left(C_n^m\cos m\gamma+S_n^m\sin m\gamma\right)\left[P_n^{m+1}(\sin\delta)-(m\tan\delta)P_n^m(\sin\delta)\right]\\
\frac{\partial U}{\partial \gamma}&=\left(\frac{\mu}{r}\right)\sum_{n=2}^{\infty}\left(\frac{R}{r}\right)^n\sum_{m=0}^{n}m\left(S_n^m\cos m\gamma-C_n^m\sin m\gamma\right)P_n^m(\sin\delta)
\end{aligned} \tag{2.10}$$

The partial derivatives of (r,δ,γ) with respect (x,y,z) are obtained from the following expressions:[37]

$$\begin{aligned}
\frac{\partial r}{\partial \vec{r}}&=\frac{\vec{r}^{\,T}}{r}\\
\frac{\partial \delta}{\partial \vec{r}}&=\frac{1}{\sqrt{x^2+y^2}}\left(-\frac{\vec{r}^{\,T}z}{r^2}+\frac{\partial z}{\partial \vec{r}}\right)\\
\frac{\partial \gamma}{\partial \vec{r}}&=\frac{1}{x^2+y^2}\left(x\frac{\partial y}{\partial \vec{r}}-y\frac{\partial x}{\partial \vec{r}}\right)
\end{aligned} \tag{2.11}$$

where,

$$\begin{aligned}
\frac{\partial x}{\partial \vec{r}}&=\begin{bmatrix}1 & 0 & 0\end{bmatrix}\\
\frac{\partial y}{\partial \vec{r}}&=\begin{bmatrix}0 & 1 & 0\end{bmatrix}\\
\frac{\partial z}{\partial \vec{r}}&=\begin{bmatrix}0 & 0 & 1\end{bmatrix}
\end{aligned} \tag{2.12}$$

Putting equations (2.10) and (2.11) back into equation (2.9) the components of the debris object's acceleration vector are obtained to be: [37]

$$
\begin{aligned}
a_x &= \left(\frac{1}{r}\frac{\partial U}{\partial r} - \frac{z}{r^2\sqrt{x^2+y^2}}\frac{\partial U}{\partial \delta} \right) x - \left(\frac{1}{x^2+y^2}\frac{\partial U}{\partial \gamma} \right) y \\
a_y &= \left(\frac{1}{r}\frac{\partial U}{\partial r} - \frac{z}{r^2\sqrt{x^2+y^2}}\frac{\partial U}{\partial \delta} \right) y + \left(\frac{1}{x^2+y^2}\frac{\partial U}{\partial \gamma} \right) x \\
a_z &= \left(\frac{1}{r}\frac{\partial U}{\partial r} \right) z + \left(\frac{\sqrt{x^2+y^2}}{r^2}\frac{\partial U}{\partial \delta} \right)
\end{aligned}
\qquad (2.13)
$$

Derivations are shown in Appendix B. The Legendre polynomials with argument $\sin\delta$ are computed using recursion relationships given by: [37]

$$
\begin{aligned}
&P_n^0(\sin\delta) = \frac{1}{n}\left[(2n-1)\sin\delta\, P_{n-1}^0(\sin\delta) - (n-1)\,P_{n-2}^0(\sin\delta)\right] \\
&P_n^m(\sin\delta) = P_{n-2}^m(\sin\delta) + (2n-1)\cos\delta\, P_{n-1}^{m-1}(\sin\delta), \qquad \mathrm{m} \neq 0, \mathrm{m} < \mathrm{n} \\
&P_n^n(\sin\delta) = (2n-1)\cos\delta\, P_{n-1}^{n-1}(\sin\delta), \qquad \mathrm{m} \neq 0, \mathrm{m} = \mathrm{n} \\
&P_n^m = 0, \quad m > \mathrm{n}
\end{aligned}
\qquad (2.14)
$$

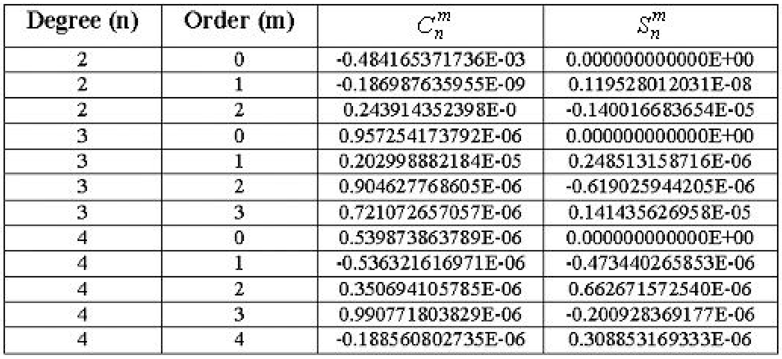
The associated Legendre functions for a 4 X 4 gravity model are given by: [18]

$$
\begin{aligned}
&P_0^0(\sin\delta) = 1 \\
&P_1^0(\sin\delta) = \sin\delta \\
&P_1^1(\sin\delta) = \cos\delta \\
&P_2^0(\sin\delta) = \frac{1}{2}\left(3\sin^2\delta - 1\right) \\
&P_2^1(\sin\delta) = 3\sin\delta\cos\delta \\
&P_2^2(\sin\delta) = 3\cos^2\delta \\
&P_3^0(\sin\delta) = \frac{1}{2}\left(5\sin^3\delta - 3\sin\delta\right) \\
&P_3^1(\sin\delta) = \frac{1}{2}\cos\delta\left(15\sin^2\delta - 3\right)
\end{aligned}
\qquad
\begin{aligned}
&P_3^2(\sin\delta) = 15\cos^2\delta\sin\delta \\
&P_3^3(\sin\delta) = 15\cos^3\delta \\
&P_4^0(\sin\delta) = \frac{1}{8}\left(35\sin^4\delta - 30\sin^2\delta + 3\right) \\
&P_4^1(\sin\delta) = \frac{5}{2}\cos\delta\left(7\sin^3\delta - 3\sin\delta\right) \\
&P_4^2(\sin\delta) = \frac{15}{2}\cos^2\delta\left(7\sin^2\delta - 1\right) \\
&P_4^3(\sin\delta) = 105\cos^3\delta\sin\delta \\
&P_4^4(\sin\delta) = 105\cos^4\delta
\end{aligned}
\qquad (2.15)
$$

2.2.1 EGM96 Gravity Model

EGM96 is a geopotential model of the Earth consisting of spherical harmonic coefficients complete to degree and order 360. For this research, a 4 X 4 model is used. Table 2.1 shows the spherical harmonic coefficients for different degree and order values. Table 2.1 shows that the degree and order of the spherical harmonic coefficients start with $(n = 2, m = 0)$. Table 2.1 was adapted from reference [39].

Table 2.1: 4 X 4 EGM96 Gravity Model



Degree (n)	Order (m)	C_n^m	S_n^m
2	0	-0.484165371736E-03	0.000000000000E+00
2	1	-0.186987635955E-09	0.119528012031E-08
2	2	0.243914352398E-0	-0.140016683654E-05
3	0	0.957254173792E-06	0.000000000000E+00
3	1	0.202998882184E-05	0.248513158716E-06
3	2	0.904627768605E-06	-0.619025944205E-06
3	3	0.721072657057E-06	0.141435626958E-05
4	0	0.539873863789E-06	0.000000000000E+00
4	1	-0.536321616971E-06	-0.473440265853E-06
4	2	0.350694105785E-06	0.662671572540E-06
4	3	0.990771803829E-06	-0.200928369177E-06
4	4	-0.188560802735E-06	0.308853169333E-06

By definition, the spherical harmonic coefficient $\left(S_n^0\right)$ is zero.[18] If the center of the coordinate system coincides with the attracting body's center of mass, the spherical harmonic coefficients $\left(C_0^1, C_1^1, \text{and } S_1^1\right)$ are zero.[18] In general, the zeroth term is contained in the $\frac{\mu}{r}$ term and the first degree $(n = 1)$ terms are zero.[18]

2.2.2 Spherical Harmonic Coefficients

There are three types of spherical harmonic coefficients. Zonal harmonics $(m = 0)$ is the zeroth order.[18] It takes into account the extra mass distribution in latitudinal regions.[18] Equation (2.16) provides the J notation commonly used for zonal harmonics (i.e. J_2, J_3, etc…).[18]

$$J_n = -C_n^0 \tag{2.16}$$

Sectorial harmonics $(m = n)$ take into account the extra mass distribution in longitudinal regions.[18] Tesseral harmonics $(n \neq m \neq 0)$ attempt to model specific regions on the Earth which depart from a perfect sphere.[18] Figure 2.5 is an illustration of the three types of spherical harmonics.[40] Figure 2.6 is an illustration of the spherical harmonics with nodal surfaces $(n = 0,1,2,3)$.[40] The second and third terms of the geopotential function given by equation (2.4) are the non-spherical potential due to the sum of the zonal and tesseral harmonic respectively.[37] The first term is referred to as the point-mass potential for Keplerian motion.[37]

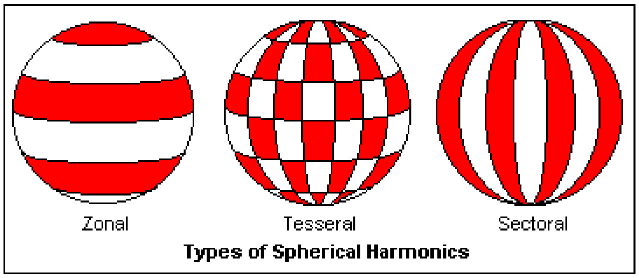


Figure 2.5: Types of Spherical Harmonics

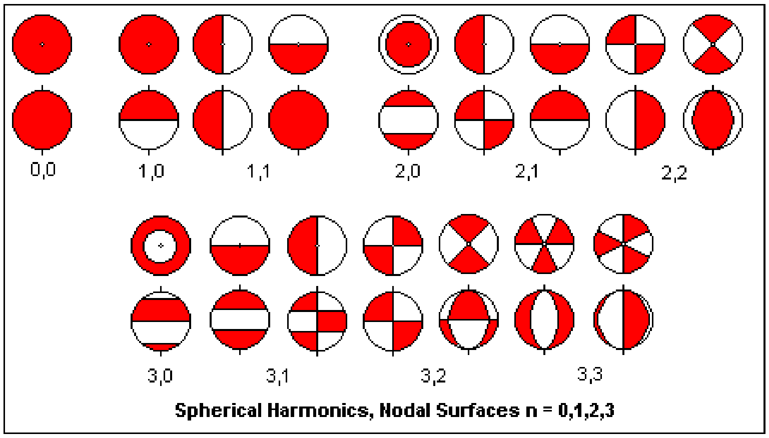


Figure 2.6: Nodal Surfaces $(n = 0,1,2,3)$

2.3 Atmosphere Model

Atmospheric drag is the dominating perturbation for satellites with perigee altitudes between 120 km to 600 km.[4] The debris object chosen for this research is small enough to where the gravity perturbations are larger than the atmospheric perturbations. An exponential model for atmospheric density is used as shown in the following equation: [18]

$$\rho_{atm} = \rho_o EXP\left[-\frac{h_{alt} - h_o}{H_{alt}}\right] \tag{2.17}$$

The acceleration due to atmospheric drag is given by: [38]

$$\vec{a}_d(\vec{r},\vec{v},t) = -\frac{1}{2}\frac{C_d A}{m}\rho_{atm}\|\vec{v}_r\|\vec{v}_r \tag{2.18}$$

The atmosphere is assumed to rotate at the same angular speed as the Earth.[38] This assumption is taken into account by equation (2.19). Equations (2.19-2.21) are given in reference [38].

$$\vec{v}_r = \vec{v} - \vec{\omega} \times \vec{r} \tag{2.19}$$

$$\vec{\omega} = \omega_e \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.20}$$

$$\vec{v}_r = \begin{bmatrix} \dot{x} + \omega_e x \\ \dot{y} - \omega_e y \\ \dot{z} \end{bmatrix} \tag{2.21}$$

CHAPTER 3

SENSOR NETWORK

3.1 Sensor Site Parameters

The longitude and latitude of each sensor (observer) site is obtained and the local sidereal time (LST), θ, is calculated. Sidereal time is the hour angle of the vernal equinox relative to the local meridian (sensor site longitude). Sidereal time is measured positively in the counter-clockwise direction when viewed from the North pole.[18] Figure 3.1 and 3.2 show latitude, East longitude, LST, and Greenwich Mean sidereal time (GMST), θ_g, of the sensor site. GMST is the sidereal time of the Greenwich meridian.[18] Both Figures 3.1 and 3.2 were adapted from reference [41].

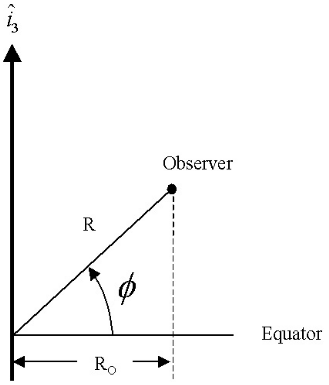


Figure 3.1: Sensor Site Latitude

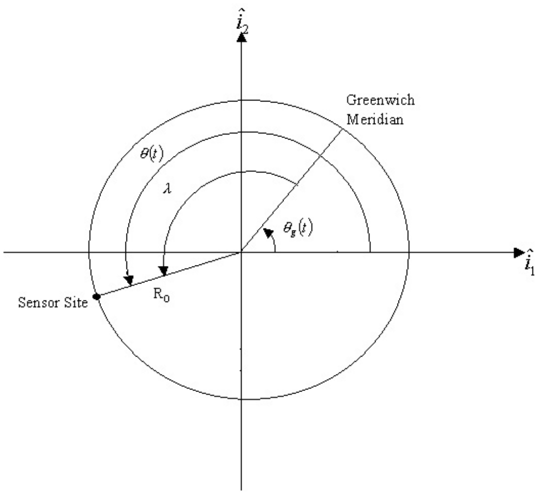


Figure 3.2: Sensor Site Longitude and Local Sidereal Time

3.2 Necessary Equations

The LST and (X,Y,Z), ECI position coordinates, of the sensor site can be obtained using the following equations: [41]

$$\begin{aligned} R_o &= R\cos(\phi) \\ \theta &= \theta_g + \lambda \\ Z &= R\sin(\phi) \\ X &= R_o\cos(\theta) \\ Y &= R_o\sin(\theta) \end{aligned} \tag{3.1}$$

where,

$$\vec{R} = X\vec{i} + Y\vec{j} + Z\vec{k} \tag{3.2}$$

To obtain the LST, the GMST needs to be calculated. To obtain GMST, the number of Julian centuries from a specified epoch needs to be evaluated. Epoch is defined as a particular instant in time.[18] The general formula for J2000 (January 1, 2000 12:00 TT) is given by equation (3.3).[18] TT is known as terrestrial time. UT1 is used for most problems.[18] As a result, equation (3.3) can be used to obtain T_{UT1}.

$$T_{xxx} = \frac{JD - 2451540}{36525} \tag{3.3}$$

then,

$$\theta_g = 67{,}310.54841^s + \left(876{,}600^h + 8{,}640{,}184.812866^s\right)T_{UT1} + 0.093104T_{UT1}^2 - 6.2\times10^{-6}T_{UT1}^3 \tag{3.4}$$

The Julian Date (JD) provides a continuous and simple method of preserving year-month-day-hour-minute-second information in one variable.[18] The fix function in MATLAB rounds the value towards zero resulting in an integer.

$$JD = 367(yr) - fix\left[\frac{7\left\{yr + fix\left(\frac{mo+9}{12}\right)\right\}}{4}\right] + fix\left(\frac{275mo}{9}\right) + d + 1721013.5 + \frac{\frac{\left(\frac{s}{60}+\min\right)}{60}+h}{24} \tag{3.5}$$

JD values are typically large. As a result, JD may affect accuracy for computer programs due to round off. The modified Julian date (MJD) given by equation (3.6) resolves this difficulty. MJD is used to reduce the size of the date by approximately 2 significant digits. MJD begins each day at midnight instead of noon.[18]

$$MJD = JD - 2400000.5 \tag{3.6}$$

To convert from seconds to degrees the following equation is used: [18]

$$1^s = \frac{1}{240^o} \tag{3.7}$$

3.3 Local Sidereal Time Validation

A comparison is made with an example given in reference [33]. In this example it is necessary to obtain the Julian Date, Greenwich Mean Sidereal Time, Local Sidereal Time, and the (X,Y,Z) coordinates in the ECI reference frame for the sensor site locations. This result was developed using MATLAB code. Chapter 8 explains the functions used for obtaining the ECI coordinates, LST, and GMST of the sensor sites. The example parameters were for a 9 hour UTC on October 01 1995. UTC uses equation (3.3). In addition, the example used a 40° North latitude and 75° West longitude observer location.

Example Solution	MATLAB Solution
$JD = 2{,}449{,}991.875$	$JD = 2{,}449{,}991.875$
$\theta_g = 144.627^\circ$	$\theta_g = 144.627^\circ$
$\theta = 69.627\ ^\circ$	$\theta = 69.627^\circ$
$\mathrm{X} = 1700.938\ \mathrm{km},\ \mathrm{Y} = 4580.302\ \mathrm{km},\ \mathrm{Z} = 4099.786\ \mathrm{km}$	$\mathrm{X} = 1700.938\ \mathrm{km},\ \mathrm{Y} = 4580.302\ \mathrm{km},\ \mathrm{Z} = 4099.786\ \mathrm{km}$

3.4 Two-Step Observability Check

A two-step approach is implemented to check if the debris object is observable by the sensor sites. The first step is to dot the sensor's local position vector into the debris object's position vector. If this result is positive then the debris object is above the local horizon. The second step is to check if the debris object is within the sensor’s azimuth, az, coverage. A mask angle is also implemented to further restrict the observability near the sensor's local horizon. A mask angle is the minimum elevation angle, el, in which the sensor site can observe the debris object.[42] For example, atmospheric distortions for optical systems (GEODSS) at elevations lower than 20 degrees limit accuracy of the observations.[18] The azimuth coverage and the minimum and maximum elevation angles for the sensor sites are given in Table 3.1. The azimuth and elevation angles are obtained using equations (4.4-4.5) that are provided in chapter 4. The azimuth is the angle measured from the North and clockwise around the local horizon to a debris object. The azimuth has values from 0 to 360 degrees.[18] The elevation is measured from the local horizon (positive up) to the debris object. The elevation has values from –90 to 90 degrees.[18] Figure 3.3 provides a view of the elevation angle as it relates to the local horizon of the observer. Figure 3.3 was adapted from reference [43]. Figure 3.4 is a diagram showing the sensor's local horizon. Figure 3.4 was adapted from reference [44]. Figure 3.5 gives a view of the sensor's local horizon with respect to the Earth (topocentric coordinate system).[45] Chapter 8 provides information on the MATLAB codes developed to perform the two-step observability check for each sensor site.

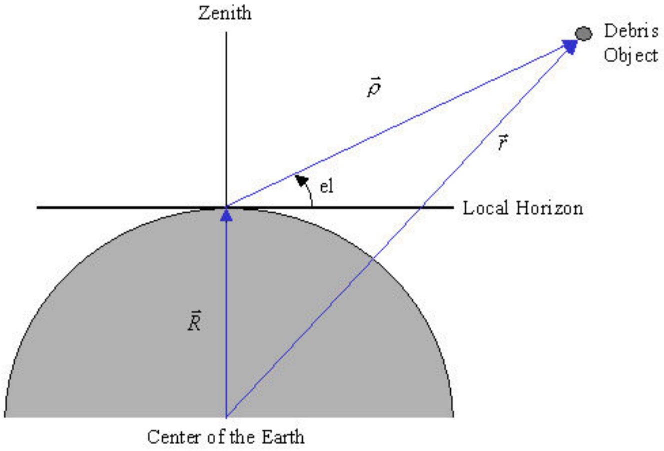


Figure 3.3: Local Horizon Elevation Angle

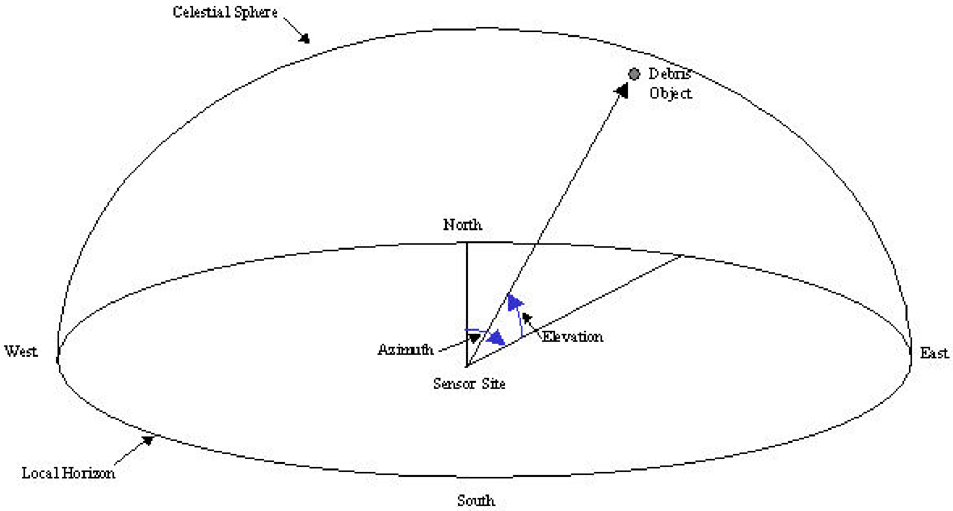


Figure 3.4: Sensor Site Local Horizon

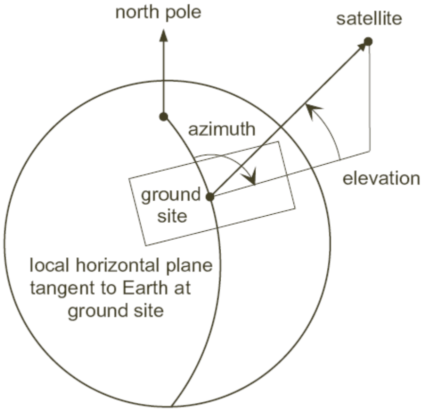


Figure 3.5: Sensor Site Local Horizon (Earth View)

3.5 Azimuth Coverage

Phased array sensors have 120, 240 or 360 degrees of azimuth coverage. This depends on the number of faces (1, 2 or 3) respectively. GEODSS sites have 360 degrees of azimuth coverage.[19] The NAVSPASUR fence detects orbital debris that fly directly overhead (zenith) through the fence. Therefore, the azimuth coverage is limited to 90 degrees and 270 degrees.[18] The zenith points radially outward from the sensor site along the local vertical.[18] The azimuth coverage for sensor sites (1-8) was obtained from reference [19]. This azimuth coverage was used to update the minimum and maximum azimuth coverage found in reference [18]. For example, sensor site 3 (Clear) has been upgraded to a 2-faced phased array sensor.[19] Therefore; the azimuth coverage has changed for sensor 3. Table 3.1 includes the updated minimum and maximum azimuth angles for the SSN sensor sites. The minimum and maximum elevation angles were obtained from reference [18].

Table 3.1: SSN Sensor Azimuth and Elevation Parameters

Sensor No.	Location	Azimuth Coverage (deg)	Minimum Azimuth (deg)	Maximum Azimuth (deg)	Minimum Elevation (deg)	Maximum Elevation (deg)
1	Eglin, FL	120	120	240	1	90
2	Cavalier, ND	120	302	62	2	45
3	Clear, AK	240	200	80	1	90
4	Maui, HI	360	0	360	20	90
5	Socorro, NM	360	0	360	20	90
6	Cape Cod, MA	240	347	227	3	80
7	Fylingdales, UK	360	0	360	4	70
8	Beale, CA	240	126	6	3	80
9	NAVSPASUR Fence (9 sites)	---	90 only	270 only	0	90

Figures (3.6-3.9) show the approximate azimuth coverage for five of the phased array sensor sites used in this research that have 1 to 2 faces. Figure 3.6 image of the Eglin sensor was adapted from reference [19]. Figure 3.7 image of the Clear sensor was adapted from reference [46]. Figure 3.8 image of the Cape Cod sensor was adapted from reference [47]. Figure 3.9 image of the Beale sensor was adapted from reference [48].

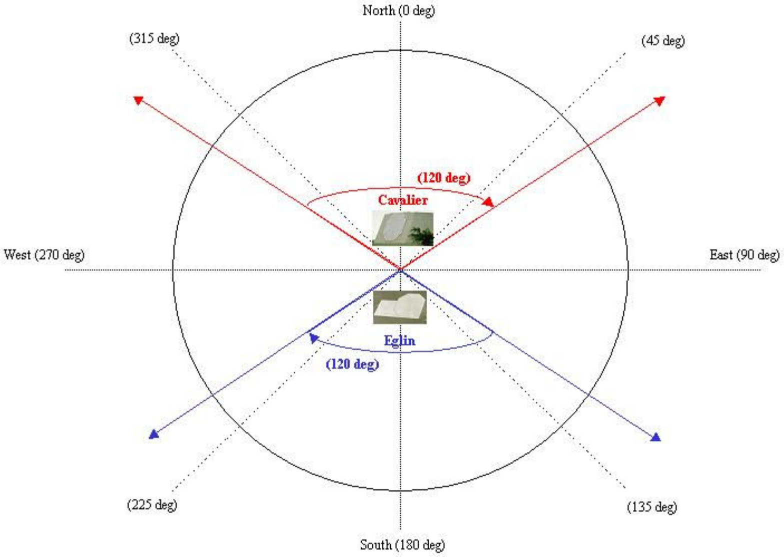


Figure 3.6: Eglin and Cavalier Azimuth Coverage

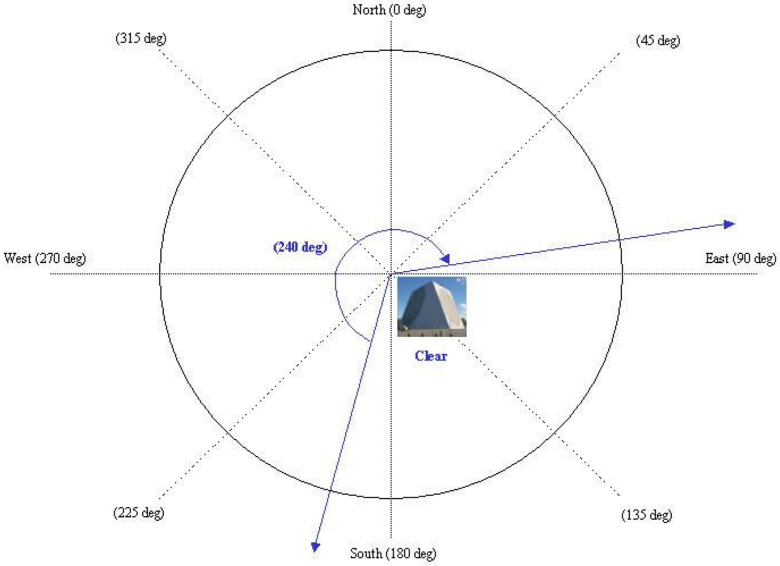


Figure 3.7: Clear Azimuth Coverage

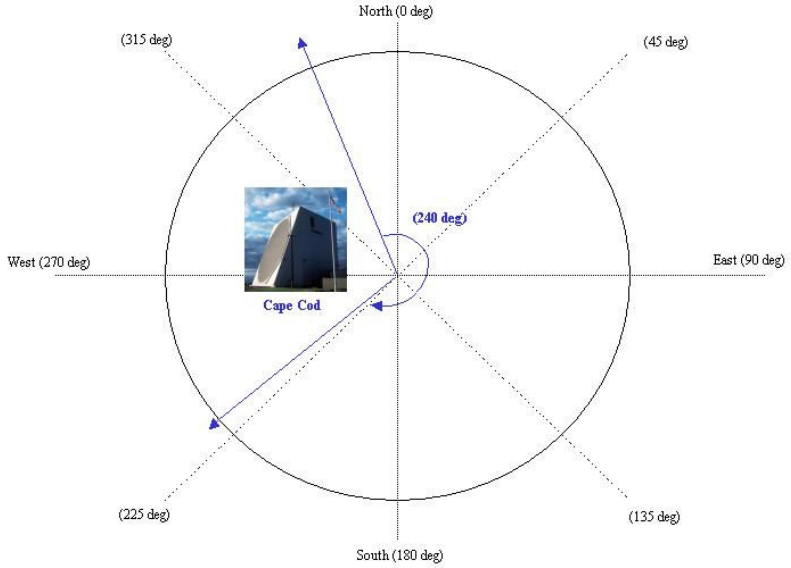


Figure 3.8: Cape Cod Azimuth Coverage

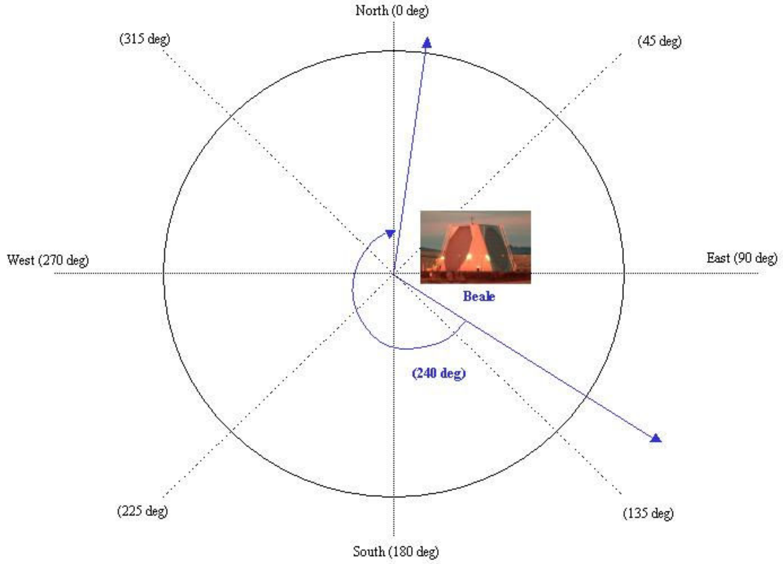


Figure 3.9: Beale Azimuth Coverage

3.5.1 NAVSPASUR Fence

Chapter 8 provides information on the MATLAB code used to perform the observability checks for the NAVSPASUR fence. The radar reflection from a debris object is received by the six NAVSPASUR receiver sites along the entire 30-degree arc of longitude provided in Table 1.3. The azimuth coverage is limited to 90 and 270 degrees only. Figure 3.10 shows the azimuth coverage for the NAVSPASUR Fence.

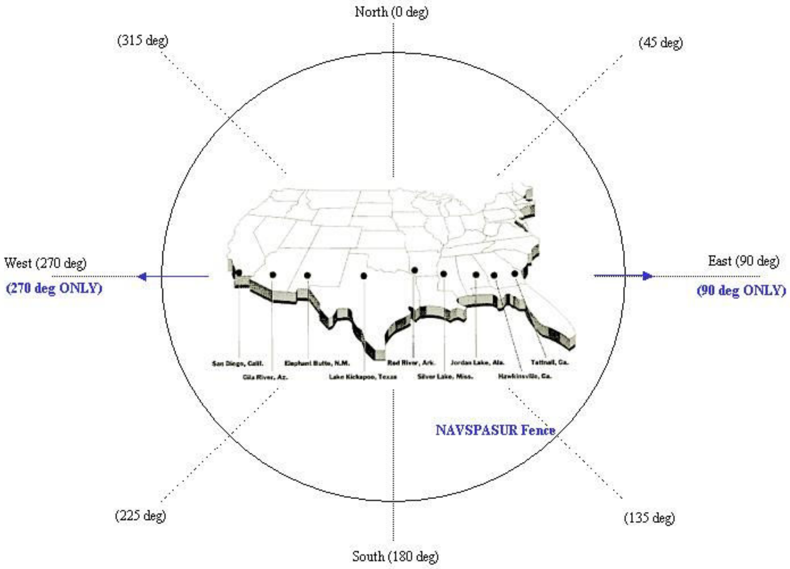


Figure 3.10: NAVSPASUR Fence Azimuth Coverage

CHAPTER 4

ESTIMATION SCHEME

4.1 Estimation Scheme Background

For the orbit determination problem, the Herrick-Gibbs method usually provides the initial estimate for the position and velocity. This helps achieve convergence for the orbit determination techniques.[18] A guess can also be used as an estimate for the initial position and velocity. An initial guess, chosen as the true orbit position and velocity, eliminates the variability of the initial guess from the analysis of the accuracy and distribution studies of the orbit determination algorithms. In addition, all of the cataloged orbits have a known starting point that can be obtained from NASA. The MATLAB algorithm created for the orbit determination problem can evaluate the orbit of the debris object using the initial estimate from either the Herrick-Gibbs method or a true orbit position and velocity guess. Chapter 8 discusses the MATLAB algorithms developed.

4.2 Debris Object Position

The Herrick-Gibbs method uses three position and time observations to obtain the second observation velocity vector. Refer to Figure 2.3 for important parameters of the orbit determination problem. The three observation position vectors of a debris object need to be evaluated from the sensor site range, azimuth, and elevation observations. To obtain range, azimuth, and elevation measurements the non-rotating inertial components of slant range vector, $\bar{\rho}$, first need to be obtained.[35]

$$\vec{\rho} = \begin{bmatrix} x - R\cos\phi\cos\theta \\ y - R\cos\phi\sin\theta \\ z - R\sin\phi \end{bmatrix} \tag{4.1}$$

Where (x, y, z) are the ECI components of vector $\vec{r}$. Then $\vec{\rho}$ needs to be converted from the inertial to the observer coordinate system (topocentric) using the elementary rotations provided in chapter 1 section 1.8.2 (equations 1.5-1.6). [18] Equation (4.2) gives the solution for the topocentric coordinates (ρ_s, ρ_e, ρ_u). Equation (4.2) expansion is provided in appendix B.

$$\begin{bmatrix} \rho_s \\ \rho_e \\ \rho_u \end{bmatrix} = C_2\left(90^0 - \phi\right)C_3(\theta)\vec{\rho} = \begin{bmatrix} \cos\theta\sin\phi & \sin\theta\sin\phi & -\cos\phi \\ -\sin\theta & \cos\theta & 0 \\ \cos\theta\cos\phi & \sin\theta\cos\phi & \sin\phi \end{bmatrix} \vec{\rho} \tag{4.2}$$

Finally, the range (ρ), azimuth (az), and elevation (el) are given by the following equations: [33]

$$\rho = \sqrt{\rho_u^2 + \rho_e^2 + \rho_s^2} \tag{4.3}$$

$$az = \tan^{-1}\left(-\frac{\rho_e}{\rho_s}\right) \tag{4.4}$$

$$el = \sin^{-1}\left(\frac{\rho_u}{\rho}\right) \tag{4.5}$$

The minus sign in the azimuth equation is necessary, since azimuth is measured clockwise from North instead of counter-clockwise from South.[33] Counter-clockwise from South is standard for a right-handed orthogonal coordinate system.[33] To evaluate the azimuth angle properly, the proper quadrant must be selected for the inverse tangent (arctangent).[33] The four quadrant inverse tangent in MATLAB is evaluated for the azimuth angle. This function gives real part azimuth values from interval $\left[-\pi,\pi\right]$ instead of interval $\left[-\frac{\pi}{2},\frac{\pi}{2}\right]$. The sensor site position vector and the slant range vector can be evaluated from the range, azimuth, elevation, latitude, and LST data using equations (4.6) and (4.7). Equation (4.6) is provided in reference [35]. Equation (4.7) is provided in reference [35] and adapted using reference [18].

$$\vec{R}_K = R\begin{bmatrix}\cos\phi\cos\theta_K \\ \cos\phi\sin\theta_K \\ \sin\phi\end{bmatrix} \quad k=1,2,3 \tag{4.6}$$

$$\left(\vec{\rho}_{seu}\right)_K = \begin{bmatrix}\rho_s \\ \rho_e \\ \rho_u\end{bmatrix} = \rho_k\begin{bmatrix}-\cos el_k\cos az_k \\ \cos el_k\sin az_k \\ \sin el_k\end{bmatrix} \quad k=1,2,3 \tag{4.7}$$

$k=1,2,3$ are subscripts describing the three observations times. The debris object position vector can then be obtained using equation (4.8) that is provided in reference [35] and adapted with reference [18]. Equation (4.8) expansion is provided in appendix B.

$$\begin{aligned}\vec{r}_k &= C_3\left(-\theta_k\right)C_2\left(-\left(90^0-\phi_k\right)\right)\left(\vec{\rho}_{seu}\right)_K + \vec{R}_k,\; k=1,2,3 \\ &= \begin{bmatrix}\sin\phi_k\cos\theta_k & -\sin\theta_k & \cos\phi_k\cos\theta_k \\ \sin\phi_k\sin\theta_k & \cos\theta_k & \cos\phi_k\sin\theta_k \\ -\cos\phi_k & 0 & \sin\phi_k\end{bmatrix}\left(\vec{\rho}_{seu}\right)_K + \vec{R}_k\end{aligned} \tag{4.8}$$

4.3 Herrick-Gibbs Middle Velocity Vector

The velocity at the second observation, $\vec{v}_2$, can be determined from the three position vectors and the corresponding time observations.[35] Equation (4.9) gives the variables used for the Taylor series expansion for the position derivative.

$$
\begin{aligned}
& g_1 = \frac{JD_{23}}{JD_{12}JD_{13}},\quad g_3 = \frac{JD_{12}}{JD_{23}JD_{13}},\quad g_2 = g_1 - g_3 \\
& h_1 = \frac{\mu(JD_{23})}{12},\quad h_3 = \frac{\mu(JD_{12})}{12},\quad h_2 = h_1 - h_3 \\
& d_k = g_k + \frac{h_k}{\|\vec{r}_k\|^3},\quad k = 1,2,3
\end{aligned}
\tag{4.9}
$$

Where,

$$JD_{ij} = JD_j - JD_i \tag{4.10}$$

Then the middle velocity vector then becomes:

$$\vec{v}_2 = -d_1\vec{r}_1 + d_2\vec{r}_2 + d_3\vec{r}_3 \tag{4.11}$$

4.3.1 Herrick-Gibbs Method Validation

Validation of the Herrick-Gibbs method is based on example (7-4) obtained from reference [18]. The MATLAB code used to solve this problem is explained in Chapter 8. Using the following three position vectors (km):

$$
\begin{aligned}
\vec{r}_1 &= 3419.85564\vec{i} + 6019.82602\vec{j} + 2784.60022\vec{k} \\
\vec{r}_2 &= 2935.91195\vec{i} + 6326.18324\vec{j} + 2660.59584\vec{k} \\
\vec{r}_1 &= 2434.95202\vec{i} + 6597.38674\vec{j} + 2521.52311\vec{k}
\end{aligned}
$$

Using the following time observations:

$$
\begin{aligned}
t_1 &= 0.0s \\
t_2 &= 1\min 16.48s \\
t_3 &= 2\min 33.04s
\end{aligned}
$$

The Herrick-Gibbs Method example solution:

$$\vec{v}_2 = -6.441645\vec{i} + 3.7776343\vec{j} - 1.720587\vec{k}$$

The Herrick-Gibbs Method using a MATLAB code described in chapter 8 gave the following solution:

$$\vec{v}_2 = -6.441557\vec{i} + 3.7775596\vec{j} - 1.720568\vec{k}$$

CHAPTER 5

MEASUREMENT GENERATION

5.1 Measurement Generation Background

A fixed step fourth-order Runge-Kutta (RK4) is used to integrate the true orbit using the dynamic models provided in chapter 2 from a given epoch through a chosen propagation time (10 hours). The integration of the orbit proceeds in interval steps of 1 second. A sensor site measures the range, azimuth, and elevation to a debris object in orbit. True orbit data is checked for observability for each sensor site. The observable true orbit data is then corrupted and added to the range, azimuth, and elevation calculations every second. The measurement errors are zero-mean Gaussian with a standard deviation of the range measurement error as 1 km, and a standard deviation of the angle measurements (azimuth, elevation) as 0.01 degrees.[35] Chapter 4 provides the necessary equations that are required to obtain the measurements. The measurement errors were chosen using data provided in reference [35] (Example 4.3). Chapter 8 provides information on the algorithms created to generate the measurements.

5.2 Measurement Generation

The measurements were obtained for the entire observable orbit (every second) for each sensor site. This makes it easier to alter the amount of observations that the estimator will use for the simulation. The measurements are stored in a text file. The measurements are then read for the estimator using a specified increment of time. This will generate a specified batch of measurements. Reference [4] evaluates measurements that were generated by the Eglin sensor. For the debris object used in reference [4], the Eglin sensor generated 11 total 4 minute long tracks with one orbital pass of the debris object over the sensor site. It was assumed that there were 25 observations per track that generated a total of 275 observations from the Eglin sensor.[4] This results in approximately a measurement every 10 seconds over the sensor site. This is the assumption that is used for the generation of measurements in this research. Two sets of observation batches are used. One batch contains 24 measurements and the other contains 240 measurements. The batch of 240 measurements is considered to be a dense observation batch. Using different batch of measurements allows for a comparison to be made between the effectiveness of the estimator and the amount of observations used. The orbit is then propagated after every 4 minutes of measurements. The length of the observation time over the sensor site is different from reference [4] as a different orbit is chosen.

CHAPTER 6

ESTIMATION PROCESS

6.1 Nonlinear Least Squares Method

Nonlinear least squares estimation (Gaussian least squares differential correction) is used to determine the orbit of the debris object from range, azimuth, and elevation observations. A batch least squares estimation approach is used to obtain the estimates.[35] The implementation of the nonlinear least squares method for the orbital debris tracking algorithm is explained in chapter 7. Figure 6.1 provides a flow chart of the algorithm used for this research. Figure 6.1 was adapted from reference [35]. k refers to the current measurement time and i refers to the current iteration. The observation section time is denoted as $t_k = \begin{bmatrix} t_1 & t_2 & \cdots & t_m \end{bmatrix}$. Measurements are used from the observation section to make a correction to the estimates during each iteration.[35]

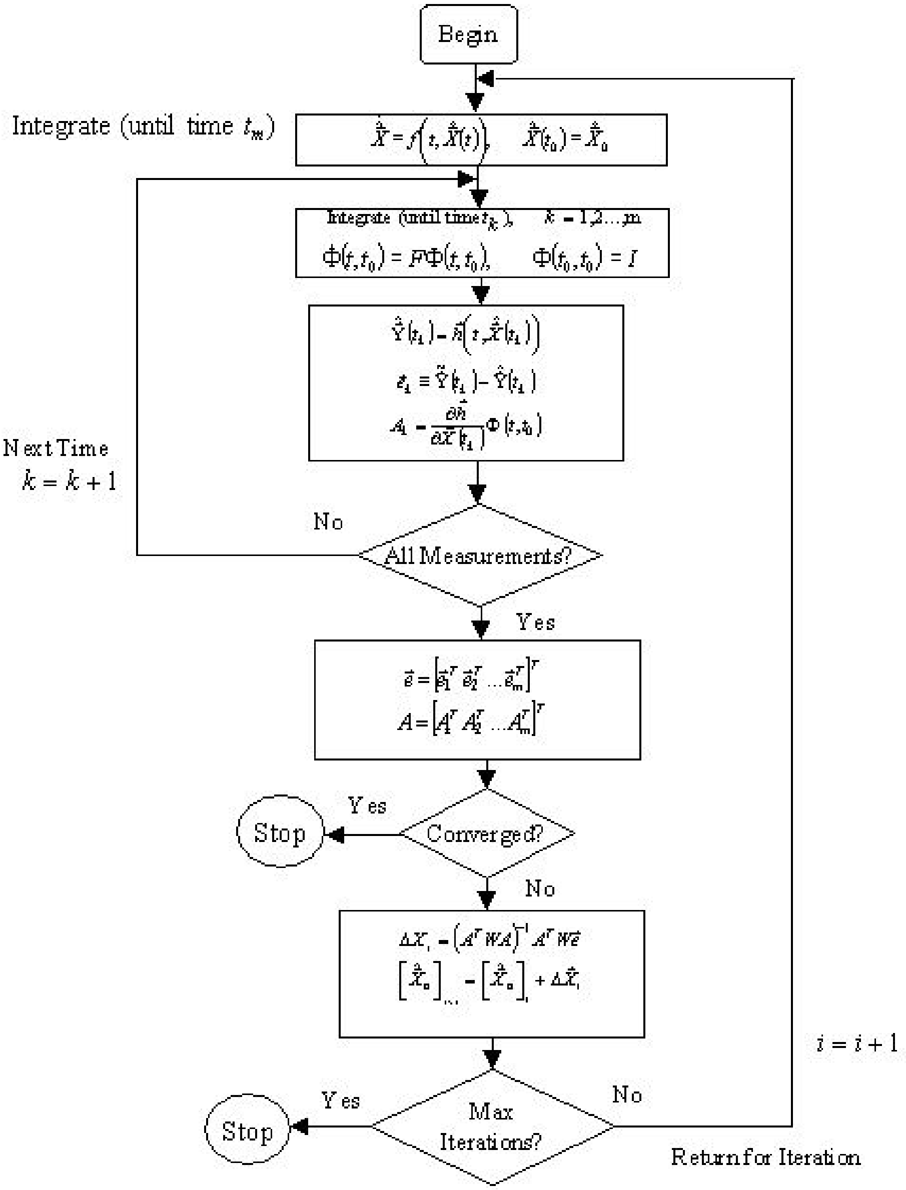

Begin
Integrate (until time t_m)
$\dot{\hat{X}} = f\left(t, \hat{X}(t)\right), \quad \hat{X}(t_0) = \hat{X}_0$
Integrate (until time t_k), $k = 1,2 \ldots m$
$\Phi(t, t_0) = F\Phi(t, t_0), \quad \Phi(t_0, t_0) = I$
$\hat{Y}(t_i) = \vec{h}\left(t, \hat{X}(t_i)\right)$
$\vec{e}_i \equiv \tilde{Y}(t_i) - \hat{Y}(t_i)$
$A_i = \frac{\partial \vec{h}}{\partial \vec{X}(t_i)} \Phi(t, t_0)$
Next Time
$k = k + 1$
No
All Measurements?
Yes
$\vec{e} = \left[\vec{e}_1^T \, \vec{e}_2^T \ldots \vec{e}_m^T\right]^T$
$A = \left[A_1^T \, A_2^T \ldots A_m^T\right]^T$
Yes
Stop
Converged?
No
$\Delta X_i = \left(A^T W A\right)^{-1} A^T W \vec{e}$
$\left[\hat{X}_0\right]_{i+1} = \left[\hat{X}_0\right]_i + \Delta \hat{X}_i$
$i = i + 1$
Yes
Stop
Max Iterations?
No
Return for Iteration

Figure 6.1: Least Squares Orbit Determination

The state, $\vec{X} = \left[\vec{r}^T \; \vec{v}^T\right]^T$, is the position and velocity of the debris object. The initial condition for the position and velocity is given as $\vec{X}_0 = \left[\vec{r}_0{}^T \; \vec{v}_0{}^T\right]^T$. The initial state is taken as the true position and velocity at the start of the observation section. The equations of motion are the derivatives of the state given as $\dot{\vec{X}} = [\bar{\mathrm{v}} \quad \vec{\mathrm{a}}]^T$. The measurement equation is given by equation (4.3-4.5) with $\bar{y} = [\rho\, \mathrm{az\, el}]^T$. The measurements are range, azimuth, and elevation.

6.1.1 State Transition Matrix

The state transition matrix is given by equation (6.1).[35] The state transition matrix moves the state through time.[18]

$$\Phi(t, t_0) \equiv \frac{\partial \vec{X}(t)}{\partial \vec{X}_0} \tag{6.1}$$

The state transition matrix is obtained using the discrete phi, Φ, method. The state matrix, F, is converted from the continuous-time to discrete-time. The state matrix is further described in section 6.1.2. The following equation is used to propagate Φ:[35]

$$\dot{\Phi}(t, t_0) = F\Phi(t, t_0), \quad \Phi(t_0, t_0) = I_{6x6} \tag{6.2}$$

The initial value of Φ is a 6 X 6 identity matrix. The discrete phi method is a faster technique compared to Battin's method.[35] Based on the results in example 4.3 in reference [35], the discrete phi method provides comparable results to Battin's method for state transition matrix propagation. This is validated using the code provided with the example.

6.1.2 Dynamic Model without Perturbations

The dynamic model with no perturbations is given by equation (2.1). The state matrix, F, is the partial derivatives (Jacobian matrix) of the equations of motion due to the state. The state matrix is given by equations (6.3-6.7). Derivations are shown in Appendix B.

$$F \equiv \frac{\partial \dot{\vec{X}}}{\partial \vec{X}} = \begin{bmatrix} \frac{\partial \vec{v}}{\partial \vec{r}} & \frac{\partial \vec{v}}{\partial \vec{v}} \\ \frac{\partial \vec{a}}{\partial \vec{r}} & \frac{\partial \vec{a}}{\partial \vec{v}} \end{bmatrix} \tag{6.3}$$

Where,

$$\frac{\partial \vec{v}}{\partial \vec{r}} = 0_{3x3} \tag{6.4}$$

$$\frac{\partial \vec{v}}{\partial \vec{v}} = I_{3x3} \tag{6.5}$$

$$\frac{\partial \vec{a}}{\partial \vec{r}} = \begin{bmatrix} \frac{3\mu x^2}{r^5} - \frac{\mu}{r^3} & \frac{3\mu xy}{r^5} & \frac{3\mu xz}{r^5} \\ \frac{3\mu xy}{r^5} & \frac{3\mu y^2}{r^5} - \frac{\mu}{r^3} & \frac{3\mu yz}{r^5} \\ \frac{3\mu xz}{r^5} & \frac{3\mu yz}{r^5} & \frac{3\mu z^2}{r^5} - \frac{\mu}{r^3} \end{bmatrix} \tag{6.6}$$

$$\frac{\partial \vec{a}}{\partial \vec{v}} = 0_{3x3} \tag{6.7}$$

6.1.3 Dynamic Model with Perturbations

The dynamic model with no perturbations is given by equation (2.3). The ECI perturbation components are given in equation (2.4). Equations (6.8 – 6.11) show the components of the state matrix:

$$F \equiv \frac{\partial \dot{\vec{X}}}{\partial \vec{X}} = \begin{bmatrix} \frac{\partial \vec{v}}{\partial \vec{r}} & \frac{\partial \vec{v}}{\partial \vec{v}} \\ \frac{\partial \vec{a}}{\partial \vec{r}} & \frac{\partial \vec{a}}{\partial \vec{v}} \end{bmatrix} \tag{6.8}$$

where,

$$\frac{\partial \bar{v}}{\partial \bar{r}} = 0_{3x3} \tag{6.9}$$

$$\frac{\partial \bar{v}}{\partial \bar{v}} = I_{3x3} \tag{6.10}$$

$$\frac{\partial \bar{a}}{\partial \bar{r}} = \begin{bmatrix} \left(\frac{3\mu x^2}{r^5} - \frac{\mu}{r^3}\right) + \frac{\partial a_{dx}}{\partial x} + \frac{\partial a_{gx}}{\partial x} & \left(\frac{3\mu xy}{r^5}\right) + \frac{\partial a_{dx}}{\partial y} + \frac{\partial a_{gx}}{\partial y} & \left(\frac{3\mu xz}{r^5}\right) + \frac{\partial a_{dx}}{\partial z} + \frac{\partial a_{gx}}{\partial z} \\ \left(\frac{3\mu xy}{r^5}\right) + \frac{\partial a_{dy}}{\partial x} + \frac{\partial a_{gy}}{\partial x} & \left(\frac{3\mu y^2}{r^5} - \frac{\mu}{r^3}\right) + \frac{\partial a_{dy}}{\partial y} + \frac{\partial a_{gy}}{\partial y} & \left(\frac{3\mu yz}{r^5}\right) + \frac{\partial a_{dy}}{\partial z} + \frac{\partial a_{gy}}{\partial z} \\ \left(\frac{3\mu xz}{r^5}\right) + \frac{\partial a_{dz}}{\partial x} + \frac{\partial a_{gz}}{\partial x} & \left(\frac{3\mu yz}{r^5}\right) + \frac{\partial a_{dz}}{\partial y} + \frac{\partial a_{gz}}{\partial y} & \left(\frac{3\mu z^2}{r^5} - \frac{\mu}{r^3}\right) + \frac{\partial a_{dz}}{\partial z} + \frac{\partial a_{gz}}{\partial z} \end{bmatrix} \tag{6.11}$$

The partial derivatives $\left(\frac{\partial \vec{a}_d}{\partial \bar{r}}, \frac{\partial \vec{a}_d}{\partial \bar{v}}\right)$ for the dynamic model with atmospheric drag perturbations are provided in Appendix A. The derivations are shown in Appendix B. The partial derivatives of the acceleration due to gravity with respect to the cartesian state components (x, y, z) are obtained by differentiating equation (2.9).[37] This leads to equation (6.12).[37] It is important to note that the acceleration due to gravity is not a function of velocity.

$$\begin{aligned} \frac{\partial \vec{a}_g}{\partial \bar{r}} = & \frac{\partial}{\partial \bar{r}}\left(\frac{\partial U}{\partial r}\right)\frac{\partial r}{\partial \bar{r}} + \frac{\partial}{\partial \bar{r}}\left(\frac{\partial U}{\partial \delta}\right)\frac{\partial \delta}{\partial \bar{r}} + \frac{\partial}{\partial \bar{r}}\left(\frac{\partial U}{\partial \gamma}\right)\frac{\partial \gamma}{\partial \bar{r}} \\ & + \frac{\partial U}{\partial r}\frac{\partial^2 r}{\partial \bar{r}^2} + \frac{\partial U}{\partial \delta}\frac{\partial^2 \delta}{\partial \bar{r}^2} + \frac{\partial U}{\partial \gamma}\frac{\partial^2 \gamma}{\partial \bar{r}^2} \end{aligned} \tag{6.12}$$

In matrix notation, the first set of partial derivatives for equation (6.12) is given by equation (6.13).[37]

$$\frac{\partial}{\partial \vec{r}}\begin{bmatrix} \frac{\partial U}{\partial r} \\ \frac{\partial U}{\partial \delta} \\ \frac{\partial U}{\partial \gamma} \end{bmatrix} = \begin{bmatrix} \frac{\partial^2 U}{\partial r^2} & \frac{\partial^2 U}{\partial r \partial \delta} & \frac{\partial^2 U}{\partial r \partial \gamma} \\ \frac{\partial^2 U}{\partial \delta \partial r} & \frac{\partial^2 U}{\partial \delta^2} & \frac{\partial^2 U}{\partial \delta \partial \gamma} \\ \frac{\partial^2 U}{\partial \gamma \partial r} & \frac{\partial^2 U}{\partial \gamma \partial \delta} & \frac{\partial^2 U}{\partial \gamma^2} \end{bmatrix} \begin{bmatrix} \frac{\partial r}{\partial \vec{r}} \\ \frac{\partial \delta}{\partial \vec{r}} \\ \frac{\partial \gamma}{\partial \vec{r}} \end{bmatrix} \tag{6.13}$$

The partial derivatives given by equations (2.10) are differentiated in spherical coordinates with respect to (r, δ, γ) as shown by the following:[37]

$$\frac{\partial^2 U}{\partial r^2} = \frac{\mu}{r^3} \sum_{n=2}^{\infty} \left(\frac{R}{r}\right)^n (n+2)(n+1) \sum_{m=0}^{n} \left(C_n^m \cos m\gamma + S_n^m \sin m\gamma\right) P_n^m (\sin\delta)$$

$$\frac{\partial^2 U}{\partial r \partial \delta} = \frac{\partial^2 U}{\partial \delta \partial r} = -\frac{\mu}{r^2} \sum_{n=2}^{\infty} \left(\frac{R}{r}\right)^n (n+1) \sum_{m=0}^{n} \left(C_n^m \cos m\gamma + S_n^m \sin m\gamma\right) \left[P_n^{m+1}(\sin\delta) - m \tan\delta \, P_n^m (\sin\delta)\right]$$

$$\frac{\partial^2 U}{\partial r \partial \gamma} = \frac{\partial^2 U}{\partial \gamma \partial r} = -\frac{\mu}{r^3} \sum_{n=2}^{\infty} \left(\frac{R}{r}\right)^n (n+1) \sum_{m=0}^{n} m P_n^m (\sin\delta) \left(S_n^m \cos m\gamma - C_n^m \sin m\gamma\right)$$

$$\frac{\partial^2 U}{\partial \delta^2} = \frac{\mu}{r} \sum_{n=2}^{\infty} \left(\frac{R}{r}\right)^n \sum_{m=0}^{n} \left(C_n^m \cos m\gamma + S_n^m \sin m\gamma\right) \left\{\tan\delta \, P_n^{m+1}(\sin\delta) + \left[m^2 \sec^2\delta - m \tan^2\delta - n(n+1)\right] P_n^m (\sin\delta)\right\}$$

$$\frac{\partial^2 U}{\partial \delta \partial \gamma} = \frac{\partial^2 U}{\partial \gamma \partial \delta} = \frac{\mu}{r} \sum_{n=2}^{\infty} \left(\frac{R}{r}\right)^n \sum_{m=0}^{n} m \left(S_n^m \cos m\gamma - C_n^m \sin m\gamma\right) \left[P_n^{m+1}(\sin\delta) - m \tan\delta \, P_n^m (\sin\delta)\right]$$

$$\frac{\partial^2 U}{\partial \gamma^2} = -\frac{\mu}{r} \sum_{n=2}^{\infty} \left(\frac{R}{r}\right)^n \sum_{m=0}^{n} m^2 P_n^m (\sin\delta) \left(C_n^m \cos m\gamma + S_n^m \sin m\gamma\right)$$

Differentiating equations (2.11) with respect to (x, y, z) gives the remaining partial derivatives of (6.12) as shown by the following equations:[18]

$$\frac{\partial^2 r}{\partial \bar{r}^2} = \frac{1}{r}\left[I - \frac{\bar{r}\bar{r}^T}{r^2}\right]$$

$$\frac{\partial^2 \delta}{\partial \bar{r}^2} = -\frac{1}{\left(x^2+y^2\right)^{\frac{3}{2}}}\left(\left(\frac{\partial z}{\partial \bar{r}}\right)^T - \frac{z\bar{r}}{r^2}\right)\left[x\left(\frac{\partial x}{\partial \bar{r}}\right) + y\left(\frac{\partial y}{\partial \bar{r}}\right)\right] - \frac{1}{r^2\sqrt{x^2+y^2}}\left[\bar{r}\left(\frac{\partial z}{\partial \bar{r}}\right) + zI - \frac{2z}{r^2}\bar{r}\bar{r}^T\right]$$

$$\frac{\partial^2 \gamma}{\partial \bar{r}^2} = -\frac{2}{\left(x^2+y^2\right)}\begin{bmatrix} -y \\ x \\ 0 \end{bmatrix}\left(x\frac{\partial x}{\partial \bar{r}} + y\frac{\partial y}{\partial \bar{r}}\right) + \frac{1}{\left(x^2+y^2\right)}\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Expansions for these equations are provided in Appendix B. Putting the partials back into equation (6.12) gives the Jacobian matrix for the acceration due to gravity as shown in equation (6.14). The matrix given by equation (6.14) is not derived due to the amount of terms that are required to be expanded. Using MATLAB it is easier to implement the Jacobian matrix for the acceleration due to gravity in steps. Chapter 8 provides further detail on the implementation of the Jacobian.

$$\frac{\partial \bar{a}_g}{\partial \bar{r}} = \begin{bmatrix} \frac{\partial a_{gx}}{\partial x} & \frac{\partial a_{gx}}{\partial y} & \frac{\partial a_{gx}}{\partial z} \\ \frac{\partial a_{gy}}{\partial x} & \frac{\partial a_{gy}}{\partial y} & \frac{\partial a_{gy}}{\partial z} \\ \frac{\partial a_{gz}}{\partial x} & \frac{\partial a_{gz}}{\partial y} & \frac{\partial a_{gz}}{\partial z} \end{bmatrix} \tag{6.14}$$

The acceleration due to atmospheric drag is a function of velocity. Therefore, the Jacobian is given by the following matrix:

$$\frac{\partial \bar{a}}{\partial \bar{v}} = \begin{bmatrix} \frac{\partial a_{dx}}{\partial \dot{x}} & \frac{\partial a_{dx}}{\partial \dot{y}} & \frac{\partial a_{dx}}{\partial \dot{z}} \\ \frac{\partial a_{dy}}{\partial \dot{x}} & \frac{\partial a_{dy}}{\partial \dot{y}} & \frac{\partial a_{dy}}{\partial \dot{z}} \\ \frac{\partial a_{dz}}{\partial \dot{x}} & \frac{\partial a_{dz}}{\partial \dot{y}} & \frac{\partial a_{dz}}{\partial \dot{z}} \end{bmatrix} \tag{6.15}$$

The expansions and derivations were verified by evaluating MATLAB functions using symbolic variables. These functions are described in chapter 8.

6.1.4 Sensitivity Matrix

The matrix H is referred to as the sensitivity matrix or the observation partials matrix.[35] H indicates how the observations are affected by changes in the state.[35] The observations are affected by position only. The sensitivity matrix is computed using the following equations: [35]

$$H = \frac{\partial \vec{h}}{\partial \vec{X}} = \begin{bmatrix} H_{11} & 0_{3x3} \end{bmatrix} \tag{6.16}$$

$$H_{11} = \begin{bmatrix} \frac{\partial \rho}{\partial x} & \frac{\partial \rho}{\partial y} & \frac{\partial \rho}{\partial z} \\ \frac{\partial az}{\partial x} & \frac{\partial az}{\partial y} & \frac{\partial az}{\partial z} \\ \frac{\partial el}{\partial x} & \frac{\partial el}{\partial y} & \frac{\partial el}{\partial z} \end{bmatrix} \tag{6.17}$$

Equation (6.18-6.26) derivations are provided in Appendix B.

$$\frac{\partial \rho}{\partial x} = \left(\rho_u \cos\phi \cos\theta - \rho_e \sin\theta + \rho_s \sin\phi \cos\theta\right) / \rho \tag{6.18}$$

$$\frac{\partial \rho}{\partial y} = \left(\rho_u \cos\phi \sin\theta + \rho_e \cos\theta + \rho_s \sin\phi \sin\theta\right) / \rho \tag{6.19}$$

$$\frac{\partial \rho}{\partial z} = \left(\rho_u \sin\phi - \rho_s \cos\phi\right) / \rho \tag{6.20}$$

$$\frac{\partial az}{\partial x} = \frac{1}{\left(\rho_s^2 + \rho_e^2\right)} \left(\rho_e \sin\phi \cos\theta + \rho_s \sin\theta\right) \tag{6.21}$$

$$\frac{\partial az}{\partial y} = \frac{1}{\left(\rho_s^2 + \rho_e^2\right)} \left(\rho_e \sin\phi \sin\theta - \rho_s \cos\theta\right) \tag{6.22}$$

$$\frac{\partial az}{\partial z} = -\frac{1}{\left(\rho_s^2 + \rho_e^2\right)} \rho_e \cos\phi \tag{6.23}$$

$$\frac{\partial el}{\partial x} = \frac{1}{\rho\left(\rho^2 - \rho_u^2\right)^{1/2}} \left(\rho \cos\phi \cos\theta - \rho_u \frac{\partial \rho}{\partial x} \right) \tag{6.24}$$

$$\frac{\partial el}{\partial y} = \frac{1}{\rho\left(\rho^2 - \rho_u^2\right)^{1/2}} \left(\rho \cos\phi \sin\theta - \rho_u \frac{\partial \rho}{\partial y} \right) \tag{6.25}$$

$$\frac{\partial el}{\partial z} = \frac{1}{\rho\left(\rho^2 - \rho_u^2\right)^{1/2}} \left(\rho \sin\phi - \rho_u \frac{\partial \rho}{\partial z} \right) \tag{6.26}$$

6.1.5 State Covariance Propagation

The state covariance matrix P, is given by the following equation:[35]

$$P = \left(A^T W A\right)^{-1} \tag{6.27}$$

where,

$$A = H\Phi\left(t, t_0\right) \tag{6.28}$$

The A matrix is stored as $A = \left[A_1^T \quad A_2^T \quad \cdots \quad A_m^T\right]^T$, where m is the number of measurements used. This covariance matrix contains the estimates for the closeness of the fit with the model dynamics.[35] The covariance matrix is propagated during each iteration in the nonlinear least squares algorithm. 3σ bounds are obtained using the diagonal elements of the estimate error-covariance matrix.[35] The bounds help to predict the performance of the estimators.[35] 3σ bound is a region in the measurement space that defines a high probability of a measurement. 3σ bounds indicate a confidence of 99.8%.[49] Equation (6.29) is used to obtain 3σ bounds.

$$\pm 3\sigma = \pm 3 \cdot [diag(P)]^{\frac{1}{2}} \tag{6.29}$$

The standard deviation in the state estimates is given by:

$$\sigma = [diag(P)]^{\frac{1}{2}} \tag{6.30}$$

6.1.6 Weight Matrix

The weight matrix weighs the relative importance of each measurement.[35] The following is the observation error covariance matrix used to obtain the weight matrix:

$$R_{\text{cov}} = \begin{bmatrix} 1^2 & 0 & 0 \\ 0 & \left(0.01\frac{\pi}{180}\right)^2 & 0 \\ 0 & 0 & \left(0.01\frac{\pi}{180}\right)^2 \end{bmatrix}$$

The variance is the square of the standard deviation of the errors provided in chapter 5. This covariance matrix is used to obtain a minimum variance estimate (optimal estimate). The minimal variance estimator is identical to the least squares estimator if the weight matrix is taken as the inverse of the observation error covariance. Furthermore, the covariance matrix is a diagonal matrix as there is no correlation with measurements. Correlation of measurements is further discussed in section 6.2. For this research the weight matrix, W, is obtained by evaluating the Kronecker product of the Identity matrix (size taken as the number of measurements) and the inverse of the covariance matrix R_{cov}.[35] Equation (6.31) gives the Kronecker product used for obtaining the weight matrix.

$$W = I_{mxm} \otimes R_{\text{cov}}^{-1} \tag{6.31}$$

The Kronecker product of (mxn) matrices C and B is shown in equation (6.32). The Kronecker product of C and B is the $mnxmn$ block matrix.[50] A block matrix is a matrix that has been assembled using smaller matrices.[51] The Kronecker product evaluated in MATLAB is further explained in chapter 8.

$$C \otimes B = \begin{bmatrix} c_{11}B & \dots & c_{1n}B \\ \vdots & \ddots & \vdots \\ c_{n1}B & \cdots & c_{nm}B \end{bmatrix} \tag{6.32}$$

6.1.7 State Estimation

The estimated nonlinear dynamic model for the nonlinear least squares method is given by equation (6.33).[35] The state is integrated using an appropriate integrator in MATLAB.

$$\dot{\hat{\bar{X}}} = \bar{f}\left(t,\hat{\bar{X}}\right) \tag{6.33}$$

The estimated measurement model for the nonlinear least squares method is given by equation (6.34). [35]

$$\hat{\bar{\mathrm{Y}}}(t_k) = \vec{h}\left(t,\hat{\bar{X}}(t_k)\right) \tag{6.34}$$

The state is estimated using the dynamic models provided in chapter 2. The estimation of the dynamic model can be different from the model used to generate the true observation data. This allows for testing of the effect of improper modeling on the solution distribution characteristics. However, improper modeling for this research is not analyzed. The differential correction is updated using equation (6.35).[35] This correction is chosen based on the minimization of the sum square of the residual errors given in equation (6.36). [35] The state estimate is update during the iterations.

$$\Delta\vec{X} = \left(A^T W A\right)^{-1} A^T W \vec{e} \tag{6.35}$$

$$J = \frac{1}{2}\vec{e}^{\,T} W \vec{e} \tag{6.36}$$

The residual (error), $\vec{e}$, is the difference between the true measurements and estimated measurements as shown in equation (6.37).[35]

$$\vec{e} = \tilde{\vec{\mathrm{Y}}}\left(t_k\right) - \hat{\vec{\mathrm{Y}}}\left(t_k\right) \tag{6.37}$$

The $\vec{e}$ matrix will be stored as $\vec{e} = \left[\vec{e}_1^{\,T} \quad \vec{e}_2^{\,T} \quad \cdots \quad \vec{e}_m^{\,T}\right]^T$, where m is the number of measurements used. The estimate of the position and velocity is then updated using equation (6.38).[35]

$$\hat{\vec{X}}_{i+1} = \hat{\vec{X}}_i + \Delta\vec{X}_i \tag{6.38}$$

The convergence is checked using the RMS (root mean square) of the state estimates given by equation (6.39).[18] The RMS is evaluated after each iteration using the current and previous state estimate.

$$RMS = \sqrt{\left(\hat{\vec{X}}_{i+1} - \hat{\vec{X}}_i\right)^2} \tag{6.39}$$

Differential corrections is a noisy process. Therefore, the convergence check is initially chosen as $1e^{-3}$. A very low tolerance of $1e^{-8}$ can cause the iterations to become unstable.[18]

6.2 Extended Kalman Filter Method

Another technique that can be used for orbit estimation is the iterated Kalman filter. A continuous-discrete extended Kalman filter is chosen for this research. This filter processes the data forward with an initial condition guess, and then processes the data backward to epoch.[35] The initial time is the start of the batch measurement section. The backward pass uses the final state estimate from the forward pass for the initial condition. The state error covariance is also reset after each forward or backward pass to ensure that there is no "new" information being propagated with each pass.[35] The state estimate is integrated using the model equations until the next measurement time within the batch section. The implementation of this iterated continuous-discrete extended Kalman filter for the orbital debris tracking algorithm is explained in chapter 7. Figure 6.2 shows the flow chart of the algorithm used for this research. Figure 6.2 was adapted from reference [35].

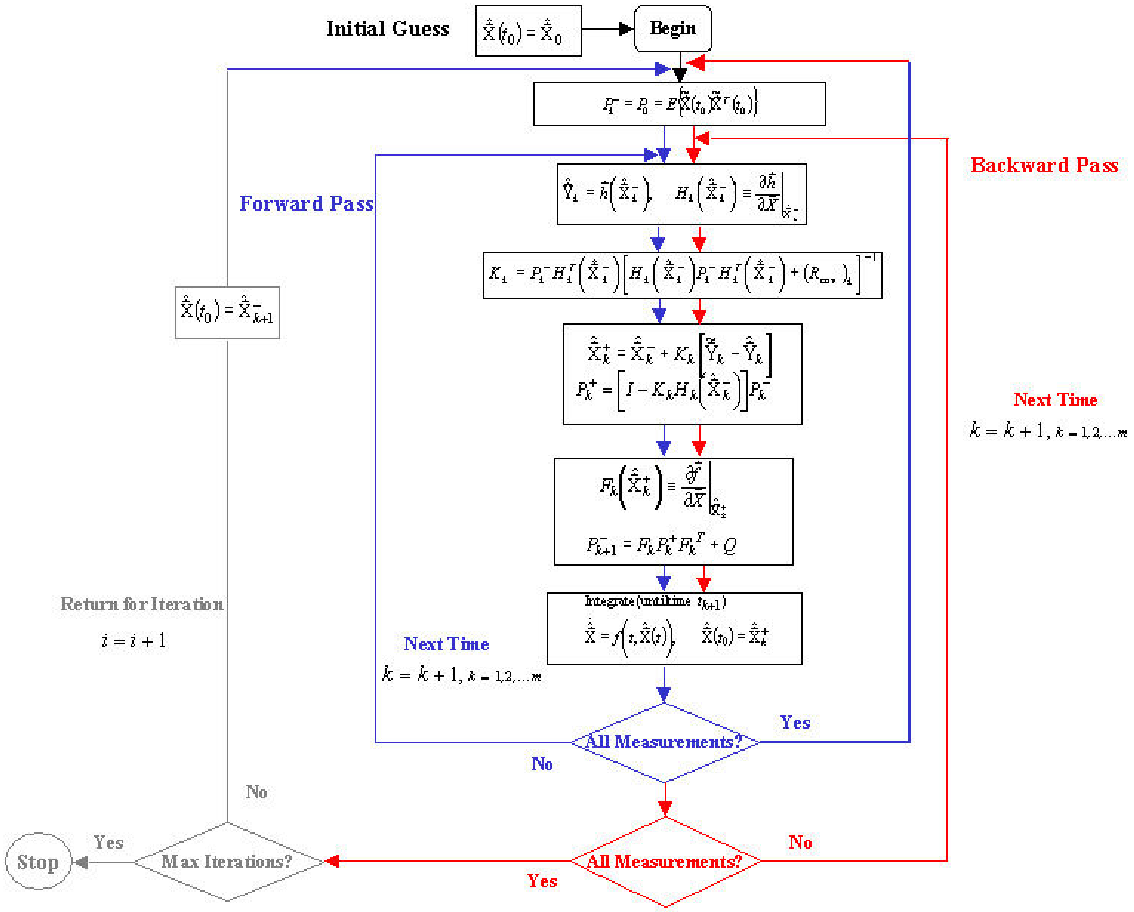


Figure 6.2: Extended Kalman Filter

6.2.1 Dynamic Model

The truth model is given by equations (6.40-6.41).

$$\dot{\vec{X}} = f\left(t, \vec{X}(t)\right) + G(t)\vec{w}(t),\ \ \vec{w}(t) \sim N\left(0, Q(t)\right) \qquad (6.40)$$

$$\tilde{\vec{y}}_k = h\left(\vec{x}_k\right) + \vec{v}_k,\ \ \vec{v}_k \sim N\left(0, \left(R_{\text{cov}}\right)_k\right) \qquad (6.41)$$

$\vec{w}(t)$, ($\vec{w}_k$ for discrete time), is a process noise on the system of differential equations.[35] The process noise is given as a normal distribution of mean 0 and a covariance matrix $Q(t)$. $\vec{v}_k$ is the measurement noise. The measurement noise is given a normal distribution of mean 0 and a covariance matrix $(R_{\text{cov}})_k$. Both $\vec{w}_k$ and $\vec{v}_k$ are assumed to be zero-mean Gaussian white-noise processes. As a result, the errors are not correlated forward or backward in time.[35] The following equations give the expectations of the noise processes:[35]

$$E\left\{\vec{v}_k \vec{v}_k{}^T\right\} = \begin{cases} 0 & k \neq j \\ (R_{\text{cov}})_k & k = j \end{cases} \tag{6.42}$$

$$E\left\{\vec{w}_k \vec{w}_k{}^T\right\} = \begin{cases} 0 & k \neq j \\ Q_k & k = j \end{cases} \tag{6.43}$$

Furthermore, it can be assumed that $\vec{w}_k$ and $\vec{v}_k$ are uncorrelated so that $E\left\{\vec{v}_k \vec{w}_k{}^T\right\} = 0$ for all k. The dynamic models are the same models as those described in section 6.1. In addition, the same state matrices, F, are used as those developed for the dynamic models described in sections (6.1.2-6.1.3).

6.2.2 Initial Conditions

The initial condition for the position and velocity is given as $\vec{X}_0 = \left[\vec{\mathrm{r}}_0{}^{\mathrm{T}}\ \vec{\mathrm{v}}_0{}^{\mathrm{T}}\right]^T$. The initial state is taken as the true position and velocity at the start of the observation batch section. The initial covariance matrix, P_0, is initialized as $P_0 = E\left\{\tilde{\vec{x}}(t_0)\tilde{\vec{x}}^T(t_0)\right\}$. P_0 is chosen as $P_0 = I_{6x6}$. A high value of P indicates high uncertainty of the current state estimate.[52] An initial guess of the true orbit is given; therefore, the state error covariance matrix is chosen conservatively. Q is set to 0. There is no process noise for the reason that a perfect understanding of the model is assumed.[4] The Kalman filter convergence can be affected/"tuned" by choosing an appropriate Q.[52] The Kalman filter for this research is not tuned. This allows for a basic comparison to be made between the two orbit determination techniques.

6.2.3 Sensitivity Matrix

The Kalman sensitivity matrix is evaluated using equation (6.44) directly. The sensitivity matrix uses the same expression provided by equations (6.16-6.17).

$$H_k\left(\hat{\vec{x}}_k^-\right) \equiv \left.\frac{\partial \vec{h}}{\partial \vec{X}}\right|_{\hat{\vec{x}}_k^-} \tag{6.44}$$

6.2.4 Gain

The Kalman gain matrix, K_k, is multiplied by the residual creating an additive correction term to the state estimate equation.[35] K_k is obtained using the following equation:[35]

$$K_k = P_k^- H_k^T\left(\hat{\vec{x}}_k^-\right)\left[H_k\left(\hat{\vec{x}}_k^-\right)P_k^- H_k^T\left(\hat{\vec{x}}_k^-\right) + \left(R_{\text{cov}}\right)_k\right]^{-1} \tag{6.45}$$

6.2.5 Updates

Using measurement estimates, $\hat{\vec{y}}_k = \vec{h}\left(\hat{\vec{x}}_k^-\right)$, the state estimate and the state covariance matrix are update with the following equations:[35]

$$\hat{\vec{x}}_k^+ = \hat{\vec{x}}_k^- + K_k\left[\tilde{\vec{y}}_k - \hat{\vec{y}}_k\right] \tag{6.46}$$

$$P_k^+ = \left[I - K_k H_k\left(\hat{\vec{x}}_k^-\right)\right]P_k^- \tag{6.47}$$

6.2.6 Propagation

The state needs to be propagated further in time until t_{k+1} of the observation section $t_k = \begin{bmatrix} t_1 & t_2 & \cdots & t_m \end{bmatrix}$. The state is integrated using an appropriate integrator in

MATLAB. The state error covariance is also propagated for each step in time discretely.

$$\dot{\hat{x}}(t) = \vec{f}\left(\hat{\vec{x}}(t), t\right) \tag{6.48}$$

$$F\left(\hat{\vec{x}}(t), t\right) \equiv \left.\frac{\partial \vec{f}}{\partial \vec{X}}\right|_{\hat{\vec{x}}(t)} \tag{6.49}$$

The discrete Lyapunov equation is used to propagate the state covariance matrix.[35]

$$P_{k+1}^{-} = F_k P_k^{+} F_k{}^{T} + Q \tag{6.50}$$

CHAPTER 7

DEBRIS OBJECT TRACKING PROBLEM

The debris object has to be tracked for a prolonged period of time using the entire sensor network. The current code is able to track the debris object for a 10-hour propagation time. The code evaluates observation data using one sensor site at a time. Chapter 8 explains the approach used for developing the MATLAB codes. Results for the debris object are for an approximate 6-hour orbit. Furthermore, the results in this chapter are shown for the Eglin sensor site. A complete estimated orbit using the SSN sensors that successfully observed the debris object is also shown in the chapter. The Maui sensor and the NAVSPASUR fence did not track the debris object, as it was never observable by those sensors. The other phased array sensors provide similar results with the tracking algorithm. Those results are not shown. The results for the GEODSS Socorro sensor are provided in appendix C.

7.1 Nonlinear Least Squares Tracking

The orbital debris tracking algorithm follows a batch nonlinear least squares estimation approach to obtain the estimate. Initially, a measurement is generated every 10 seconds during each 240-second (4 minute) observation interval. This batch of measurements is then used to compute the orbital debris position and velocity estimate at the start of the observation section. Each batch contains 24 measurements. Results are also obtained for a batch of 240 measurements. This batch was generated using a measurement taken every second during each 240-second (4 minute) observation interval. The debris object then follows a propagated trajectory of approximately 6 hours using estimated position and velocity data every 240 seconds. Chapter 8 provides information on the MATLAB functions created for this problem.

7.1.1 NLS Eglin Position and Velocity Errors (No Perturbations)

The following results are for the Eglin sensor when the true orbit does not include perturbations. The results are shown for a batch of 24 and 240 measurements. Figures (7.1-7.2) show the observable true orbit and estimated orbit for the Eglin sensor. Figures (7.3-7.6) show the position and velocity errors for the state estimates.

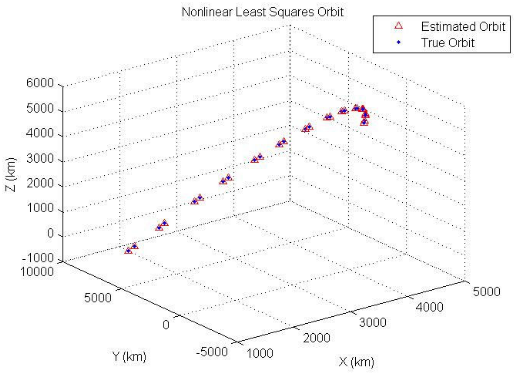


Figure 7.1: NLS Eglin Debris Orbit (No Perturbations Batch 24)

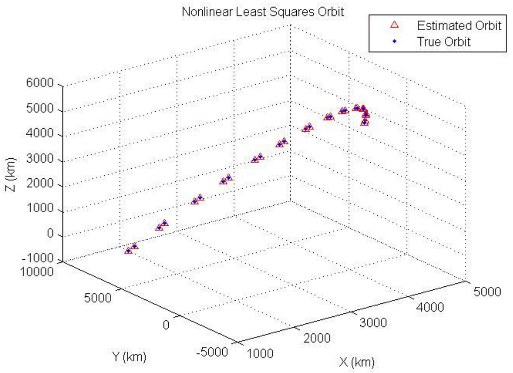


Figure 7.2: NLS Eglin Debris Orbit (No Perturbations Batch 240)

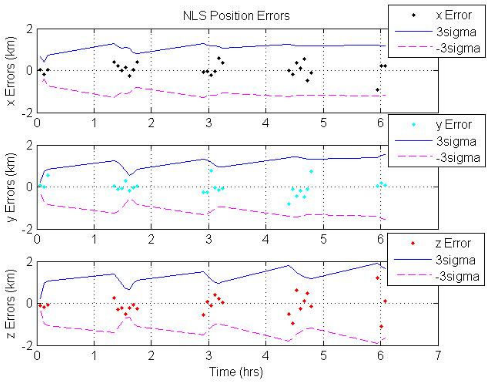


Figure 7.3: NLS Eglin Position Errors
(No Perturbations Batch 24)

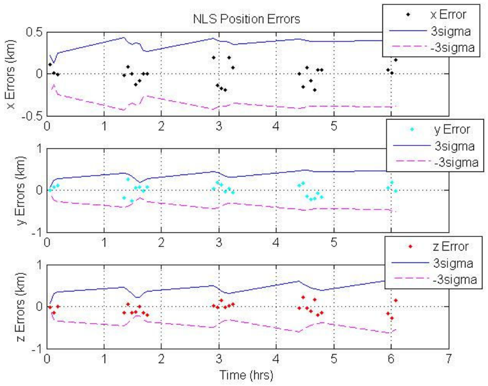


Figure 7.4: NLS Eglin Position Errors
(No Perturbations Batch 240)

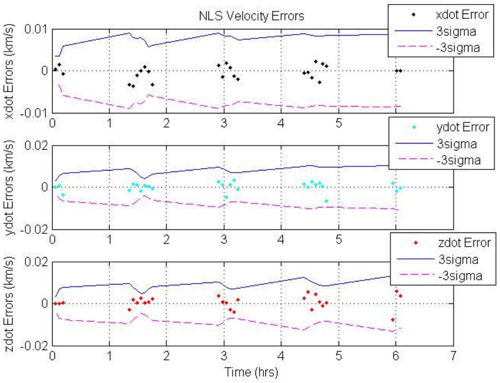


Figure 7.5: NLS Eglin Velocity Errors
(No Perturbations Batch 24)

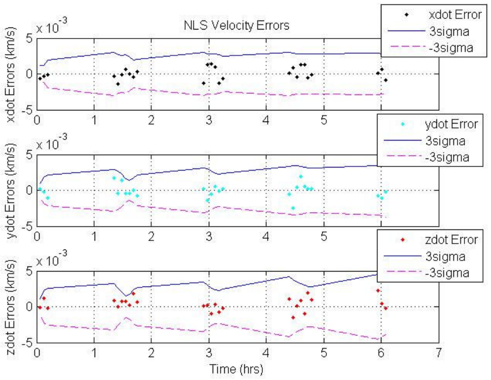


Figure 7.6: NLS Eglin Velocity Errors
(No Perturbations Batch 240)

7.1.2 NLS Eglin Standard Deviations (No Perturbations)

Figures (7.7-7.10) give the position and velocity standard deviations for the Eglin sensor when the true orbit does not include perturbations.

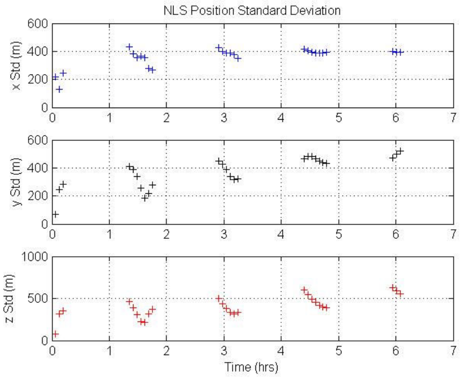


Figure 7.7: NLS Eglin Position Standard Deviation (No Perturbations Batch 24)

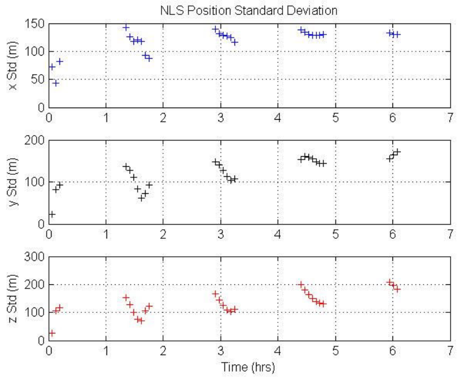


Figure 7.8: NLS Eglin Position Standard Deviation (No Perturbations Batch 240)

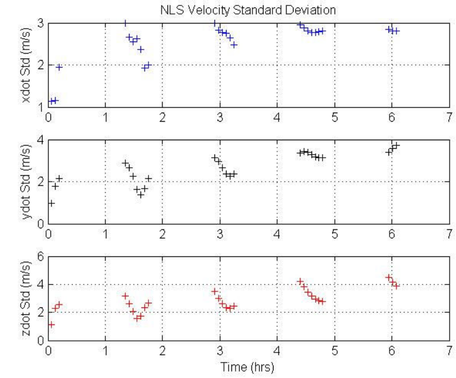


Figure 7.9: NLS Eglin Velocity Standard Deviation (No Perturbations Batch 24)

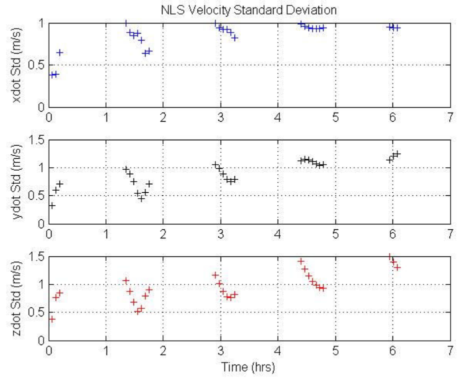


Figure 7.10: NLS Eglin Velocity Standard Deviation (No Perturbations Batch 240)

7.1.3 NLS Eglin Position and Velocity Errors (Perturbations)

The following results are for the Eglin sensor when the orbit has perturbations. Figures (7.11-7.12) show the observable true orbit and estimated orbit for the Eglin sensor. Figures (7.13-7.16) show the errors for the state estimates.

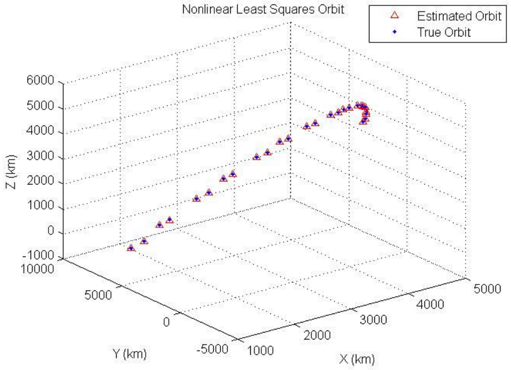


Figure 7.11: NLS Eglin Debris Orbit (Perturbations Batch 24)

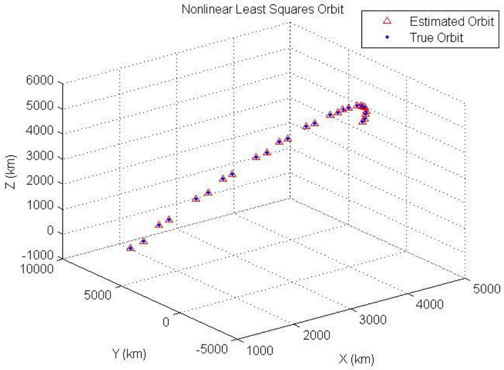


Figure 7.12: NLS Eglin Debris Orbit (Perturbations Batch 240)

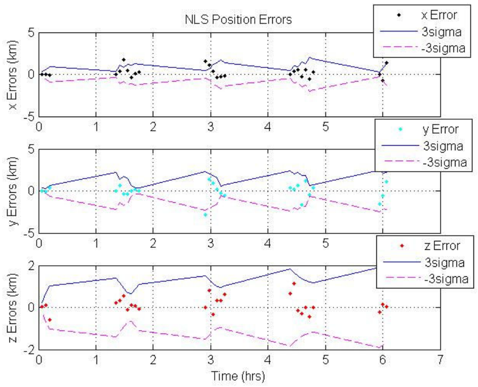


Figure 7.13: NLS Eglin Position Errors (Perturbations Batch 24)

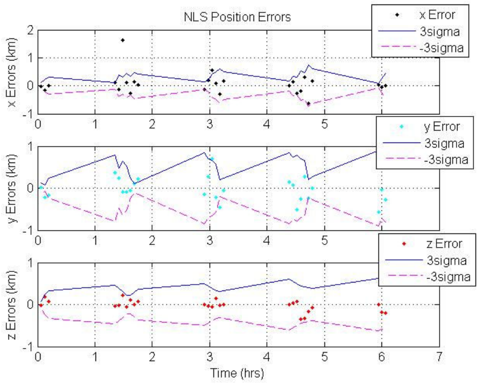


Figure 7.14: NLS Eglin Position Errors (Perturbations Batch 240)

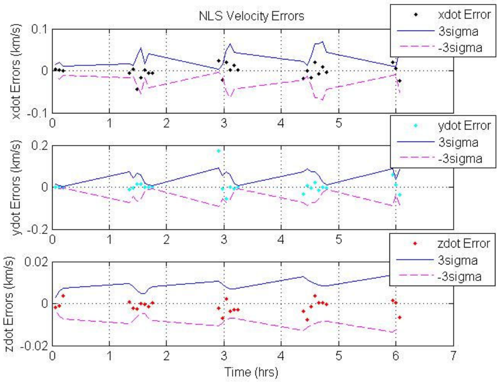


Figure 7.15: NLS Eglin Velocity Errors (Perturbations Batch 24)

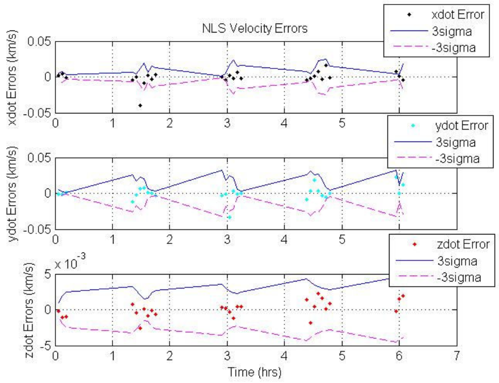


Figure 7.16: NLS Eglin Velocity Errors (Perturbations Batch 240)

7.1.4 NLS Eglin Standard Deviations (Perturbations)

Figures (7.17-7.20) show the position and velocity standard deviations for the Eglin sensor when the true orbit has perturbations.

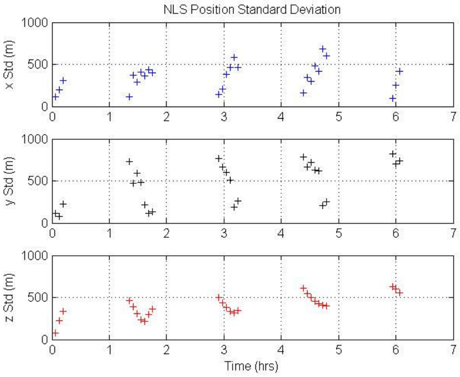


Figure 7.17: NLS Eglin Position Standard Deviation (Perturbations Batch 24)

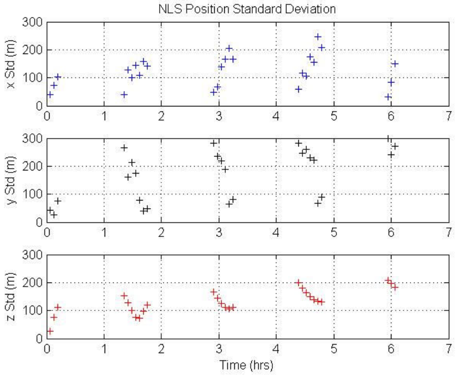


Figure 7.18: NLS Eglin Position Standard Deviation (Perturbations Batch 240)

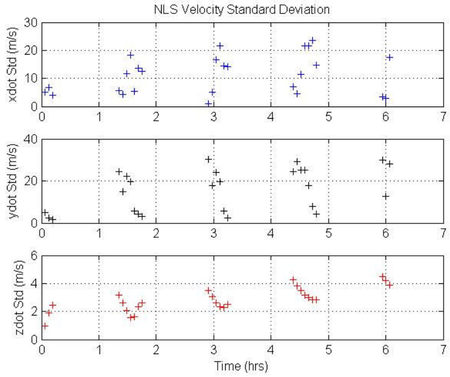


Figure 7.19: NLS Eglin Velocity Standard Deviation (Perturbations Batch 24)

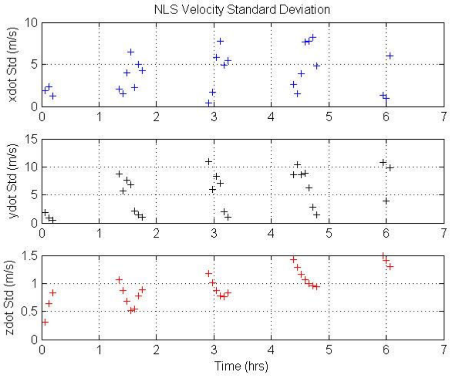


Figure 7.20: NLS Eglin Velocity Standard Deviation (Perturbations Batch 240)

7.1.5 NLS Tracked Debris Object

Figure 7.21 shows the observable unperturbed estimated orbit for the debris object tracked by the 7 sensor sites during the 6-hour propagation time using 24 measurements. The 240-measurement batch will not show a difference, since the size of the errors is much smaller than the size of the axes in the plots. In addition, the perturbed orbit will not show a difference. The NLS estimator using Socorro sensor data does not estimate the orbit as accurately as the phased array sensors. Figure 7.22 provides a close up view of Figure 7.21. The Socorro sensor is a GEODDS optical sensor that does not observe the range of the debris object. As a result, the position and velocity is estimated using only the angle measurements. This impairs the ability of the estimator to fit the data accurately. See appendix C for NLS Socorro sensor results.

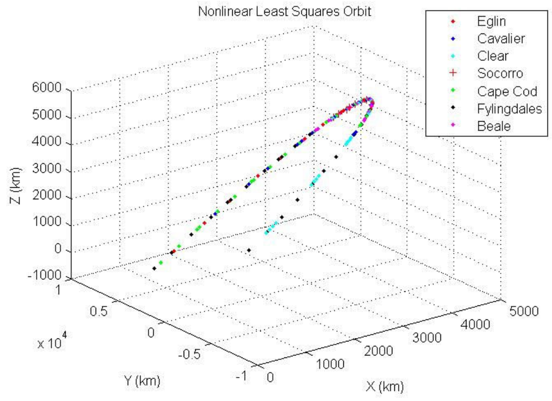


Figure 7.21: NLS Observable Debris Orbit (No Perturbations Batch 24)

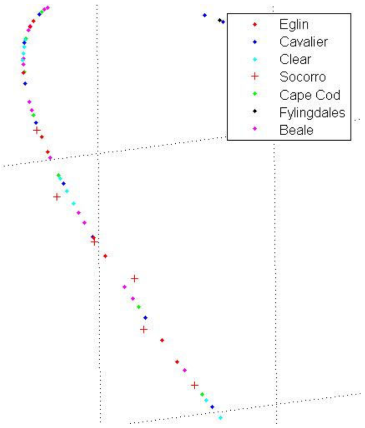


Figure 7.22: NLS Observable Debris Orbit Close-Up (No Perturbations Batch 24)

7.1.6 NLS Estimation Validation

The convergence of the least squares orbit determination algorithm is chosen initially as $1e^{-3}$. The maximum number of iterations is set to 10 to reduce computer-processing time. For the unperturbed orbit, the NLS estimator using Eglin sensor data achieves a convergence after 2 iterations using both a 24 and 240-measurement batch.

Figure 7.8 and Figure 7.10 show that the standard deviation improves if the estimator uses a dense batch of observations.

For the perturbed orbit, the estimator using 24 measurements does not achieve the specified convergence before the maximum iterations is reached. Instead of increasing the iteration limit, the tolerance is changed to $1e^{-2}$. The estimator still does not converge after this tolerance. This indicates that there are not enough observations for the estimator to provide a proper fit using the linearized force model equations. The estimator is then tested using a dense set of observations. The estimator achieves a $1e^{-2}$ convergence before the maximum 10 iterations for a batch of 240 measurements. The dense observation batch has also improved the standard deviations in the position and velocity estimates.

The nonlinear least squares estimator performed in the same way for the other phased array sensors that use range, azimuth, and elevation measurements. As a result, those plots are not shown. The Socorro GEODSS sensor uses only azimuth and elevation measurements. The unperturbed orbit estimates using Socorro observations are worse compared to the estimates from the phased array sensors (batch 24 and 240). The estimates improve if the dense batch of measurements is used; however, they are still highly inaccurate. The NLS estimator cannot estimate the position and velocity of the debris object using Socorro data for an orbit with perturbations. The NLS state error covariance information is useless, as it does not reflect the state errors correctly. This indicates that the estimated model does not match the physical model correctly.[18] The range measurement is required to accurately fit the observations to the true position and velocity of the debris object in orbit. If a dense batch of measurements is used, the estimates become even less accurate. Using more bad measurements does not help to estimate the state accurately.

7.2 Extended Kalman Filter Tracking

The extended Kalman filter method is used to determine the position and velocity of the debris object. The orbital debris tracking algorithm is set up similarly to the batch least squares estimation approach. Initially, a measurement is generated every 10 seconds during each 240-second (4 minute) observation interval. The EKF estimator uses each measurement sequentially to obtain an estimate. The estimates are propagated forward to the end of the 240-second batch section and then propagated backward to the start time of the batch section. As a result, the EKF obtains an estimate of the initial position and velocity at the start of the batch section. Initially, each batch contains 24 measurements. Results are then obtained for a batch of 240 measurements. The debris object then follows a propagated trajectory of approximately 6 hours using the estimated position and velocity data every 240 seconds. Chapter 8 provides information on the MATLAB functions created for this problem.

7.2.1 EKF Eglin Position and Velocity Errors (No Perturbations)

The following results are for the Eglin sensor when the true orbit does not include perturbations. The results are shown for a batch of 24 and 240 measurements. Figures (7.23-7.24) show the observable true orbit and estimated orbit for the Eglin sensor. Figures (7.25-7.26) show the position and velocity errors for the state estimates.

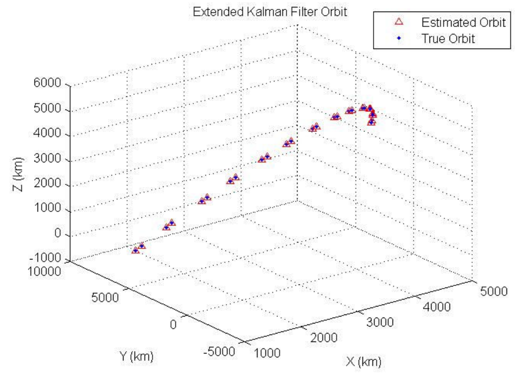


Figure 7.23: EKF Eglin Debris Orbit
(No Perturbations Batch 24)

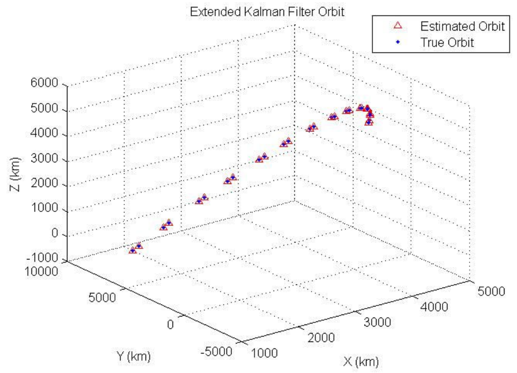


Figure 7.24: EKF Eglin Debris Orbit
(No Perturbations Batch 240)

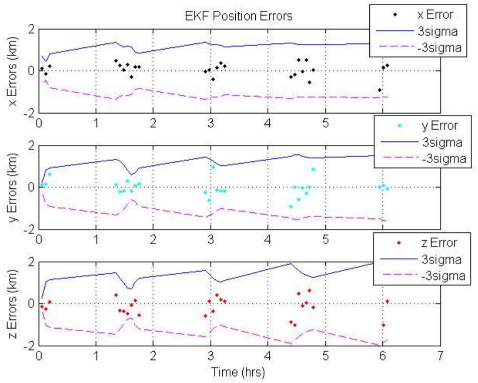


Figure 7.25: EKF Eglin Position Errors
(No Perturbations Batch 24)

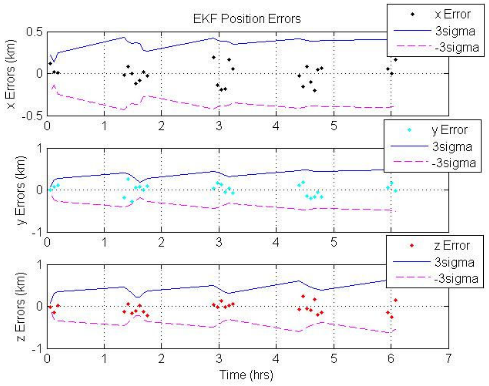


Figure 7.26: EKF Eglin Position Errors
(No Perturbations Batch 240)

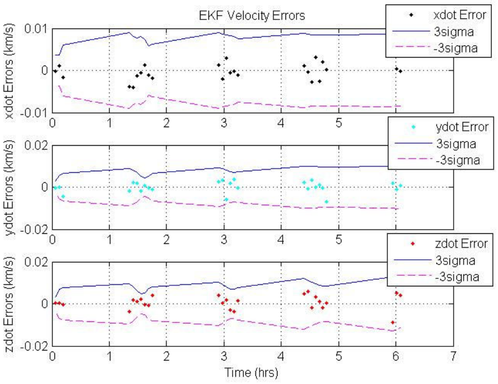


Figure 7.27: EKF Eglin Velocity Errors
(No Perturbations Batch 24)

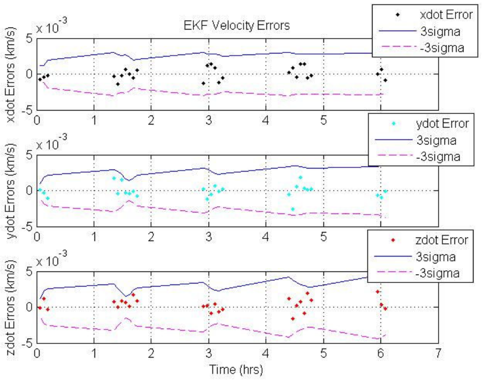


Figure 7.28: EKF Eglin Velocity Errors
(No Perturbations Batch 240)

7.2.2 EKF Eglin Standard Deviations (No Perturbations)

Figures (7.29-7.32) give the position and velocity standard deviations for the Eglin sensor when the true orbit does not include perturbations.

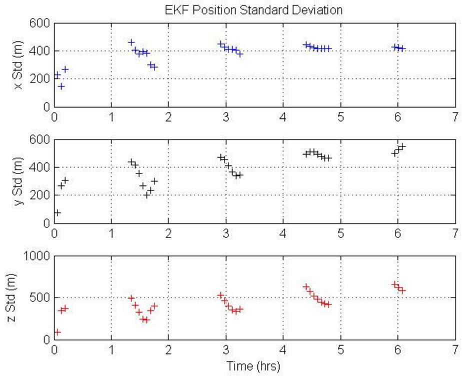


Figure 7.29: EKF Eglin Position Standard Deviation (No Perturbations Batch 24)

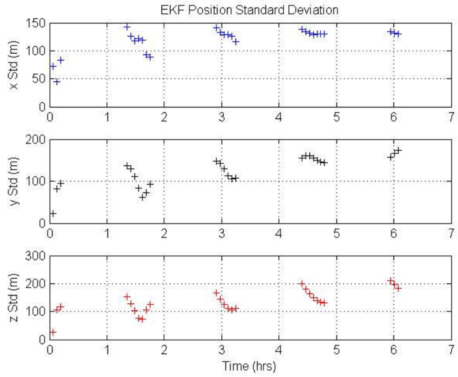


Figure 7.30: EKF Eglin Position Standard Deviation (No Perturbations Batch 240)

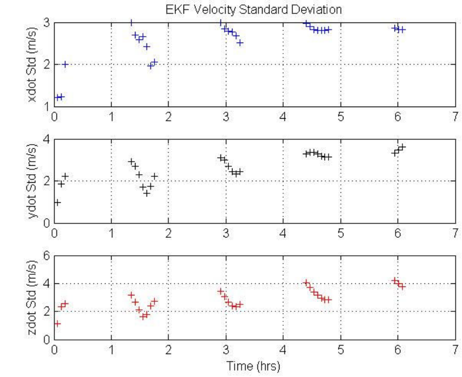


Figure 7.31: EKF Eglin Velocity Standard Deviation (No Perturbations Batch 24)

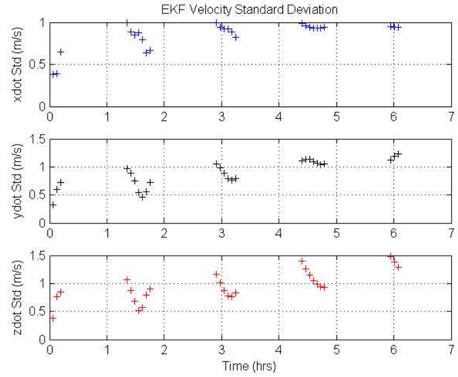


Figure 7.32: EKF Eglin Velocity Standard Deviation (No Perturbations Batch 240)

7.2.3 EKF Eglin Position and Velocity Errors (Perturbations)

The following results are for the Eglin sensor when the orbit has perturbations. Figures (7.33-7.34) show the observable true orbit and estimated orbit for the Eglin sensor. Figures (7.35-7.38) show the errors for the state estimates.

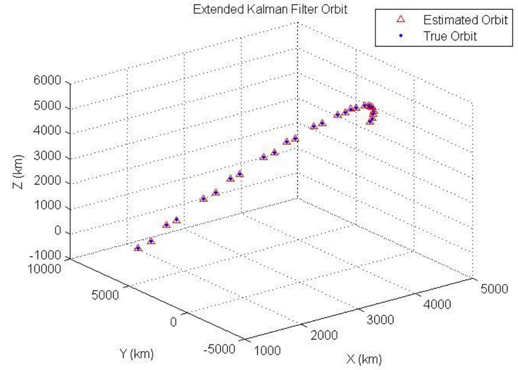


Figure 7.33: EKF Eglin Debris Orbit (Perturbations Batch 24)

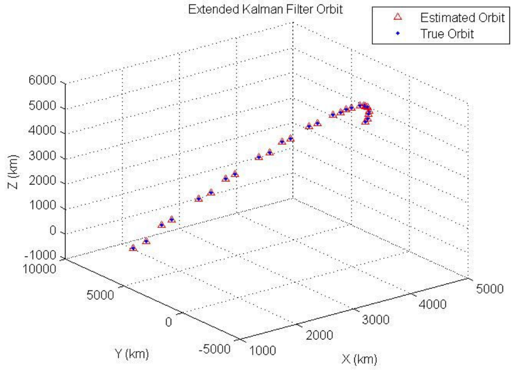


Figure 7.34: EKF Eglin Debris Orbit (Perturbations Batch 240)

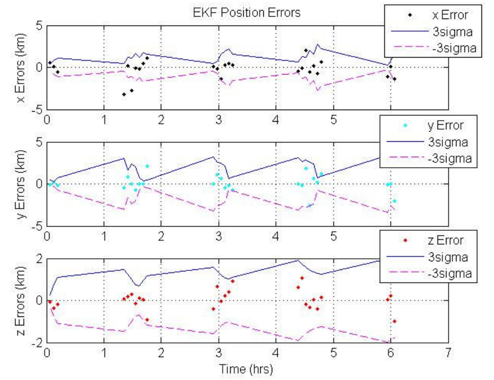


Figure 7.35: EKF Eglin Position Errors (Perturbations Batch 24)

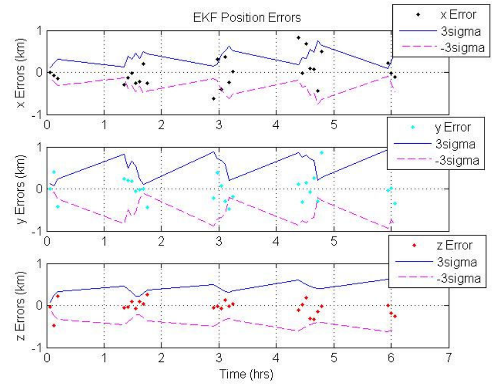


Figure 7.36: EKF Eglin Position Errors (Perturbations Batch 240)

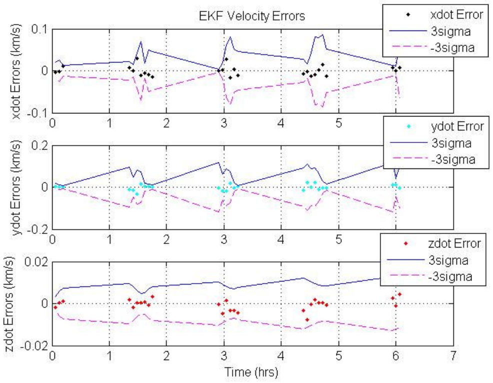


Figure 7.37: EKF Eglin Velocity Errors (Perturbations Batch 24)

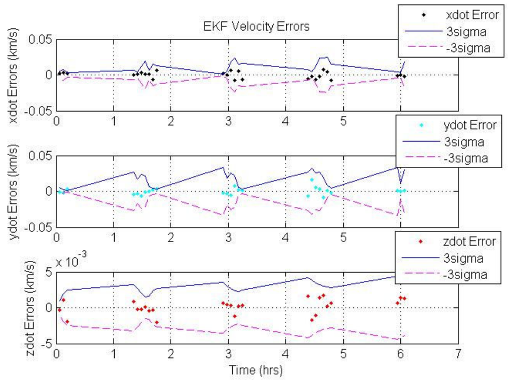


Figure 7.38: EKF Eglin Velocity Errors (Perturbations Batch 240)

7.2.4 EKF Eglin Standard Deviations (Perturbations)

Figures (7.39-7.42) show the position and velocity standard deviations for the Eglin sensor when the true orbit has perturbations.

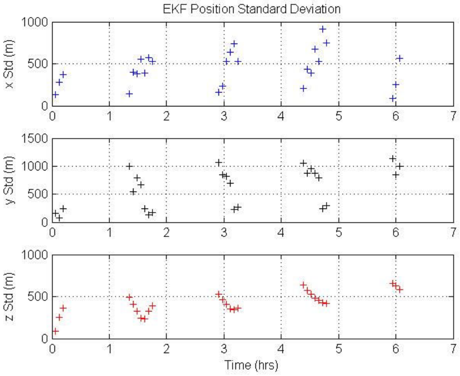


Figure 7.39: EKF Eglin Position Standard Deviation (Perturbations Batch 24)

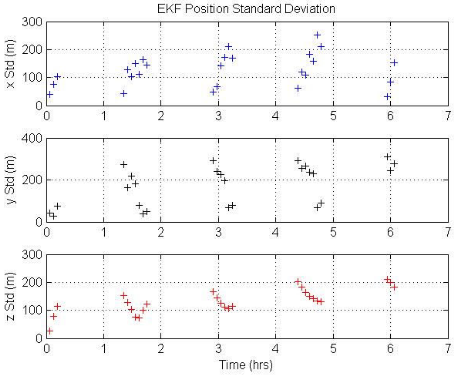


Figure 7.40: EKF Eglin Position Standard Deviation (Perturbations Batch 240)

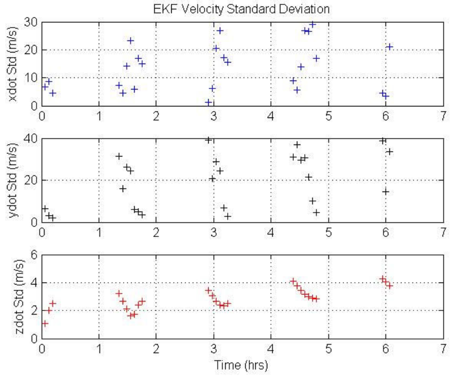


Figure 7.41: EKF Eglin Velocity Standard Deviation (Perturbations Batch 24)

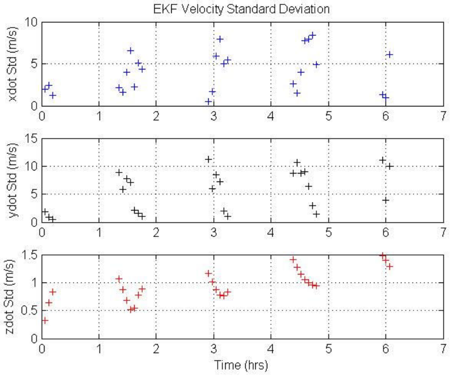


Figure 7.42: EKF Eglin Velocity Standard Deviation (Perturbations Batch 240)

7.2.5 EKF Tracked Debris Object

Figure 7.43 shows the observable estimated orbit for the debris object tracked by the 7 sensor sites during the 6-hour propagation time using 24 measurements. The 240-

measurement batch will not show a difference, since the size of the errors is much smaller than the size of the axes in the plots. In addition, the perturbed orbit will not show a difference. Figure 7.44 provides a close up view of the tracked orbit. Similar to the NLS, the EKF estimator using Socorro sensor data does not estimate the orbit as accurately as the phased array sensors. See appendix C for EKF Socorro sensor results. The EKF estimator; however, has better results for the unperturbed orbit compared to the NLS estimator using Socorro sensor data. This is further discussed in the next section of this chapter.

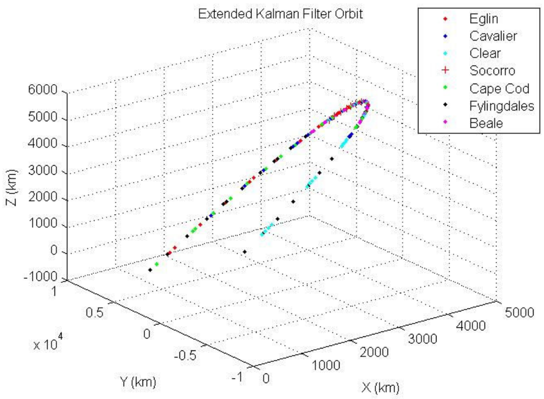


Figure 7.43: EKF Observable Debris Orbit (No Perturbations Batch 24)

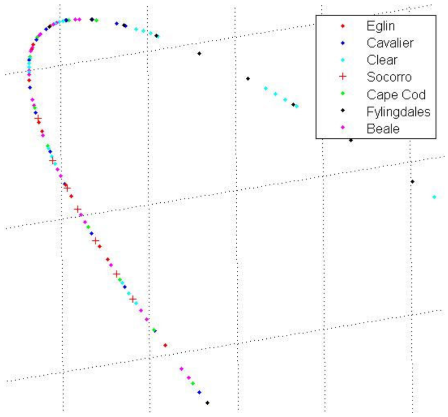


Figure 7.44: EKF Observable Debris Orbit Close-Up (No Perturbations Batch 24)

7.2.6 EKF Estimation Validation

The ode45 integrator is used to integrate the EKF state estimates instead of the RK4 integrator. The RK4 integrator requires an appropriate time step that can be obtained through trial and error. A time step of dt/100 provides adequate estimates. The ode45 integrator; however, is much faster and gives similar results. NLS uses the RK4 integrator. With RK4 it is easier to control the number of states integrated using a time step input. This is helpful with the batch least squares approach, as the state estimates are required to be the same size as the measurements.

The extended Kalman filter estimates were obtained using 1 iteration. Increasing iterations does not improve the accuracy of the estimate. For the unperturbed orbit, the extended Kalman filter estimates are comparable to the nonlinear least square estimates. The EKF estimates improve if the estimator uses a dense batch of measurements. The extended Kalman filter method state estimate accuracy decreases for the orbit that has perturbations. Increasing the measurements improves the accuracy of the estimates. The EKF estimator provides better state estimates compared to the NLS estimator for an unperturbed orbit using Socorro observations (batch 24 and 240). However, the estimates using Socorro observations are still worse compared to the estimates from the other phased array sensors including Eglin. Furthermore, similar to the NLS estimator, the EKF estimator cannot estimate the position and velocity of the debris object using Socorro data for an orbit with perturbations. The EKF state error covariance information is useless. This indicates that the estimated model does not match the physical model correctly.[18] The range measurement is required to accurately fit the observations to the true position and velocity of the debris object in orbit. If a dense batch of measurements is used, the estimates become even less accurate. Similarly with the NLS estimator, using more bad measurements does not help to estimate the state accurately.

Overall, the EKF does not produce better results than the NLS estimator. This indicates that the baseline EKF used for this research requires tuning to improve the results. The process noise error covariance matrix should be incorporated into the estimator and updated continuously. Future work will acknowledge this component for the EKF.

CHAPTER 8

MATLAB CODE DOCUMENTATION

To achieve the research objectives MATLAB codes have been developed. Figures (8.1-8.11) provide flowcharts showing the MATLAB functions used.

8.1 Background Calculations

The **(ECIcalculation.m)** algorithm evaluates the LST using sensor site longitude data provided in Table 1.3. The GMST, also known as the right ascension of Greenwich, on (Oct 1, 1995, 9h UTC) is calculated to be 144.627 deg. The Julian date is obtained using **(Juliandate.m)** function. Figure 8.1 provides a flowchart for the **(ECIcalculation.m)** algorithm.

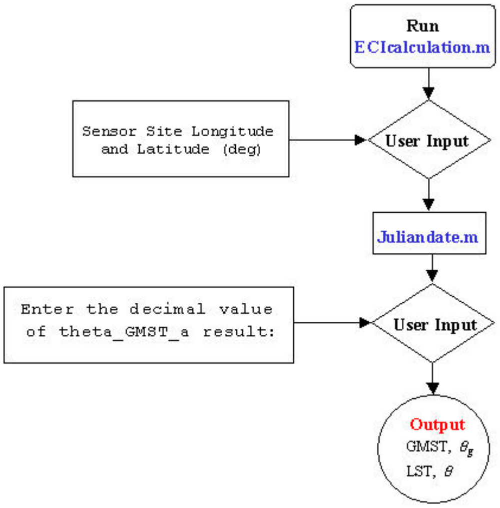

Run
ECIcalculation.m
Sensor Site Longitude and Latitude (deg)
User Input
Juliandate.m
Enter the decimal value of theta_GMST_a result:
User Input
Output
GMST, θ_g
LST, θ

Figure 8.1: Background Calculations

The **(rectTOorbital.m)** algorithm converts the rectangular (cartesian) coordinates to orbital coordinates. Specifically, the inclination and period of the orbit using the initial position and velocity of the ISS orbit is evaluated.

8.2 Measurement Generation (Part I)

(True_Orbit_text_files.m) generates true orbit (position and velocity) data for a debris object in a 10-hour orbit (1 second step interval). The true orbit data is generated for a dynamic model with or without perturbations. Two text files are created for each dynamic model. The time of observation and (position and velocity) data are stored in separate text files. The true initial position and velocity of the debris object in the ISS orbit is taken as the true ISS position and velocity (beginning on orbit 1998) provided in chapter 1 (section 1.5.2). Figure 8.2 provides a flowchart for the **(True_Orbit_text_files.m)** algorithm.

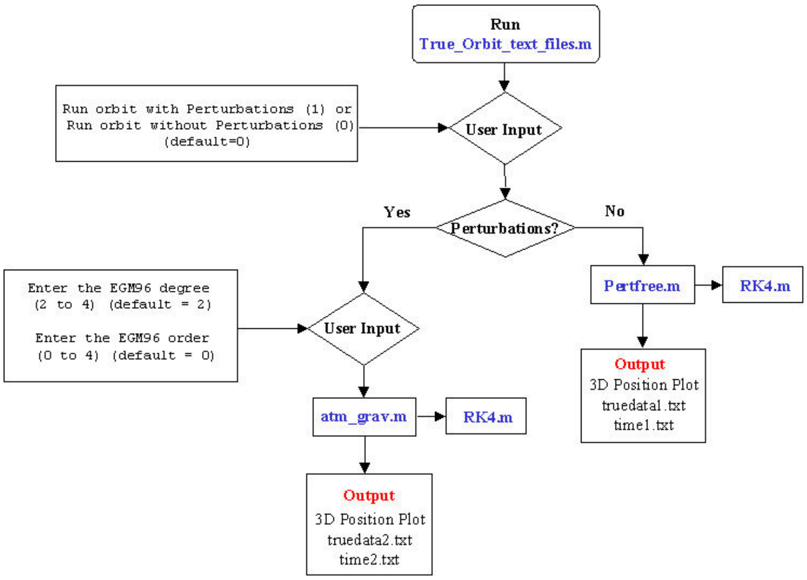


Figure 8.2: True Orbit Generation

8.2.1 Dynamic Model

To run **(True_Orbit_text_files.m)** without atmospheric drag and gravity perturbations the user enters (default) 0. This value is then stored as **(ent=0)**. The **(PertFree.m)** function contains the basic two-body orbital equation of motion dynamic model for this choice. To run **(True_Orbit_text_files.m)** with atmospheric drag and gravity perturbations the user enters 1. This value is then stored as **(ent=1)**. The gravity coefficients (degree and order) are chosen by the user. The **(atm_grav.m)** function contains the dynamic model for this choice. This function uses the basic two-body orbital equation of motion with acceleration due to atmospheric drag (exponential model) and gravity (maximum 4 X 4 complex gravity model). The user chooses the degree (default 2) and order (default 0) of the complex gravity model.

The **(geopartial.m)** function solves the three partial derivatives of the geopotential with respect to (r,δ,γ). Within this function the gravity coefficients and the Legendre polynomials with argument $(\sin\delta)$ are calculated. The **(gravmodel.m)** function calculates the (C_n^m) and (S_n^m) coefficients depending on the degree and order chosen for the gravity model. **(gravitymodel.txt)** is opened and stored as variable **gmodel**. The degree and order is then matched and the coefficients are obtained from the stored table of data. The **(legpoly.m)** function evaluates the Legendre polynomial with argument $(\sin\delta)$ for a 4 X 4 gravity model. This function can be expanded to include larger uncertainties. However, this will require a lot more computer processing time. The research requirement is for a 4 X 4 gravity model only. Equations (2.15) are the first few expansions for the associated Legendre Function used in **(legpoly.m)**. If

(ent=0) the true orbit data and time are stored as **(truedata1.txt and time1.txt)**. If **(ent=1)** the true orbit data and time (with exponential atmospheric drag and the 4 X 4 gravity model) are stored as **(truedata2.txt and time2.txt)**. Figure 8.3 provides a flowchart for the functions used within **(atm_grav.m)**.

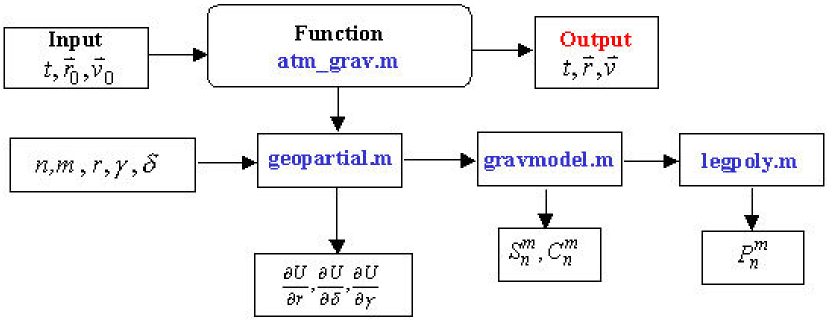

Input
$t, \vec{r}_0, \vec{v}_0$
Function
atm_grav.m
Output
$t, \vec{r}, \vec{v}$
n, m, r, γ, δ
geopartial.m
gravmodel.m
legpoly.m
$\frac{\partial U}{\partial r}, \frac{\partial U}{\partial \delta}, \frac{\partial U}{\partial \gamma}$
S_n^m, C_n^m
P_n^m

Figure 8.3: Perturbation Functions

8.2.2 Observability Check

The **(True_Gen.m)** algorithm checks for the observability of the debris object for each of the 9 sensor sites provided in Table 1.3. Sensor sites that have less than 360 degrees of azimuth coverage use the **(radarcheck.m)** function. This function evaluates the complete two-step check for observability as mentioned in Chapter 3 (section 3.4). Sensor sites with 360 degrees of azimuth coverage use the **(radarcheck1.m)** function. This function ignores step two for the observability check, since there is no need to evaluate the azimuth angle coverage. Both functions also check for the minimum and maximum elevation angles for each sensor site (See Table 3.1). Two text files (for each dynamic model) are created for each of the 9 sensor sites. The sensor site number, time, and true orbit (position and velocity) data are stored in the text files. The size of text file (column length) is given as an output. This value needs to be updated for the **(Measurement_Gen.m)** and **(Estimation_Gen.m)** algorithms if the **(True_Gen.m)** algorithm is run. The NAVSPASUR fence uses the **(radarcheckfence.m**) to check for observability of the debris object. Figure 8.4 provides a flowchart for the **(True_Gen.m)** algorithm and the text files created.

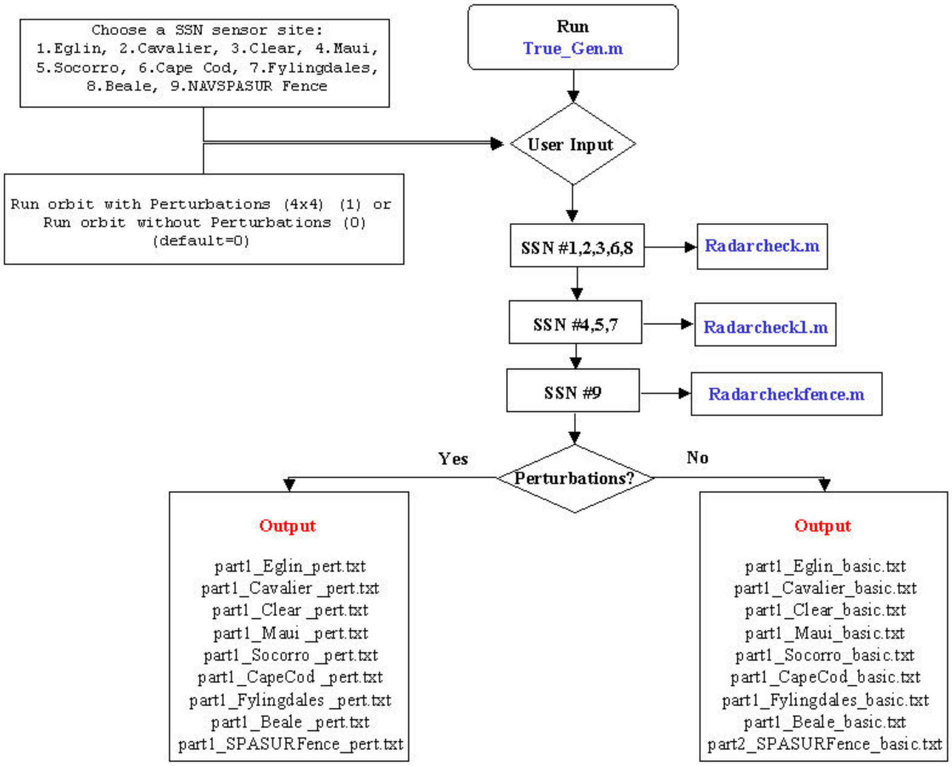


Figure 8.4: Sensor Observability Checks

8.3 Measurement Generation (Part II)

The **(Measurement_Gen.m)** algorithm reads the true orbit files and corrupts each of the sensor measurements using the **(measurements.m)** function. The range, azimuth and elevation measurements are obtained for the phased array radar sites using the true position and velocity data for each observation. The azimuth and elevation measurements are obtained for the GEODSS sites (4 and 5) and the NAVSPASUR fence (9) using the true position and velocity data for each observation. The column length is required to reshape the read text file data into a mxn matrix. Chapter 5 discusses the error type chosen for range, azimuth and elevation measurements. The measurement time, sensor number and corrupted measurements are written to an "actual" measurement file that is read by the orbit estimation program. The "actual" measurement file can be edited without having to re-integrate the true orbit. For a statistical analysis of the outcome of the orbit estimation process this is very convenient. Two text files (for each dynamic model) are created for each of the 9 sensor sites. The **(vecnorm.m)** function used in the **(measurements.m)** function takes the norm of each $x(i,:)$ to produce a vector norm output, i.e. $out = \sqrt{x(:,1)^2 + x(:,2)^2 + \cdots}$. This function was written by John L. Crassidis that is provided in reference [35]. Figure 8.5 provides a flowchart for the **(Measurement_Gen.m)** algorithm and the text files created.

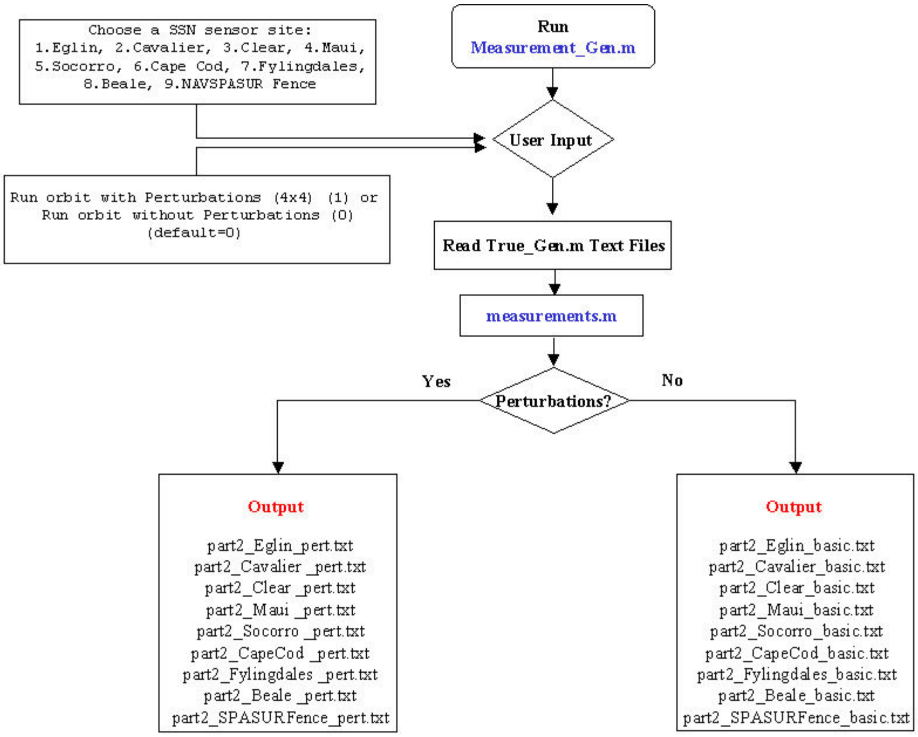


Figure 8.5: Measurement Generation

8.4 Estimation Process

The **(Estimation_Gen.m)** algorithm evaluates the orbit determination problem for the debris object using each of the 9 sensor sites. The program runs one sensor site at a time. To run **(Estimation_Gen.m)** without atmospheric drag and gravity perturbations the user inputs (default) 0. This value is then stored as **(ent=0)**. To run **(Estimation_Gen.m)** with atmospheric drag and gravity perturbations the user inputs 1. This value is then stored as **(ent=1)**. The algorithm will run for the degree and order specified in the **(True_Orbit_text_files.m)** algorithm. The 4 X 4 complex gravity is used based on the research objective. The user can choose to run the Nonlinear Least Squares (NLS) method (default, tech=0) or the Extended Kalman Filter (EKF) method (tech=1) as the orbit determination technique. The actual measurement files are read and stored. The observation data is used for the orbit estimators. The true orbit data file is also read and stored. The true orbit data is used as an initial guess for the orbit estimators. The **(proporbit.m)** function propagates the orbit every 240 seconds using either the NLS **(orbit_NLS.m)** or the EKF **(orbit_ekf.m)** estimator. The actual observations are stored every 1 to 10 seconds (240 or 24 measurement batch) and the initial guess of position and velocity at the start of the observation section is sent to the estimator. The initial time is taken as the current time value of the true initial position and velocity of the observation section. The final time is taken as the value in the observable orbit 240 seconds later. Data output includes the time, state vector and state error covariance matrix. In addition, plots using the **(plotresults.m)** function are generated showing the error in the position and velocity of the estimates and the standard deviations. 3σ bounds are included in the plots. Plots showing the observable true orbit and estimated orbit are also generated for each sensor site. The **(proporbit.m)** function also contains the Herrick-Gibbs method (**HG_method.m**) that can be used to obtain the initial position and velocity of the debris object in order to determine the orbit. This method works if the interval time is small. Figure 8.6 provides a flowchart for the **(Estimation_Gen.m)** algorithm. Figure 8.7 provides a flowchart for the **(proporbit.m)** function.

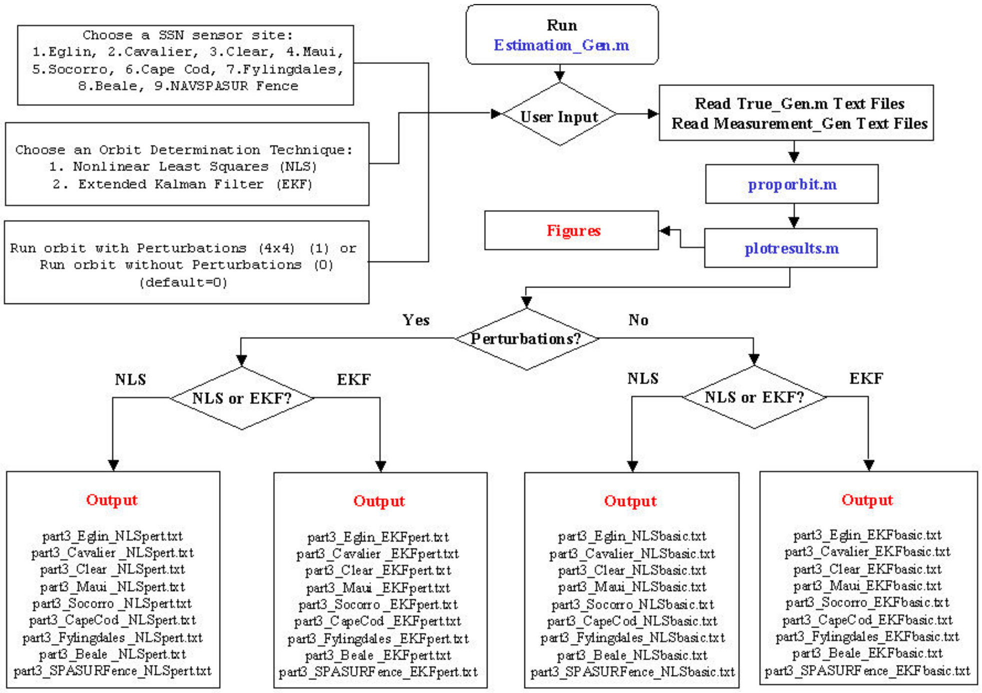

Choose a SSN sensor site:
1.Eglin, 2.Cavalier, 3.Clear, 4.Maui,
5.Socorro, 6.Cape Cod, 7.Fylingdales,
8.Beale, 9.NAVSPASUR Fence
Choose an Orbit Determination Technique:
1. Nonlinear Least Squares (NLS)
2. Extended Kalman Filter (EKF)
Run orbit with Perturbations (4x4) (1) or
Run orbit without Perturbations (0)
(default=0)
Run
Estimation_Gen.m
User Input
Read True_Gen.m Text Files
Read Measurement_Gen Text Files
proporbit.m
Figures
plotresults.m
Yes
Perturbations?
No
NLS
NLS or EKF?
EKF
NLS
NLS or EKF?
EKF
Output
part3_Eglin_NLSpert.txt
part3_Cavalier _NLSpert.txt
part3_Clear _NLSpert.txt
part3_Maui _NLSpert.txt
part3_Socorro _NLSpert.txt
part3_CapeCod _NLSpert.txt
part3_Fylingdales _NLSpert.txt
part3_Beale _NLSpert.txt
part3_SPASURFence_NLSpert.txt
Output
part3_Eglin_EKFpert.txt
part3_Cavalier _EKFpert.txt
part3_Clear _EKFpert.txt
part3_Maui _EKFpert.txt
part3_Socorro _EKFpert.txt
part3_CapeCod _EKFpert.txt
part3_Fylingdales _EKFpert.txt
part3_Beale _EKFpert.txt
part3_SPASURFence_EKFpert.txt
Output
part3_Eglin_NLSbasic.txt
part3_Cavalier_NLSbasic.txt
part3_Clear_NLSbasic.txt
part3_Maui_NLSbasic.txt
part3_Socorro_NLSbasic.txt
part3_CapeCod_NLSbasic.txt
part3_Fylingdales_NLSbasic.txt
part3_Beale_NLSbasic.txt
part3_SPASURFence_NLSbasic.txt
Output
part3_Eglin_EKFbasic.txt
part3_Cavalier_EKFbasic.txt
part3_Clear_EKFbasic.txt
part3_Maui_EKFbasic.txt
part3_Socorro_EKFbasic.txt
part3_CapeCod_EKFbasic.txt
part3_Fylingdales_EKFbasic.txt
part3_Beale_EKFbasic.txt
part3_SPASURFence_EKFbasic.txt

Figure 8.6: Estimation Generation

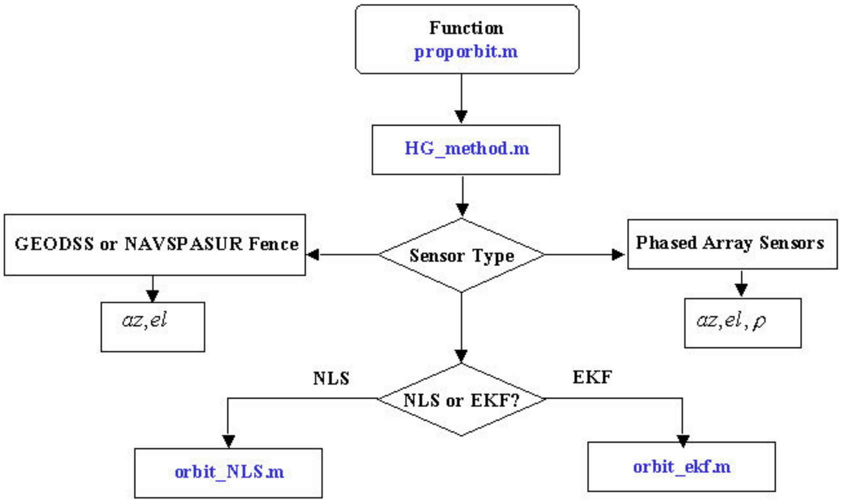


Figure 8.7: Orbit Propagation Function

8.4.1 Nonlinear Least Squares Algorithm

Figure 8.8 provides a flowchart for the **(orbit_NLS.m)** function. The least squares method follows the technique provided in reference [35]. The least squares solution is evaluated in the function (**orbit_NLS.m**). This function estimates the dynamic models provided in chapter 2. The estimator is set to run 10 iterations until a specified convergence is achieved. The entire batch measurement file is used to update the state estimate. The weight matrix is obtained by evaluating the Kronecker product of the Identity matrix (size taken as the number of measurements) and the observation error covariance matrix, $(R_{\text{cov}})^{-1}$.[35] **(kron.m)** is a function provided in the MATLAB library. **Kron(X,Y)** of two matrices is the larger matrix formed from all possible products of the elements of X with those of Y.[53] The discrete phi method is used to evaluate the state transition matrix (Φ). The state matrix, F, is converted from the continuous-time to discrete-time using the **(c2d.m)** function provided in the MATLAB library. The **(orb_prop.m)** function is the state propagation routine used to integrate the state estimates until the end of the batch section. **(orb_prop.m)** contains the dynamic model functions as described in section 8.2.1. **(ent=0)** evaluates the Jacobian for the basic two-body orbital equation of motion. **(ent=1)** evaluates the Jacobian for the equation of motion with acceleration due to atmospheric drag and complex gravity. **RK4.m** is used to integrate the states. The derivations were verified using the MATLAB functions **(Jacobian.m)** and **(GravityJacobian.m)** that evaluate symbolic variables. The acceleration due to gravity partial derivatives with respect to position for the Jacobian matrix are obtained using the **(apospartial.m)** function. Figure 8.9

provides a flowchart for the **(apospartial.m)** function. The size of the text file (column length) is given as an output. This value needs to be updated for each sensor site in the **(Finalplots.m)** algorithm if the **(Estimation_Gen.m)** is run.

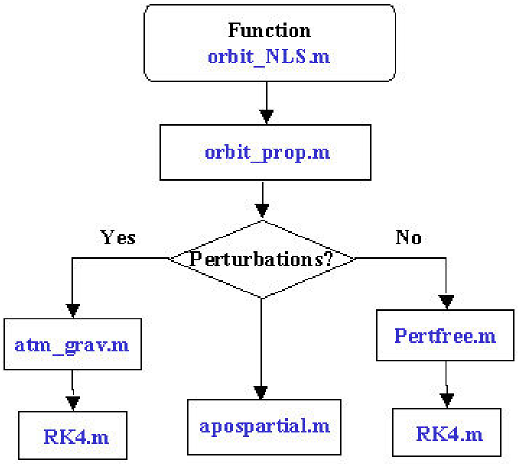


Figure 8.8: Nonlinear Least Squares Function

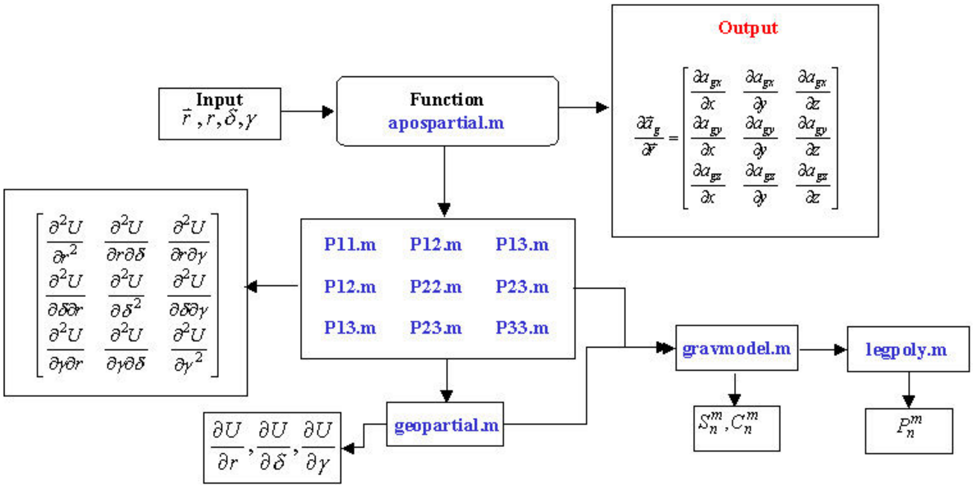

Output
$\frac{\partial \vec{a}_g}{\partial \vec{r}} = \begin{bmatrix} \frac{\partial a_{gx}}{\partial x} & \frac{\partial a_{gx}}{\partial y} & \frac{\partial a_{gx}}{\partial z} \\ \frac{\partial a_{gy}}{\partial x} & \frac{\partial a_{gy}}{\partial y} & \frac{\partial a_{gy}}{\partial z} \\ \frac{\partial a_{gz}}{\partial x} & \frac{\partial a_{gz}}{\partial y} & \frac{\partial a_{gz}}{\partial z} \end{bmatrix}$
Input
$\vec{r}, r, \delta, \gamma$
Function
apospartial.m
$\begin{bmatrix} \frac{\partial^2 U}{\partial r^2} & \frac{\partial^2 U}{\partial r \partial \delta} & \frac{\partial^2 U}{\partial r \partial \gamma} \\ \frac{\partial^2 U}{\partial \delta \partial r} & \frac{\partial^2 U}{\partial \delta^2} & \frac{\partial^2 U}{\partial \delta \partial \gamma} \\ \frac{\partial^2 U}{\partial \gamma \partial r} & \frac{\partial^2 U}{\partial \gamma \partial \delta} & \frac{\partial^2 U}{\partial \gamma^2} \end{bmatrix}$
P11.m P12.m P13.m
P12.m P22.m P23.m
P13.m P23.m P33.m
gravmodel.m
legpoly.m
S_n^m, C_n^m
P_n^m
geopartial.m
$\frac{\partial U}{\partial r}, \frac{\partial U}{\partial \delta}, \frac{\partial U}{\partial \gamma}$

Figure 8.9: Gravity Partial Derivatives Function

8.4.2 Extended Kalman Filter Algorithm

Figure 8.10 provides a flowchart for the **(orbit_ekf.m)** funtion. The extended Kalman filter algorithm follows the technique provided in reference [35]. The extended Kalman filter is evaluated in the function **(orbit_ekf.m)**. This program estimates the dynamic models provided in chapter 2. The program is set to run 1 iteration. **(orbit_ekf.m)** goes through each measurement sequentially to update the state. Furthermore, the estimates are propagated forward and then backward in time to obtain an estimate of the state at the beginning of the batch section. The state matrix, F, is converted from the continuous-time to discrete-time using the **(c2d.m)** function provided in the MATLAB library. The **(orb_prop_ode45.m)** function is the state propagation routine used to integrate the state estimates until the next measurement time. **(orb_prop_ode45.m)** contains the dynamic model functions as described in section 8.2.1. **(ent=0)** evaluates the Jacobian for the basic two-body orbital equation of motion. **(ent=1)** evaluates the Jacobian for the equation of motion with acceleration due to atmospheric drag and complex gravity. **Ode45.m** is used to integrate the states. The derivations were verified using the MATLAB functions (**Jacobian.m)** and **(GravityJacobian.m)** that evaluate symbolic variables. The acceleration due to gravity partial derivatives with respect to position for the Jacobian matrix are obtained using the **(apospartial.m)** function. Figure 8.9 provides a flowchart for the **(apospartial.m)** function. The size of the text file (column length) is given as an output. This value needs to be updated for each sensor site in the **(Finalplots.m)** algorithm if the **(Estimation_Gen.m)** is run.

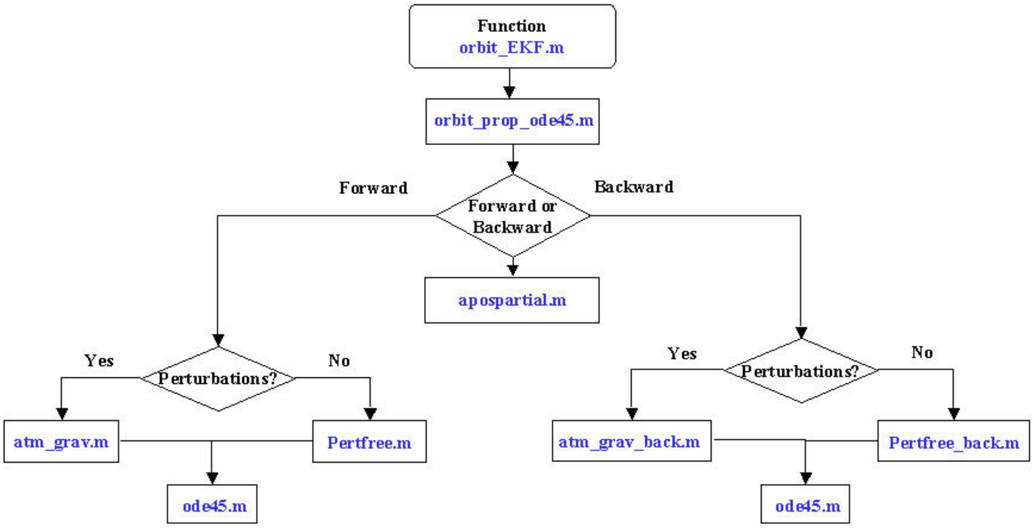


Figure 8.10: Extended Kalman Filter Function

8.4.3 Final Plots

A plot showing the tracked debris orbit using observable SSN sensor sites can be generated using the **(Finalplots.m)** algorithm. Figure 8.11 provides the flowchart used for this algorithm.

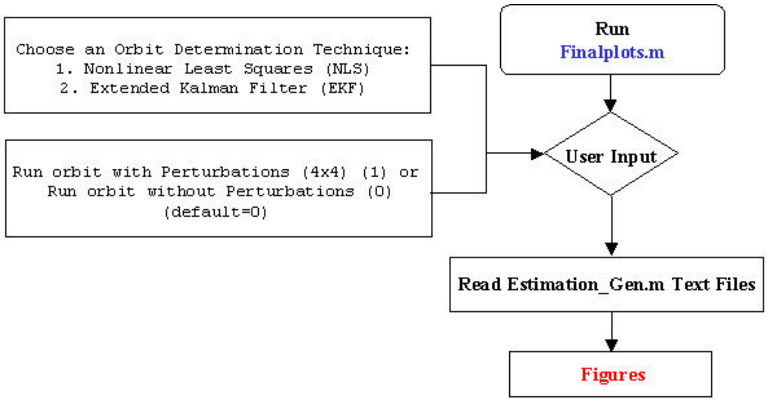

Choose an Orbit Determination Technique:
1. Nonlinear Least Squares (NLS)
2. Extended Kalman Filter (EKF)
Run orbit with Perturbations (4x4) (1) or
Run orbit without Perturbations (0)
(default=0)
Run
Finalplots.m
User Input
Read Estimation_Gen.m Text Files
Figures

Figure 8.11: Final Plots Function

CHAPTER 9

CONCLUSION

The continuous-discrete extended Kalman filter does not improve the state estimates over the nonlinear batch least squares method, for the tracked debris object in an unperturbed orbit using phased array sensor observations (batch 24 and 240). The results in chapter 7 show that the accuracy of the state is worse for both estimators for a dynamic model with perturbations due to atmospheric drag and complex gravity. The state estimates for both the nonlinear batch least squares and the extended Kaman filter improve if a dense batch of measurements is used. This applies to the estimator using both the unperturbed and perturbed dynamic models. Both estimators give inaccurate results using Socorro sensor observations. The EKF estimator performs better than the NLS estimator for the unperturbed orbit using Socorro sensor data. The perturbed orbit; however, is not estimated correctly for both estimators, as the estimators are not able to fit the measurement data to the model equations without the range observation. The state error covariance matrix is useless for both estimators.

The benefit of using the extended Kalman filter over the nonlinear least squares estimator is that it can be tuned to incorporate force model errors. Force model errors for this research stem from the inaccurate exponential atmospheric model and the truncated 4 X 4 complex gravity model. The process noise error covariance matrix can be changed and through a process of trial and error geared towards achieving a better estimate. For this research, the extended Kalman filter was designed to provide comparable results to the nonlinear least square technique. As a result, the process noise error covariance was ignored.

The nonlinear batch least squares technique is the primary estimation technique used in the industry for orbit determination applications.[11] Limited analytical theories are generally used for orbit determination problems.[11] As a result, there has not been much interest in correcting the covariance matrix. For future work related to this research, the extended Kalman filter should be tuned in order to improve the estimates. For a longer fit of tracked data, the tuned EKF will improve computer-processing time, since only 1 iteration is required to obtain estimates. The NLS requires more iterations to achieve a convergence in the state estimates. Another important component that needs to be addressed for future work is the development of a realistic atmospheric model. The exponential atmospheric model used for this research does not incorporate atmospheric uncertainty that is present in the constantly changing atmosphere. Overall, the research is successful. The nine SSN sensors have been recreated to generate measurements of an orbital debris object in an ISS orbit. The debris object tracking MATLAB code developed for this research is versatile as it can be easily altered to accommodate a user's particular problem.

APPENDIX A

NECESSARY EQUATIONS

A-1 Jacobian Partials

Appendix A provides the partial derivatives of the acceleration due to drag with respect to position and velocity.

A-1.1 Acceleration Due to Atmospheric Drag

$$\frac{\partial a_{dx}}{\partial x} = \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\omega_e\left(\dot{y}-\omega_e x\right)\left(\dot{x}+\omega_e y\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dx}}{\partial y} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\omega_e\left[2\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dx}}{\partial z} = 0$$

$$\frac{\partial a_{dy}}{\partial x} = \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\omega_e\left[\left(\dot{x}+\omega_e y\right)^2+2\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dy}}{\partial y} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\omega_e\left(\dot{y}-\omega_e x\right)\left(\dot{x}+\omega_e y\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dy}}{\partial z} = 0$$

$$\frac{\partial a_{dz}}{\partial x} = \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\omega_e\dot{z}\left(\dot{y}-\omega_e x\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dz}}{\partial y} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}\left(\dot{x}+\omega_e y\right)\omega_e\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dz}}{\partial z} = 0$$

$$\frac{\partial a_{dx}}{\partial \dot{x}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[2\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dx}}{\partial \dot{y}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left(\dot{x}+\omega_e y\right)\left(\dot{y}-\omega_e x\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dx}}{\partial \dot{z}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}\left(\dot{x}+\omega_e y\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dy}}{\partial \dot{x}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left(\dot{x}+\omega_e y\right)\left(\dot{y}-\omega_e x\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dy}}{\partial \dot{y}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[\left(\dot{x}+\omega_e y\right)^2+2\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dy}}{\partial \dot{z}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}\left(\dot{y}-\omega_e x\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dz}}{\partial \dot{x}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}\left(\dot{x}+\omega_e y\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dz}}{\partial \dot{y}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}\left(\dot{y}-\omega_e x\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dz}}{\partial \dot{z}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+2\dot{z}^2\right]\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

APPENDIX B

DERIVATIONS

B-1 Acceleration Due to Complex Gravity Derivations

B-1.1 Acceleration Due to Complex Gravity Cartesian Components

Equations (2.11) are given by:

$$\frac{\partial r}{\partial \vec{r}} = \frac{\vec{r}^{\,T}}{r}$$
$$\frac{\partial \delta}{\partial \vec{r}} = \frac{1}{\sqrt{x^2+y^2}}\left(-\frac{\vec{r}^{\,T} z}{r^2} + \frac{\partial z}{\partial \vec{r}}\right)$$
$$\frac{\partial \gamma}{\partial \vec{r}} = \frac{1}{x^2+y^2}\left(x\frac{\partial y}{\partial \vec{r}} - y\frac{\partial x}{\partial \vec{r}}\right)$$

where,

$$\vec{r} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$
$$r = \sqrt{x^2+y^2+z^2}$$

The expansions for the components are given by the following:

$$\frac{\partial r}{\partial \vec{r}} = \frac{\begin{bmatrix} x \\ y \\ z \end{bmatrix}^T}{r}$$
$$= \begin{bmatrix} \frac{x}{r} & \frac{y}{r} & \frac{z}{r} \end{bmatrix}$$

$$\frac{\partial \delta}{\partial \vec{r}} = \frac{1}{\sqrt{x^2+y^2}}\left(-\frac{\begin{bmatrix} x & y & z \end{bmatrix} z}{r^2} + \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}\right) = \frac{1}{\sqrt{x^2+y^2}}\left(-\frac{\begin{bmatrix} zx & zy & z^2 \end{bmatrix}}{r^2} + \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}\right)$$
$$= \frac{1}{\sqrt{x^2+y^2}}\left(\begin{bmatrix} \frac{-zx}{r^2} & \frac{-zy}{r^2} & \frac{1-z^2}{r^2} \end{bmatrix}\right) = \frac{1}{\sqrt{x^2+y^2}}\left(\begin{bmatrix} \frac{-zx}{r^2} & \frac{-zy}{r^2} & \frac{\left(x^2+y^2\right)}{r^2} \end{bmatrix}\right)$$
$$= \begin{bmatrix} \frac{-zx}{r^2\sqrt{x^2+y^2}} & \frac{-zy}{r^2\sqrt{x^2+y^2}} & \frac{\sqrt{x^2+y^2}}{r^2} \end{bmatrix}$$

$$\frac{\partial\gamma}{\partial\vec{r}}=\frac{1}{x^2+y^2}\left(x\frac{\partial y}{\partial\vec{r}}-y\frac{\partial x}{\partial\vec{r}}\right)=\frac{1}{x^2+y^2}\left(x\begin{bmatrix}0 & 1 & 0\end{bmatrix}-y\begin{bmatrix}1 & 0 & 0\end{bmatrix}\right)$$
$$=\frac{1}{x^2+y^2}\begin{bmatrix}-y & x & 0\end{bmatrix}$$
$$=\begin{bmatrix}\frac{-y}{x^2+y^2} & \frac{x}{x^2+y^2} & 0\end{bmatrix}$$

The acceleration due to complex gravity in cartesian coordinates is given by:

$$\vec{a}_g(\vec{r},t)=\frac{\partial U}{\partial r}\left(\frac{\partial r}{\partial\vec{r}}\right)^T+\frac{\partial U}{\partial\delta}\left(\frac{\partial\delta}{\partial\vec{r}}\right)^T+\frac{\partial U}{\partial\gamma}\left(\frac{\partial\gamma}{\partial\vec{r}}\right)^T$$
$$=\frac{\partial U}{\partial r}\left(\begin{bmatrix}\frac{x}{r} & \frac{y}{r} & \frac{z}{r}\end{bmatrix}\right)^T+\frac{\partial U}{\partial\delta}\left(\begin{bmatrix}\frac{-zx}{r^2\sqrt{x^2+y^2}} & \frac{-zy}{r^2\sqrt{x^2+y^2}} & \frac{\sqrt{x^2+y^2}}{r^2}\end{bmatrix}\right)^T$$
$$+\frac{\partial U}{\partial\gamma}\left(\begin{bmatrix}\frac{-y}{x^2+y^2} & \frac{x}{x^2+y^2} & 0\end{bmatrix}\right)^T$$
$$=\frac{\partial U}{\partial r}\begin{bmatrix}\frac{x}{r}\\ \frac{y}{r}\\ \frac{z}{r}\end{bmatrix}+\frac{\partial U}{\partial\delta}\begin{bmatrix}\frac{-zx}{r^2\sqrt{x^2+y^2}}\\ \frac{-zy}{r^2\sqrt{x^2+y^2}}\\ \frac{\sqrt{x^2+y^2}}{r^2}\end{bmatrix}+\frac{\partial U}{\partial\gamma}\begin{bmatrix}\frac{-y}{x^2+y^2}\\ \frac{x}{x^2+y^2}\\ 0\end{bmatrix}$$

The components of the debris object’s acceleration vector then become:

$$a_x=\left(\frac{1}{r}\frac{\partial U}{\partial r}-\frac{z}{r^2\sqrt{x^2+y^2}}\frac{\partial U}{\partial\delta}\right)x-\left(\frac{1}{x^2+y^2}\frac{\partial U}{\partial\gamma}\right)y$$
$$a_y=\left(\frac{1}{r}\frac{\partial U}{\partial r}-\frac{z}{r^2\sqrt{x^2+y^2}}\frac{\partial U}{\partial\delta}\right)y+\left(\frac{1}{x^2+y^2}\frac{\partial U}{\partial\gamma}\right)x$$
$$a_z=\left(\frac{1}{r}\frac{\partial U}{\partial r}\right)z+\left(\frac{\sqrt{x^2+y^2}}{r^2}\frac{\partial U}{\partial\delta}\right)$$

B-2 Coordinate Transformations

The coordinate transformations discussed in chapter 4 are shown in this section.

B-2.1 Topocentric Coordinate System Transformation

The following trigonometric identities are used to evaluate the topocentric coordinates (ρ_s, ρ_e, ρ_u):

$$
\begin{aligned}
&\sin(-t) = -\sin(t) \\
&\cos(-t) = \cos(t) \\
&\sin(90^0 - t) = \sin(90)\cos(t) - \cos(90)\sin(t) = \cos(t) \\
&\cos(90^0 - t) = \cos(90)\cos(t) + \sin(90)\sin(t) = \sin(t)
\end{aligned}
$$

The topocentric coordinates become:

$$
\begin{aligned}
\begin{bmatrix} \rho_s \\ \rho_e \\ \rho_u \end{bmatrix} &= C_2(90^0 - \phi)C_3(\theta)\vec{\rho} \\
&= \begin{bmatrix} \cos(90^0 - \phi) & 0 & -\sin(90^0 - \phi) \\ 0 & 1 & 0 \\ \sin(90^0 - \phi) & 0 & \cos(90^0 - \phi) \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \vec{\rho} \\
&= \begin{bmatrix} \sin\phi & 0 & -\cos\phi \\ 0 & 1 & 0 \\ \cos\phi & 0 & \sin\phi \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \vec{\rho} \\
&= \begin{bmatrix} \cos\theta\sin\phi & \sin\theta\sin\phi & -\cos\phi \\ -\sin\theta & \cos\theta & 0 \\ \cos\theta\cos\phi & \sin\theta\cos\phi & \sin\phi \end{bmatrix} \vec{\rho}
\end{aligned}
$$

B-2.2 Earth-Centered Inertial Frame Transformation

The following trigonometric identities are used to evaluate the ECI coordinates (x, y, z) for the three position vectors:

$$
\begin{aligned}
&\sin(-t) = -\sin(t) \\
&\cos(-t) = \cos(t) \\
&\sin(-90^0 + t) = \sin(-90)\cos(t) + \cos(-90)\sin(t) = -\cos(t) \\
&\cos(-90^0 + t) = \cos(-90)\cos(t) - \sin(-90)\sin(t) = \sin(t)
\end{aligned}
$$

The three position vectors become:

$$\vec{r}_k = C_3(-\theta_k)C_2\left(-\left(90^0 - \phi_k\right)\right)\left(\vec{\rho}_{seu}\right)_K + \vec{R}_k,\ k = 1,2,3$$

$$
= \begin{bmatrix} \cos(-\theta_k) & \sin(-\theta_k) & 0 \\ -\sin(-\theta_k) & \cos(-\theta_k) & 0 \\ 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} \cos\left(-\left(90^0 - \phi_k\right)\right) & 0 & -\sin\left(-\left(90^0 - \phi_k\right)\right) \\ 0 & 1 & 0 \\ \sin\left(-\left(90^0 - \phi_k\right)\right) & 0 & \cos\left(-\left(90^0 - \phi_k\right)\right) \end{bmatrix}
\left(\vec{\rho}_{seu}\right)_K + \vec{R}_k
$$

$$
= \begin{bmatrix} \cos\theta_k & -\sin\theta_k & 0 \\ \sin\theta_k & \cos\theta_k & 0 \\ 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} \sin\phi_k & 0 & \cos\phi_k \\ 0 & 1 & 0 \\ -\cos\phi_k & 0 & \sin\phi_k \end{bmatrix}
\left(\vec{\rho}_{seu}\right)_K + \vec{R}_k
$$

$$
= \begin{bmatrix} \sin\phi_k \cos\theta_k & -\sin\theta_k & \cos\phi_k \cos\theta_k \\ \sin\phi_k \sin\theta_k & \cos\theta_k & \cos\phi_k \sin\theta_k \\ -\cos\phi_k & 0 & \sin\phi_k \end{bmatrix}
\left(\vec{\rho}_{seu}\right)_K + \vec{R}_k
$$

B-3 Jacobian Derivations

This section provides the derivations of the Jacobian for the dynamic models.

B-3.1 Basic Two-Body Equation of Motion

The dynamic model without perturbations is given by equation (2.1) as:

$$\vec{a} = -\frac{\mu}{r^3}\vec{r}$$

where,

$$r = \sqrt{x^2 + y^2 + z^2}$$

$$\vec{a} = -\frac{\mu}{\left(x^2 + y^2 + z^2\right)^{\frac{3}{2}}}\begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

The partial derivatives become:

$$\frac{\partial \vec{a}}{\partial \vec{r}} = \begin{bmatrix} \frac{\partial}{\partial x}\left(-\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}x\right) & \frac{\partial}{\partial y}\left(-\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}x\right) & \frac{\partial}{\partial z}\left(-\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}x\right) \\ \frac{\partial}{\partial x}\left(-\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}y\right) & \frac{\partial}{\partial y}\left(-\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}y\right) & \frac{\partial}{\partial z}\left(-\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}y\right) \\ \frac{\partial}{\partial x}\left(-\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}z\right) & \frac{\partial}{\partial y}\left(-\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}z\right) & \frac{\partial}{\partial z}\left(-\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}}z\right) \end{bmatrix}$$

$$= \begin{bmatrix} -\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}} + \frac{3\mu x^2}{\left(x^2+y^2+z^2\right)^{\frac{5}{2}}} & \frac{3\mu xy}{\left(x^2+y^2+z^2\right)^{\frac{5}{2}}} & \frac{3\mu xz}{\left(x^2+y^2+z^2\right)^{\frac{5}{2}}} \\ \frac{3\mu xy}{\left(x^2+y^2+z^2\right)^{\frac{5}{2}}} & -\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}} + \frac{3\mu y^2}{\left(x^2+y^2+z^2\right)^{\frac{5}{2}}} & \frac{3\mu yz}{\left(x^2+y^2+z^2\right)^{\frac{5}{2}}} \\ \frac{3\mu xz}{\left(x^2+y^2+z^2\right)^{\frac{5}{2}}} & \frac{3\mu yz}{\left(x^2+y^2+z^2\right)^{\frac{5}{2}}} & -\frac{\mu}{\left(x^2+y^2+z^2\right)^{\frac{3}{2}}} + \frac{3\mu z^2}{\left(x^2+y^2+z^2\right)^{\frac{5}{2}}} \end{bmatrix}$$

$$\boxed{\frac{\partial \vec{a}}{\partial \vec{r}} = \begin{bmatrix} \frac{3\mu x^2}{r^5} - \frac{\mu}{r^3} & \frac{3\mu xy}{r^5} & \frac{3\mu xz}{r^5} \\ \frac{3\mu xy}{r^5} & \frac{3\mu y^2}{r^5} - \frac{\mu}{r^3} & \frac{3\mu yz}{r^5} \\ \frac{3\mu xz}{r^5} & \frac{3\mu yz}{r^5} & \frac{3\mu z^2}{r^5} - \frac{\mu}{r^3} \end{bmatrix}}$$

B-3.2 Acceleration Due to Atmospheric Drag

The dynamic model with perturbations is given by equation (2.3) as:

$$\vec{a} = -\frac{\mu}{r^3}\vec{r} + \vec{a}_d(\vec{r},\vec{v},t) + \vec{a}_g(\vec{r},t)$$

The acceleration due to atmospheric drag is given by:

$$\vec{a}_d(\vec{r},\vec{v},t) = -\frac{1}{2}\frac{C_d A}{m}\rho_{atm}\|\vec{v}_r\|\vec{v}_r$$

with the velocity relative to the atmosphere as:

$$\vec{v}_r = \vec{v} - \vec{\omega}\times\vec{r} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} - \omega_e \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} - \begin{bmatrix} -\omega_e y \\ \omega_e x \\ 0 \end{bmatrix} = \begin{bmatrix} \dot{x} + \omega_e y \\ \dot{y} - \omega_e x \\ \dot{z} \end{bmatrix}$$

The acceleration due to atmospheric drag becomes:

$$\vec{a}_d(\vec{r},\vec{v},t) = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\begin{bmatrix} \dot{x} + \omega_e y \\ \dot{y} - \omega_e x \\ \dot{z} \end{bmatrix}$$

The components are given by:

$$a_{dx} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,(\dot{x}+\omega_e y)$$
$$a_{dy} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,(\dot{y}-\omega_e x)$$
$$a_{dz} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,(\dot{z})$$

The components of the partial derivatives are given by:

$$\frac{\partial \vec{a}_d}{\partial \vec{r}} = \begin{bmatrix} \frac{\partial a_{dx}}{\partial x} & \frac{\partial a_{dx}}{\partial y} & \frac{\partial a_{dx}}{\partial z} \\ \frac{\partial a_{dy}}{\partial x} & \frac{\partial a_{dy}}{\partial y} & \frac{\partial a_{dy}}{\partial z} \\ \frac{\partial a_{dz}}{\partial x} & \frac{\partial a_{dz}}{\partial y} & \frac{\partial a_{dz}}{\partial z} \end{bmatrix} \qquad \frac{\partial \vec{a}}{\partial \vec{v}} = \begin{bmatrix} \frac{\partial a_{dx}}{\partial \dot{x}} & \frac{\partial a_{dx}}{\partial \dot{y}} & \frac{\partial a_{dx}}{\partial \dot{z}} \\ \frac{\partial a_{dy}}{\partial \dot{x}} & \frac{\partial a_{dy}}{\partial \dot{y}} & \frac{\partial a_{dy}}{\partial \dot{z}} \\ \frac{\partial a_{dz}}{\partial \dot{x}} & \frac{\partial a_{dz}}{\partial \dot{y}} & \frac{\partial a_{dz}}{\partial \dot{z}} \end{bmatrix}$$

The partial derivatives with respect to position become:

$$a_{dx} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,(\dot{x}+\omega_e y)$$

$$\frac{\partial a_{dx}}{\partial x} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}(\dot{x}+\omega_e y)(\dot{y}-\omega_e x)(-\omega_e)$$

$$= \frac{1}{2}\rho_{atm}\frac{C_d A}{m} w_e (\dot{y}-\omega_e x)(\dot{x}+\omega_e y)\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}$$

$$a_{dx} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,(\dot{x}+\omega_e y)$$

$$\frac{\partial a_{dx}}{\partial y} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}(\dot{x}+\omega_e y)(\dot{x}+\omega_e y)\omega_e$$

$$-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,\omega_e$$

$$= \frac{-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}(\dot{x}+\omega_e y)(\dot{x}+\omega_e y)\omega_e}{\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{\frac{1}{2}}} - \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,\omega_e$$

$$= \frac{-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}(\dot{x}+\omega_e y)^2\omega_e - \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]\omega_e}{\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{\frac{1}{2}}}$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\omega_e\left[2(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dx}}{\partial z} = 0$$

$$a_{dy} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}\,(\dot{y}-\omega_e x)$$

$$\frac{\partial a_{dy}}{\partial x} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}(\dot{y}-\omega_e x)(\dot{y}-\omega_e x)(-\omega_e)$$

$$-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}\,(-\omega_e)$$

$$=\frac{-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}(\dot{y}-\omega_e x)(\dot{y}-\omega_e x)(-\omega_e)}{\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{\frac{1}{2}}}-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}\,(-\omega_e)$$

$$=\frac{\frac{1}{2}\rho_{atm}\frac{C_d A}{m}(\dot{y}-\omega_e x)^2\omega_e+\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]\omega_e}{\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{\frac{1}{2}}}$$

$$=\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\omega_e\left[(\dot{x}+\omega_e y)^2+2(\dot{y}-\omega_e x)^2+\dot{z}^2\right]\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}$$

$$a_{dy} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}\,(\dot{y}-\omega_e x)$$

$$\frac{\partial a_{dy}}{\partial y} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}(\dot{y}-\omega_e x)(\dot{x}+\omega_e y)\omega_e$$

$$=-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\omega_e(\dot{y}-\omega_e x)(\dot{x}+\omega_e y)\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dy}}{\partial z}=0$$

$$a_{dz} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}\,(\dot{z})$$

$$\frac{\partial a_{dz}}{\partial x} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}(\dot{z})(\dot{y}-\omega_e x)(-\omega_e)$$

$$=\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\omega_e\dot{z}(\dot{y}-\omega_e x)\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}$$

$$a_{dz} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2}\left(\dot{z}\right)$$

$$\frac{\partial a_{dz}}{\partial y} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}\left(\dot{z}\right)\left(\dot{x}+\omega_e y\right)\omega_e$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}\left(\dot{x}+\omega_e y\right)\omega_e\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$\frac{\partial a_{dz}}{\partial z} = 0$$

The partial derivatives with respect to velocity become:

$$a_{dx} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2}\left(\dot{x}+\omega_e y\right)$$

$$\frac{\partial a_{dx}}{\partial \dot{x}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}\left(\dot{x}+\omega_e y\right)\left(\dot{x}+\omega_e y\right)$$

$$-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2}$$

$$= \frac{-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left(\dot{x}+\omega_e y\right)\left(\dot{x}+\omega_e y\right)}{\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{\frac{1}{2}}} - \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2}$$

$$= \frac{-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left(\dot{x}+\omega_e y\right)^2-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]}{\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{\frac{1}{2}}}$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[2\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$a_{dx} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2}\left(\dot{x}+\omega_e y\right)$$

$$\frac{\partial a_{dx}}{\partial \dot{y}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}\left(\dot{x}+\omega_e y\right)\left(\dot{y}-\omega_e x\right)$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left(\dot{x}+\omega_e y\right)\left(\dot{y}-\omega_e x\right)\left[\left(\dot{x}+\omega_e y\right)^2+\left(\dot{y}-\omega_e x\right)^2+\dot{z}^2\right]^{-1/2}$$

$$a_{dx} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}\,(\dot{x}+\omega_e y)$$

$$\frac{\partial a_{dx}}{\partial \dot{z}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}(\dot{x}+\omega_e y)\dot{z}$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}(\dot{x}+\omega_e y)\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}$$

$$a_{dy} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}\,(\dot{y}-\omega_e x)$$

$$\frac{\partial a_{dy}}{\partial \dot{x}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}(\dot{x}+\omega_e y)(\dot{y}-\omega_e x)$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}(\dot{x}+\omega_e y)(\dot{y}-\omega_e x)\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}$$

$$a_{dy} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}\,(\dot{y}-\omega_e x)$$

$$\frac{\partial a_{dy}}{\partial \dot{y}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}(\dot{y}-\omega_e x)(\dot{y}-\omega_e x)$$

$$-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}$$

$$= \frac{-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}(\dot{y}-\omega_e x)^2}{\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{\frac{1}{2}}} - \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}$$

$$= \frac{-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}(\dot{y}-\omega_e x)^2 - \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]}{\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{\frac{1}{2}}}$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+2(\dot{y}-\omega_e x)^2+\dot{z}^2\right]\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}$$

$$a_{dy} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2}\,(\dot{y}-\omega_e x)$$

$$\frac{\partial a_{dy}}{\partial \dot{z}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}(\dot{y}-\omega_e x)\dot{z}$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}(\dot{y}-\omega_e x)\left[(\dot{x}+\omega_e y)^2+(\dot{y}-\omega_e x)^2+\dot{z}^2\right]^{-1/2}$$

$$a_{dz} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,(\dot{z})$$

$$\frac{\partial a_{dz}}{\partial \dot{x}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}(\dot{z})(\dot{x}+\omega_e y)$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}(\dot{x}+\omega_e y)\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}$$

$$a_{dz} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,(\dot{z})$$

$$\frac{\partial a_{dz}}{\partial \dot{y}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}(\dot{z})(\dot{y}-\omega_e x)$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}(\dot{y}-\omega_e x)\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}$$

$$a_{dz} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}\,(\dot{z})$$

$$\frac{\partial a_{dz}}{\partial \dot{z}} = -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}(\dot{z})(\dot{z})$$

$$-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}$$

$$= \frac{-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}^2}{\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{\frac{1}{2}}} - \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\sqrt{(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2}$$

$$= \frac{-\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\dot{z}^2 - \frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]}{\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{\frac{1}{2}}}$$

$$= -\frac{1}{2}\rho_{atm}\frac{C_d A}{m}\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + 2\dot{z}^2\right]\left[(\dot{x}+\omega_e y)^2 + (\dot{y}-\omega_e x)^2 + \dot{z}^2\right]^{-1/2}$$

B-3.3 Acceleration Due to Complex Gravity

As mentioned in chapter 6, the partial derivatives for the acceleration due to complex gravity are not derived. However, the following expansions are shown, as they are required for MATLAB algorithms.

$$\frac{\partial^2 r}{\partial \vec{r}^2} = \frac{1}{r}\left[I - \frac{\vec{r}\vec{r}^T}{r^2}\right]$$

$$\frac{\partial^2 \delta}{\partial \vec{r}^2} = -\frac{1}{\left(x^2 + y^2\right)^{\frac{3}{2}}}\left(\left(\frac{\partial z}{\partial \vec{r}}\right)^T - \frac{z\vec{r}}{r^2}\right)\left[x\left(\frac{\partial x}{\partial \vec{r}}\right) + y\left(\frac{\partial y}{\partial \vec{r}}\right)\right] - \frac{1}{r^2\sqrt{x^2 + y^2}}\left[\vec{r}\left(\frac{\partial z}{\partial \vec{r}}\right) + zI - \frac{2z}{r^2}\vec{r}\vec{r}^T\right]$$

$$\frac{\partial^2 \gamma}{\partial \vec{r}^2} = -\frac{2}{\left(x^2 + y^2\right)}\begin{bmatrix} -y \\ x \\ 0 \end{bmatrix}\left(x\frac{\partial x}{\partial \vec{r}} + y\frac{\partial y}{\partial \vec{r}}\right) + \frac{1}{\left(x^2 + y^2\right)}\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The expansions are given as follows:

$$\frac{\partial^2 r}{\partial \vec{r}^2} = \frac{1}{r}\left[I - \frac{\vec{r}\vec{r}^T}{r^2}\right] = \frac{1}{r}\left\{I - \frac{1}{r^2}\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\begin{bmatrix} x & y & z \end{bmatrix}\right)\right\} = \frac{1}{r}\left\{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{1}{r^2}\begin{bmatrix} x^2 & xy & xz \\ xy & y^2 & yz \\ xz & yz & z^2 \end{bmatrix}\right\}$$

$$= \frac{1}{r}\begin{bmatrix} 1 - \frac{x^2}{r^2} & \frac{-xy}{r^2} & \frac{-xz}{r^2} \\ \frac{-xy}{r^2} & 1 - \frac{y^2}{r^2} & \frac{-yz}{r^2} \\ \frac{-xz}{r^2} & \frac{-yz}{r^2} & 1 - \frac{z^2}{r^2} \end{bmatrix} = \frac{1}{r}\begin{bmatrix} \frac{r^2 - x^2}{r^2} & \frac{-xy}{r^2} & \frac{-xz}{r^2} \\ \frac{-xy}{r^2} & \frac{r^2 - y^2}{r^2} & \frac{-yz}{r^2} \\ \frac{-xz}{r^2} & \frac{-yz}{r^2} & \frac{r^2 - z^2}{r^2} \end{bmatrix}$$

where,

$$r^2 = x^2 + y^2 + z^2$$
$$r^2 - x^2 = y^2 + z^2$$
$$r^2 - y^2 = x^2 + z^2$$
$$r^2 - z^2 = x^2 + y^2$$

then,

$$\boxed{\frac{\partial^2 r}{\partial \vec{r}^2} = \begin{bmatrix} \frac{y^2+z^2}{r^3} & \frac{-xy}{r^3} & \frac{-xz}{r^3} \\ \frac{-xy}{r^3} & \frac{x^2+z^2}{r^3} & \frac{-yz}{r^3} \\ \frac{-xz}{r^3} & \frac{-yz}{r^3} & \frac{x^2+y^2}{r^3} \end{bmatrix}}$$

$$\frac{\partial^2 \delta}{\partial \vec{r}^2} = -\frac{1}{\left(x^2+y^2\right)^{\frac{3}{2}}}\left(\left(\frac{\partial z}{\partial \vec{r}}\right)^T - \frac{z\vec{r}}{r^2}\right)\left[x\left(\frac{\partial x}{\partial \vec{r}}\right) + y\left(\frac{\partial y}{\partial \vec{r}}\right)\right] - \frac{1}{r^2\sqrt{x^2+y^2}}\left[\vec{r}\left(\frac{\partial z}{\partial \vec{r}}\right) + zI - \frac{2z}{r^2}\vec{r}\vec{r}^{\,T}\right]$$

$$= -\frac{1}{\left(x^2+y^2\right)^{\frac{3}{2}}}\left(\begin{bmatrix}0 & 0 & 1\end{bmatrix}^T - \frac{z\begin{bmatrix}x \\ y \\ z\end{bmatrix}}{r^2}\right)\left(x\begin{bmatrix}1 & 0 & 0\end{bmatrix} + y\begin{bmatrix}0 & 1 & 0\end{bmatrix}\right)$$

$$-\frac{1}{r^2\sqrt{x^2+y^2}}\left[\begin{bmatrix}x \\ y \\ z\end{bmatrix}\begin{bmatrix}0 & 0 & 1\end{bmatrix} + z\begin{bmatrix}1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1\end{bmatrix} - \frac{2z}{r^2}\begin{bmatrix}x^2 & xy & xz \\ xy & y^2 & yz \\ xz & yz & z^2\end{bmatrix}\right]$$

$$= -\frac{1}{\left(x^2+y^2\right)^{\frac{3}{2}}}\left(\begin{bmatrix}0 \\ 0 \\ 1\end{bmatrix} - z\begin{bmatrix}\frac{x}{r^2} \\ \frac{y}{r^2} \\ \frac{z}{r^2}\end{bmatrix}\right)\begin{bmatrix}x & y & 0\end{bmatrix} - \frac{1}{r^2\sqrt{x^2+y^2}}\left[\begin{bmatrix}0 & 0 & x \\ 0 & 0 & y \\ 0 & 0 & z\end{bmatrix} + \begin{bmatrix}z & 0 & 0 \\ 0 & z & 0 \\ 0 & 0 & z\end{bmatrix} - \frac{2z}{r^2}\begin{bmatrix}x^2 & xy & xz \\ xy & y^2 & yz \\ xz & yz & z^2\end{bmatrix}\right]$$

$$= -\frac{1}{\left(x^2+y^2\right)^{\frac{3}{2}}}\begin{bmatrix}\frac{-zx}{r^2} \\ \frac{-zy}{r^2} \\ 1-\frac{z^2}{r^2}\end{bmatrix}\begin{bmatrix}x & y & 0\end{bmatrix} - \frac{1}{r^2\sqrt{x^2+y^2}}\left[\begin{bmatrix}0 & 0 & x \\ 0 & 0 & y \\ 0 & 0 & z\end{bmatrix} + \begin{bmatrix}z & 0 & 0 \\ 0 & z & 0 \\ 0 & 0 & z\end{bmatrix} - \frac{2z}{r^2}\begin{bmatrix}x^2 & xy & xz \\ xy & y^2 & yz \\ xz & yz & z^2\end{bmatrix}\right]$$

$$= -\frac{1}{\left(x^2+y^2\right)^{\frac{3}{2}}} \begin{bmatrix} \frac{-zx}{r^2} \\ \frac{-zy}{r^2} \\ \frac{r^2-z^2}{r^2} \end{bmatrix} \begin{bmatrix} x & y & 0 \end{bmatrix} - \frac{1}{r^2\sqrt{x^2+y^2}} \begin{bmatrix} z-\frac{2z}{r^2}x^2 & -\frac{2z}{r^2}xy & x-\frac{2z}{r^2}xz \\ -\frac{2z}{r^2}xy & z-\frac{2z}{r^2}y^2 & y-\frac{2z}{r^2}yz \\ -\frac{2z}{r^2}xz & -\frac{2z}{r^2}yz & 2z-\frac{2z}{r^2}z^2 \end{bmatrix}$$

$$= -\frac{1}{r^2\left(x^2+y^2\right)^{\frac{3}{2}}} \begin{bmatrix} -zx^2 & -zxy & 0 \\ -zxy & -zy^2 & 0 \\ \left(r^2-z^2\right)x & \left(r^2-z^2\right)y & 0 \end{bmatrix} - \frac{1}{r^4\sqrt{x^2+y^2}} \begin{bmatrix} \left(zr^2-2zx^2\right) & -2zxy & \left(xr^2-2xz^2\right) \\ -2zxy & \left(zr^2-2zy^2\right) & \left(yr^2-2yz^2\right) \\ -2xz^2 & -2yz^2 & \left(2zr^2-2z^3\right) \end{bmatrix}$$

$$= \begin{bmatrix} \frac{zx^2}{r^2\left(x^2+y^2\right)^{\frac{3}{2}}} - \frac{\left(zr^2-2zx^2\right)}{r^4\sqrt{x^2+y^2}} & \frac{zxy}{r^2\left(x^2+y^2\right)^{\frac{3}{2}}} + \frac{2zxy}{r^4\sqrt{x^2+y^2}} & -\frac{\left(xr^2-2xz^2\right)}{r^4\sqrt{x^2+y^2}} \\ \frac{zxy}{r^2\left(x^2+y^2\right)^{\frac{3}{2}}} + \frac{2zxy}{r^4\sqrt{x^2+y^2}} & \frac{zy^2}{r^2\left(x^2+y^2\right)^{\frac{3}{2}}} - \frac{\left(zr^2-2zy^2\right)}{r^4\sqrt{x^2+y^2}} & -\frac{\left(yr^2-2yz^2\right)}{r^4\sqrt{x^2+y^2}} \\ \frac{-\left(r^2-z^2\right)x}{r^2\left(x^2+y^2\right)^{\frac{3}{2}}} + \frac{2xz^2}{r^4\sqrt{x^2+y^2}} & \frac{-\left(r^2-z^2\right)y}{r^2\left(x^2+y^2\right)^{\frac{3}{2}}} + \frac{2yz^2}{r^4\sqrt{x^2+y^2}} & -\frac{\left(2zr^2-2z^3\right)}{r^4\sqrt{x^2+y^2}} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{zx^2r^2-\left(zr^2-2zx^2\right)\left(x^2+y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & \frac{zxyr^2+2zxy\left(x^2+y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & -\frac{\left(xr^2-2xz^2\right)}{r^4\sqrt{x^2+y^2}} \\ \frac{zxyr^2+2zxy\left(x^2+y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & \frac{zy^2r^2-\left(zr^2-2zy^2\right)\left(x^2+y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & -\frac{\left(yr^2-2yz^2\right)}{r^4\sqrt{x^2+y^2}} \\ \frac{-\left(r^2-z^2\right)xr^2+2xz^2\left(x^2+y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & \frac{-\left(r^2-z^2\right)yr^2+2yz^2\left(x^2+y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & -\frac{\left(2zr^2-2z^3\right)}{r^4\sqrt{x^2+y^2}} \end{bmatrix}$$

$$\boxed{\frac{\partial^2\delta}{\partial\bar{r}^2} = \begin{bmatrix} \frac{z\left(-r^2y^2+2x^4+2x^2y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & \frac{zxy\left(r^2+2x^2+2y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & -\frac{x\left(r^2-2z^2\right)}{r^4\sqrt{x^2+y^2}} \\ \frac{zxy\left(r^2+2x^2+2y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & \frac{z\left(-r^2x^2+2y^4+2x^2y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & -\frac{y\left(r^2-2z^2\right)}{r^4\sqrt{x^2+y^2}} \\ \frac{x\left(-r^4+r^2z^2+2x^2z^2+2z^2y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & \frac{y\left(-r^4+r^2z^2+2x^2z^2+2z^2y^2\right)}{r^4\left(x^2+y^2\right)^{\frac{3}{2}}} & -\frac{2z\left(r^2-z^2\right)}{r^4\sqrt{x^2+y^2}} \end{bmatrix}}$$

$$\frac{\partial^2\gamma}{\partial\bar{r}^2} = -\frac{2}{\left(x^2+y^2\right)}\begin{bmatrix}-y\\ x\\ 0\end{bmatrix}\left(x\frac{\partial x}{\partial\bar{r}}+y\frac{\partial y}{\partial\bar{r}}\right)+\frac{1}{\left(x^2+y^2\right)}\begin{bmatrix}0&-1&0\\1&0&0\\0&0&0\end{bmatrix}$$

$$= -\frac{2}{\left(x^2+y^2\right)}\begin{bmatrix}-y\\ x\\ 0\end{bmatrix}\left(x\begin{bmatrix}1&0&0\end{bmatrix}+y\begin{bmatrix}0&1&0\end{bmatrix}\right)+\frac{1}{\left(x^2+y^2\right)}\begin{bmatrix}0&-1&0\\1&0&0\\0&0&0\end{bmatrix} = -\frac{2}{\left(x^2+y^2\right)}\begin{bmatrix}-y\\ x\\ 0\end{bmatrix}\begin{bmatrix}x&y&0\end{bmatrix}$$

$$+\frac{1}{\left(x^2+y^2\right)}\begin{bmatrix}0&-1&0\\1&0&0\\0&0&0\end{bmatrix}$$

$$= -\frac{2}{\left(x^2+y^2\right)}\begin{bmatrix}-xy&-y^2&0\\x^2&yx&0\\0&0&0\end{bmatrix}+\frac{1}{\left(x^2+y^2\right)}\begin{bmatrix}0&-1&0\\1&0&0\\0&0&0\end{bmatrix}$$

$$= -\frac{2}{\left(x^2+y^2\right)}\left\{\begin{bmatrix}-xy&-y^2&0\\x^2&yx&0\\0&0&0\end{bmatrix}+\begin{bmatrix}0&-1&0\\1&0&0\\0&0&0\end{bmatrix}\right\} = \frac{1}{\left(x^2+y^2\right)}\left\{\begin{bmatrix}2xy&2y^2&0\\-2x^2&-2yx&0\\0&0&0\end{bmatrix}+\begin{bmatrix}0&-1&0\\1&0&0\\0&0&0\end{bmatrix}\right\}$$

$$\boxed{\frac{\partial^2\gamma}{\partial\bar{r}^2} = \frac{1}{\left(x^2+y^2\right)}\begin{bmatrix}2xy&\left(2y^2-1\right)&0\\\left(1-2x^2\right)&-2yx&0\\0&0&0\end{bmatrix}}$$

B-3.4 Sensitivity Matrix Partials

The partials given in equations (6.18-6.26) are derived. The slant range vector from equation (4.1) and the topocentric coordinates from equation (4.2) give the following:

$$\bar{\rho} = \begin{bmatrix}x-R\cos\phi\cos\theta\\ y-R\cos\phi\sin\theta\\ z-R\sin\phi\end{bmatrix}$$

$$\begin{bmatrix}\rho_s\\ \rho_e\\ \rho_u\end{bmatrix}=\begin{bmatrix}\cos\theta\sin\phi & \sin\theta\sin\phi & -\cos\phi\\ -\sin\theta & \cos\theta & 0\\ \cos\theta\cos\phi & \sin\theta\cos\phi & \sin\phi\end{bmatrix}\vec{\rho}$$

$$=\begin{bmatrix}\cos\theta\sin\phi & \sin\theta\sin\phi & -\cos\phi\\ -\sin\theta & \cos\theta & 0\\ \cos\theta\cos\phi & \sin\theta\cos\phi & \sin\phi\end{bmatrix}\begin{bmatrix}x-R\cos\phi\cos\theta\\ y-R\cos\phi\sin\theta\\ z-R\sin\phi\end{bmatrix}$$

$$\begin{bmatrix}\rho_s\\ \rho_e\\ \rho_u\end{bmatrix}=\begin{bmatrix}\cos\theta\sin\phi & \sin\theta\sin\phi & -\cos\phi\\ -\sin\theta & \cos\theta & 0\\ \cos\theta\cos\phi & \sin\theta\cos\phi & \sin\phi\end{bmatrix}\begin{bmatrix}x-R\cos\phi\cos\theta\\ y-R\cos\phi\sin\theta\\ z-R\sin\phi\end{bmatrix}$$

$$=\begin{bmatrix}\cos\theta\sin\phi(x-R\cos\phi\cos\theta) & \sin\theta\sin\phi(y-R\cos\phi\sin\theta) & -\cos\phi(z-R\sin\phi)\\ -\sin\theta(x-R\cos\phi\cos\theta) & \cos\theta(y-R\cos\phi\sin\theta) & 0\\ \cos\theta\cos\phi(x-R\cos\phi\cos\theta) & \sin\theta\cos\phi(y-R\cos\phi\sin\theta) & \sin\phi(z-R\sin\phi)\end{bmatrix}$$

where,

$$\begin{aligned}\rho_s &= \cos\theta\sin\phi(x-R\cos\phi\cos\theta)+\sin\theta\sin\phi(y-R\cos\phi\sin\theta)-\cos\phi(z-R\sin\phi)\\ &= x\cos\theta\sin\phi-R\cos\phi\sin\phi\cos^2\theta+y\sin\theta\sin\phi-R\cos\phi\sin\phi\sin^2\theta-z\cos\phi+R\sin\phi\cos\phi\\ &= x\cos\theta\sin\phi+y\sin\theta\sin\phi-z\cos\phi-R\cos\phi\sin\phi\left(\cos^2\theta+\sin^2\theta\right)+R\sin\phi\cos\phi\\ &= x\cos\theta\sin\phi+y\sin\theta\sin\phi-z\cos\phi\end{aligned}$$

$$\begin{aligned}\rho_e &= -\sin\theta(x-R\cos\phi\cos\theta)+\cos\theta(y-R\cos\phi\sin\theta)\\ &= -x\sin\theta+R\cos\phi\cos\theta\sin\theta+y\cos\theta-R\cos\phi\sin\theta\cos\theta\\ &= -x\sin\theta+y\cos\theta\end{aligned}$$

$$\begin{aligned}\rho_u &= \cos\theta\cos\phi(x-R\cos\phi\cos\theta)+\sin\theta\cos\phi(y-R\cos\phi\sin\theta)+\sin\phi(z-R\sin\phi)\\ &= x\cos\theta\cos\phi-R\cos^2\phi\cos^2\theta+y\sin\theta\cos\phi-R\cos^2\phi\sin^2\theta-z\sin\phi+R\sin^2\phi\\ &= x\cos\theta\cos\phi+y\sin\theta\cos\phi+z\sin\phi-R\cos^2\phi\left(\cos^2\theta+\sin^2\theta\right)+R\sin^2\phi\\ &= x\cos\theta\cos\phi+y\sin\theta\cos\phi+z\sin\phi-R\end{aligned}$$

Equations (4.3-4.5) give:

$$\rho = \left(\rho_u^2 + \rho_e^2 + \rho_s^2\right)^{1/2}$$

$$az = \tan^{-1}\left(-\frac{\rho_e}{\rho_s}\right)$$

$$el = \sin^{-1}\left(\frac{\rho_u}{\rho}\right)$$

The partials with respect to (x, y, z) given by equations (6.18-6.26) are then computed as:

$$\frac{\partial \rho}{\partial x} = \frac{1}{2}\left[2\rho_s \cos\theta \sin\phi - 2\rho_e \sin\theta + 2\rho_u \cos\theta \cos\phi\right]/\rho$$
$$= \left(\rho_s \cos\theta \sin\phi - \rho_e \sin\theta + \rho_u \cos\theta \cos\phi\right)/\rho$$

$$\frac{\partial \rho}{\partial y} = \frac{1}{2}\left[2\rho_s \sin\theta \sin\phi + 2\rho_e \cos\theta + 2\rho_u \sin\theta \cos\phi\right]/\rho$$
$$= \left(\rho_s \sin\theta \sin\phi + \rho_e \cos\theta + \rho_u \sin\theta \cos\phi\right)/\rho$$

$$\frac{\partial \rho}{\partial z} = \frac{1}{2}\left[-2\rho_s \cos\phi + 2\rho_u \sin\phi\right]/\rho$$
$$= \left(-\rho_s \cos\phi + \rho_u \sin\phi\right)/\rho$$

$$\frac{\partial az}{\partial x} = \frac{1}{1 + \left(-\frac{\rho_e}{\rho_s}\right)^2}\left[\frac{(-\rho_s)(-\sin\theta) - (-\rho_e)\cos\theta \sin\phi}{\rho_s^{\ 2}}\right]$$
$$= \frac{\rho_s^{\ 2}}{\left(\rho_s^{\ 2} + \rho_e^{\ 2}\right)}\left(\frac{\rho_s \sin\theta + \rho_e \cos\theta \sin\phi}{\rho_s^{\ 2}}\right)$$
$$= \frac{1}{\left(\rho_s^{\ 2} + \rho_e^{\ 2}\right)}\left(\rho_s \sin\theta + \rho_e \cos\theta \sin\phi\right)$$

$$\frac{\partial az}{\partial y}=\frac{1}{1+\left(-\frac{\rho_e}{\rho_s}\right)^2}\left[\frac{(-\rho_s)(\cos\theta)-(-\rho_e)\sin\theta\sin\phi}{\rho_s{}^2}\right]$$

$$=\frac{\rho_s{}^2}{\left(\rho_s{}^2+\rho_e{}^2\right)}\left(\frac{-\rho_s\cos\theta+\rho_e\sin\theta\sin\phi}{\rho_s{}^2}\right)$$

$$=\frac{1}{\left(\rho_s{}^2+\rho_e{}^2\right)}\left(-\rho_s\cos\theta+\rho_e\sin\theta\sin\phi\right)$$

$$\frac{\partial az}{\partial z}=\frac{1}{1+\left(-\frac{\rho_e}{\rho_s}\right)^2}\left[\frac{-(-\rho_e)(-\cos\phi)}{\rho_s{}^2}\right]$$

$$=\frac{\rho_s{}^2}{\left(\rho_s{}^2+\rho_e{}^2\right)}\left(\frac{-\rho_e\cos\phi}{\rho_s{}^2}\right)$$

$$=-\frac{1}{\left(\rho_s{}^2+\rho_e{}^2\right)}\rho_e\cos\phi$$

$$\frac{\partial el}{\partial x}=\frac{1}{\sqrt{1-\left(\frac{\rho_u}{\rho}\right)^2}}\left[\frac{\rho\cos\theta\cos\phi-\rho_u\frac{\partial\rho}{\partial x}}{\rho^2}\right]$$

$$=\frac{\rho}{\sqrt{\rho^2-\rho_u{}^2}}\left[\frac{\rho\cos\theta\cos\phi-\rho_u\frac{\partial\rho}{\partial x}}{\rho^2}\right]$$

$$=\frac{1}{\rho\sqrt{\rho^2-\rho_u{}^2}}\left(\rho\cos\theta\cos\phi-\rho_u\frac{\partial\rho}{\partial x}\right)$$

$$\frac{\partial el}{\partial y}=\frac{1}{\sqrt{1-\left(\frac{\rho_u}{\rho}\right)^2}}\left[\frac{\rho\sin\theta\cos\phi-\rho_u\frac{\partial\rho}{\partial y}}{\rho^2}\right]$$

$$=\frac{\rho}{\sqrt{\rho^2-\rho_u{}^2}}\left[\frac{\rho\sin\theta\cos\phi-\rho_u\frac{\partial\rho}{\partial y}}{\rho^2}\right]$$

$$=\frac{1}{\rho\sqrt{\rho^2-\rho_u{}^2}}\left(\rho\sin\theta\cos\phi-\rho_u\frac{\partial\rho}{\partial y}\right)$$

$$\frac{\partial el}{\partial z}=\frac{1}{\sqrt{1-\left(\frac{\rho_u}{\rho}\right)^2}}\left[\frac{\rho\sin\phi-\rho_u\frac{\partial\rho}{\partial z}}{\rho^2}\right]$$

$$=\frac{\rho}{\sqrt{\rho^2-\rho_u{}^2}}\left[\frac{\rho\sin\phi-\rho_u\frac{\partial\rho}{\partial z}}{\rho^2}\right]$$

$$=\frac{1}{\rho\sqrt{\rho^2-\rho_u{}^2}}\left(\rho\sin\phi-\rho_u\frac{\partial\rho}{\partial z}\right)$$

APPENDIX C

MATLAB RESULTS

C-1 SSN Sensor Results (No Perturbations)

C-1.1 NLS Socorro Sensor Results (No Perturbations)

The following results are for the Socorro sensor debris object orbit without perturbations. The results are shown for a batch of 24 and 240 measurements.

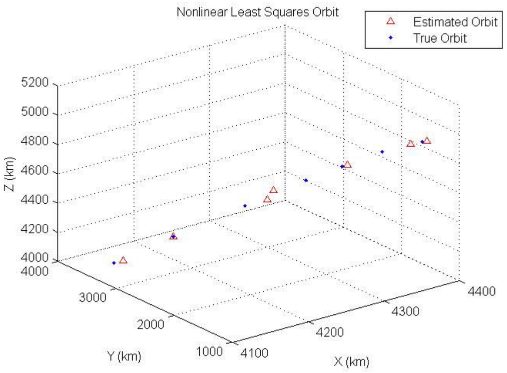


Figure C.1: NLS Socorro Debris Orbit (No Perturbations Batch 24)

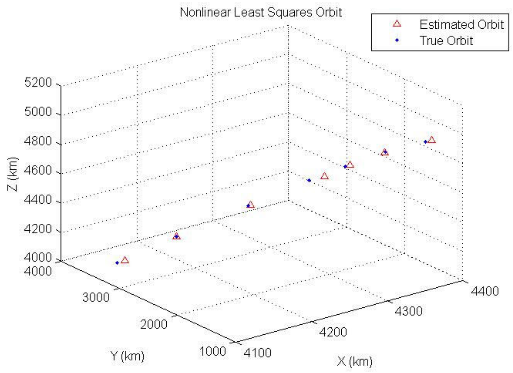


Figure C.2: NLS Socorro Debris Orbit (No Perturbations Batch 240)

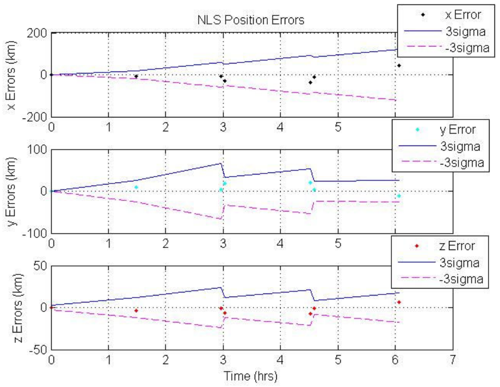


Figure C.3: NLS Socorro Position Errors (No Perturbations Batch 24)

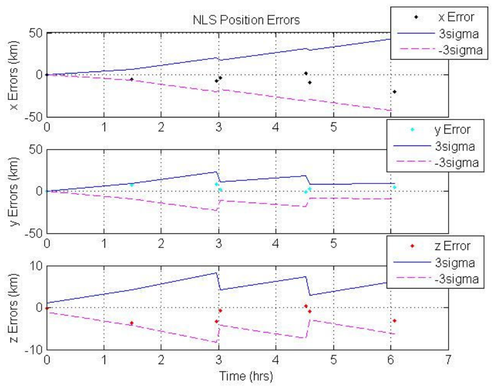


Figure C.4: NLS Socorro Position Errors (No Perturbations Batch 240)

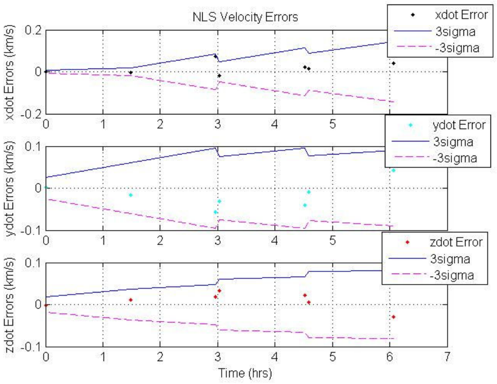


Figure C.5: NLS Socorro Velocity Errors (No Perturbations Batch 24)

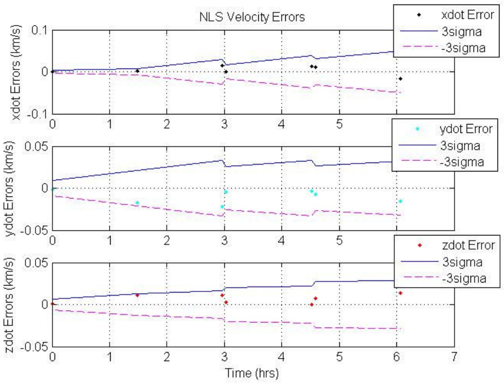


Figure C.6: NLS Socorro Velocity Errors (No Perturbations Batch 240)

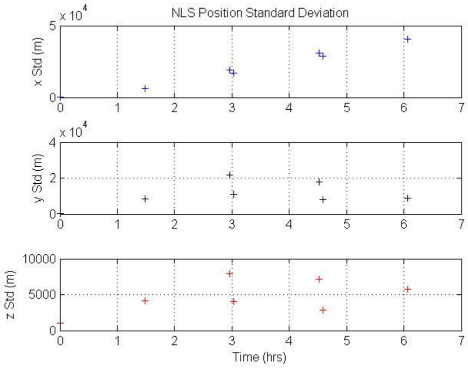


Figure C.7: NLS Socorro Position Standard Deviation (No Perturbations Batch 24)

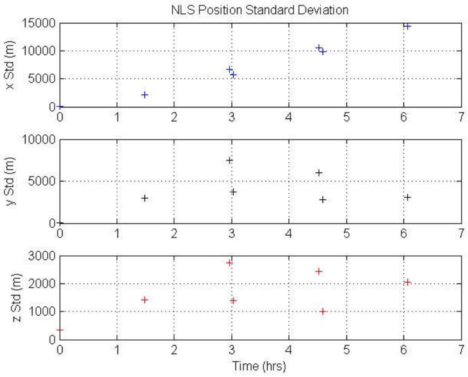


Figure C.8: NLS Socorro Position Standard Deviation (No Perturbations Batch 240)

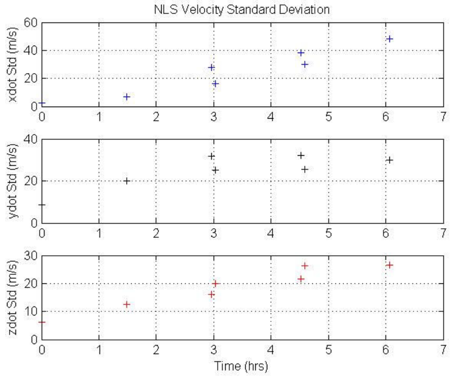


Figure C.9: NLS Socorro Velocity Standard Deviation (No Perturbations Batch 24)

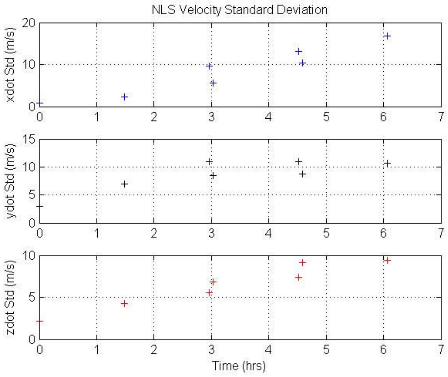


Figure C.10: NLS Socorro Velocity Standard Deviation (No Perturbations Batch 240)

C-1.2 EKF Socorro Sensor Results (No Perturbations)

The following results are for the Socorro sensor debris object orbit without perturbations. The results are shown for a batch of 24 and 240 measurements.

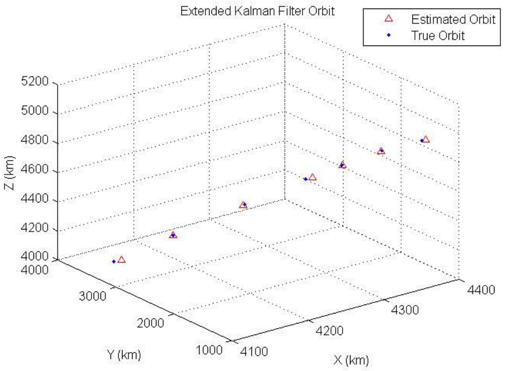


Figure C.11: EKF Socorro Debris Orbit (No Perturbations Batch 24)

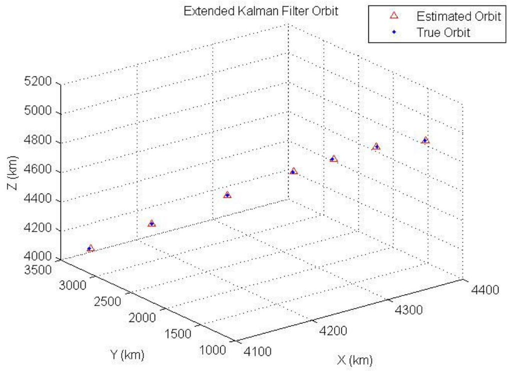


Figure C.12: EKF Socorro Debris Orbit (No Perturbations Batch 240)

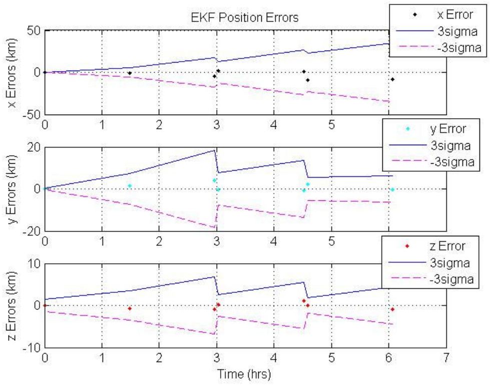


Figure C.13: EKF Socorro Position Errors (Perturbations Batch 24)

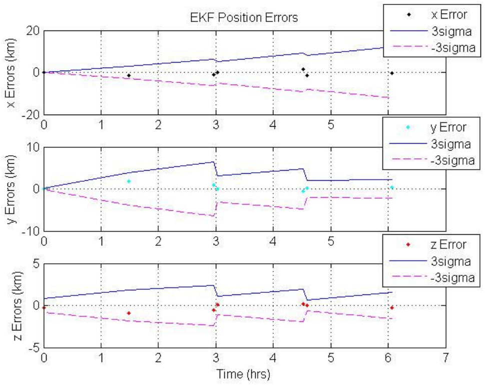


Figure C.14: EKF Socorro Position Errors (Perturbations Batch 240)

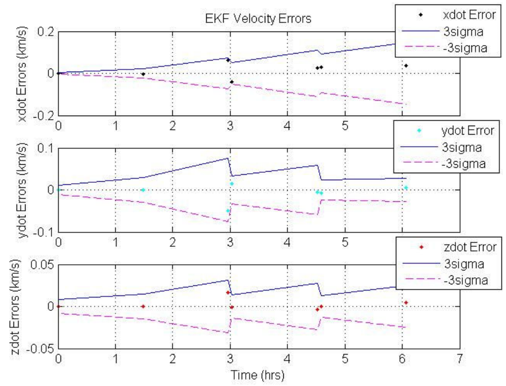


Figure C.15: EKF Socorro Velocity Errors (No Perturbations Batch 24)

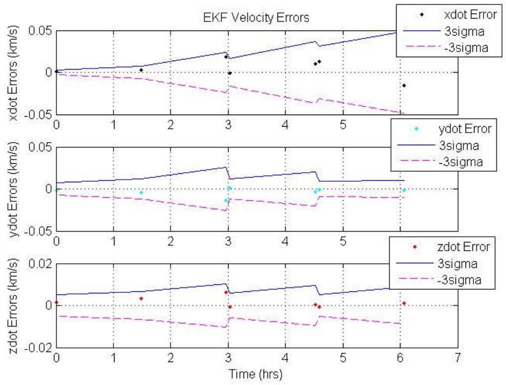


Figure C.16: EKF Socorro Velocity Errors (No Perturbations Batch 240)

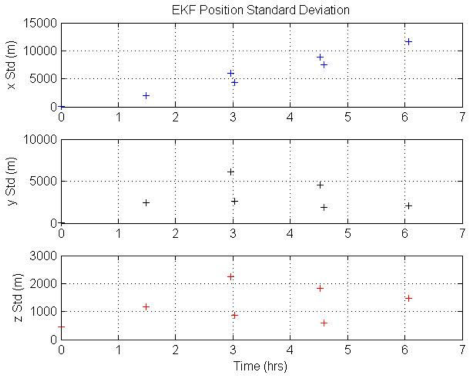


Figure C.17: EKF Socorro Position Standard Deviation (No Perturbations Batch 24)

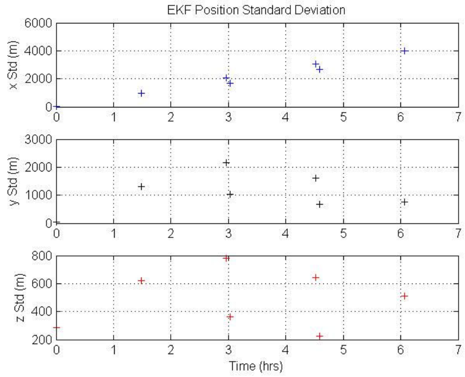


Figure C.18: EKF Socorro Position Standard Deviation (No Perturbations Batch 240)

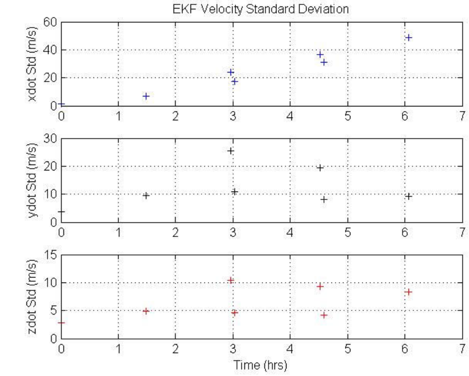


Figure C.19: EKF Socorro Velocity Standard Deviation (No Perturbations Batch 24)

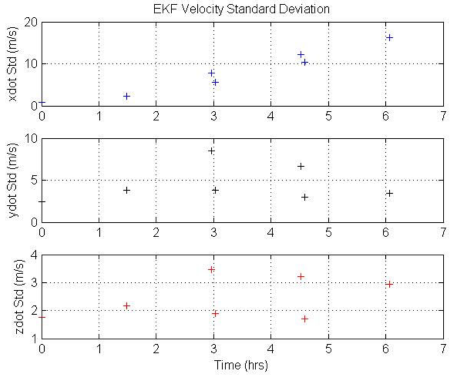


Figure C.20: EKF Socorro Velocity Standard Deviation (No Perturbations Batch 240)

C-2 SSN Sensor Results (Perturbations)

C-2.1 NLS Socorro Sensor Results (Perturbations)

The following results are for the Socorro sensor debris object orbit with perturbations. The results are shown for a batch of 24 and 240 measurements.

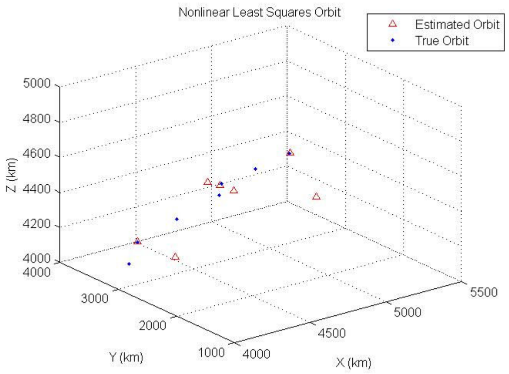


Figure C.21: NLS Socorro Debris Orbit (Perturbations Batch 24)

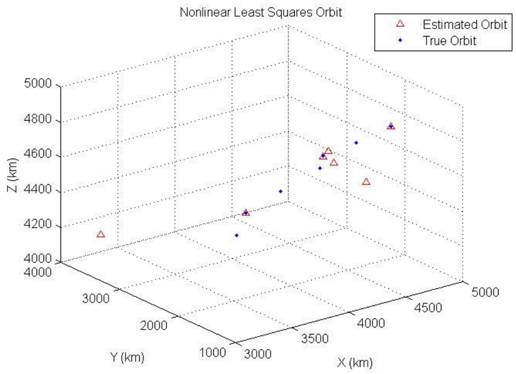


Figure C.22: NLS Socorro Debris Orbit (Perturbations Batch 240)

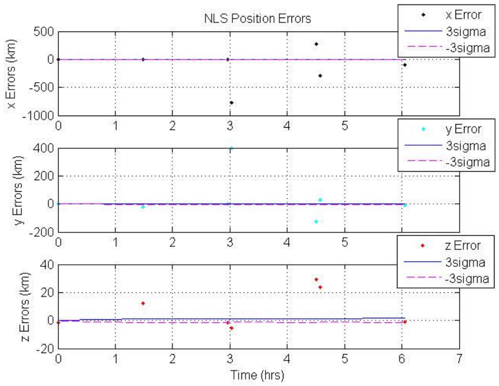


Figure C.23: NLS Socorro Position Errors (Perturbations Batch 24)

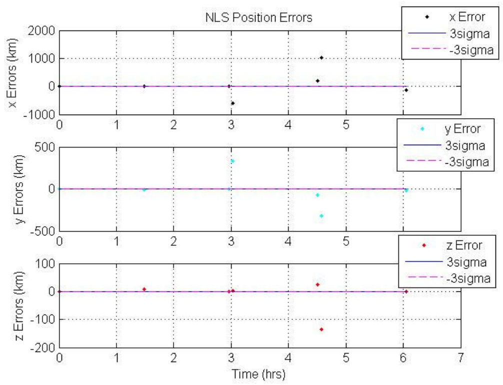


Figure C.24: NLS Socorro Position Errors (Perturbations Batch 240)

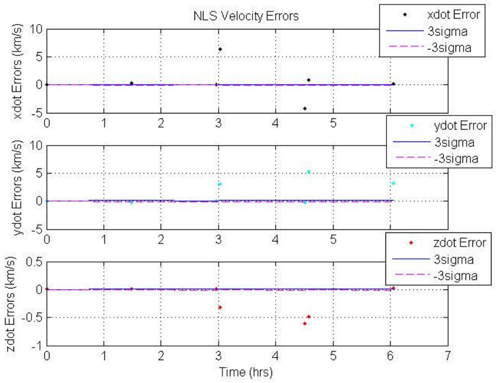


Figure C.25: NLS Socorro Velocity Errors (Perturbations Batch 24)

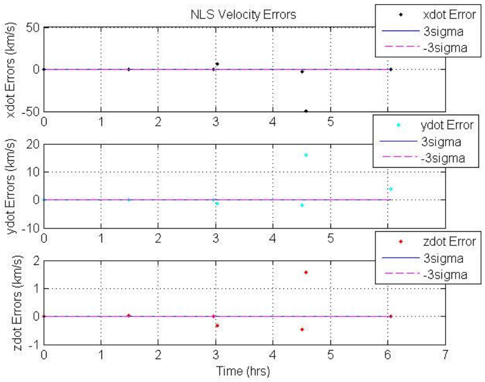


Figure C.26: NLS Socorro Velocity Errors (Perturbations Batch 240)

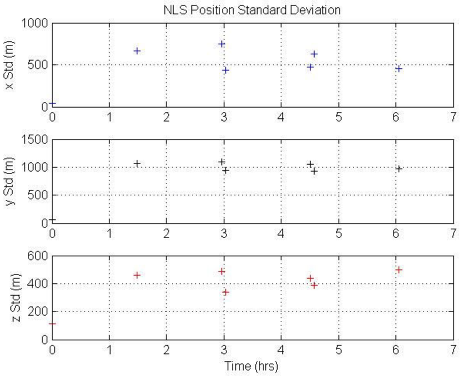


Figure C.27: NLS Socorro Position Standard Deviation (Perturbations Batch 24)

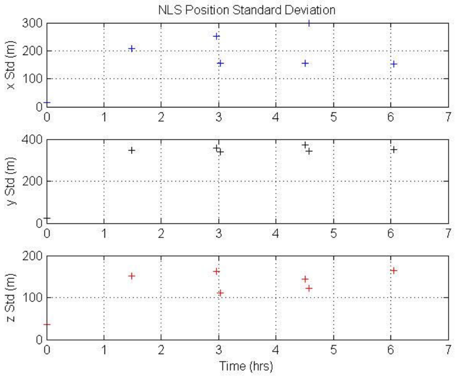


Figure C.28: NLS Socorro Position Standard Deviation (Perturbations Batch 240)

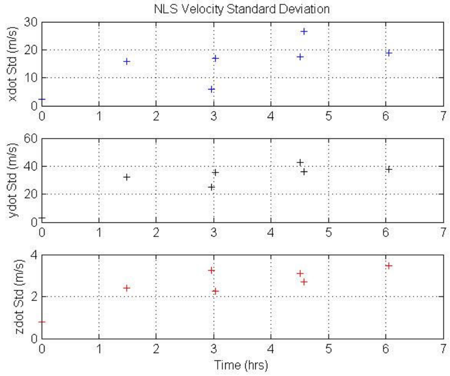


Figure C.29: NLS Socorro Velocity Standard Deviation (Perturbations Batch 24)

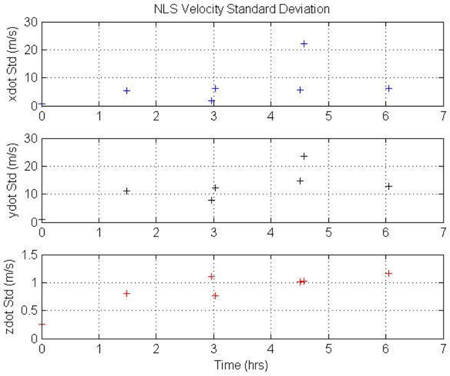


Figure C.30: NLS Socorro Velocity Standard Deviation (Perturbations Batch 240)

C-2.2 EKF Socorro Sensor Results (Perturbations)

The following results are for the Socorro sensor debris object orbit with perturbations. The results are shown for a batch of 24 and 240 measurements.

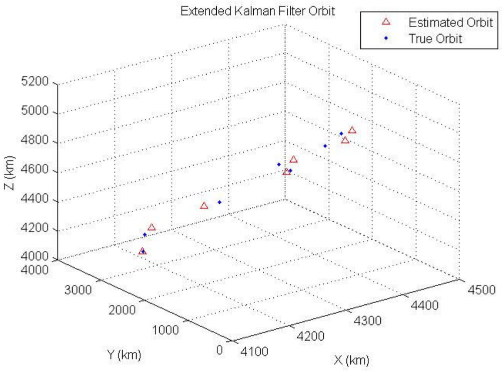


Figure C.31: EKF Socorro Debris Orbit (Perturbations Batch 24)

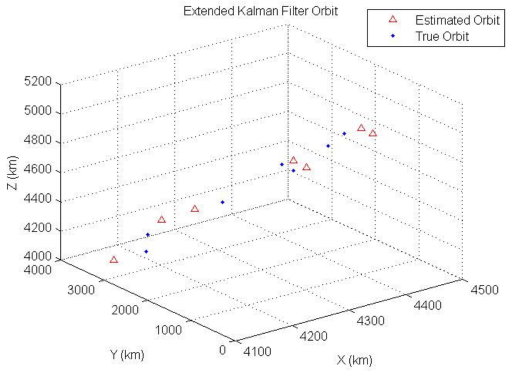


Figure C.32: EKF Socorro Debris Orbit (Perturbations Batch 240)

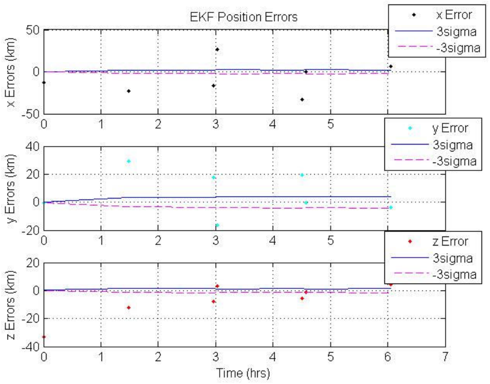


Figure C.33: EKF Socorro Position Errors (Perturbations Batch 24)

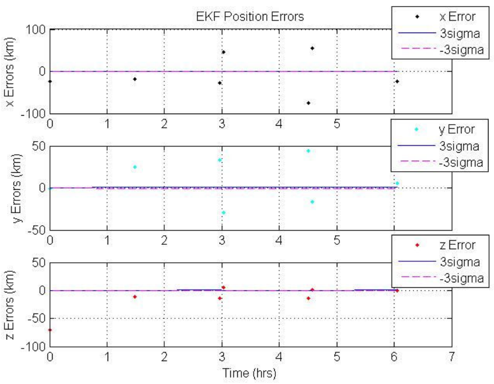


Figure C.34: EKF Socorro Position Errors (Perturbations Batch 240)

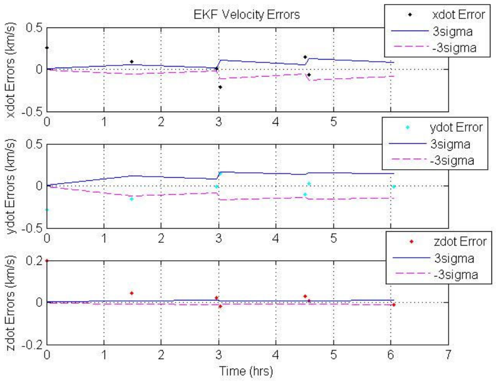


Figure C.35: EKF Socorro Velocity Errors (Perturbations Batch 24)

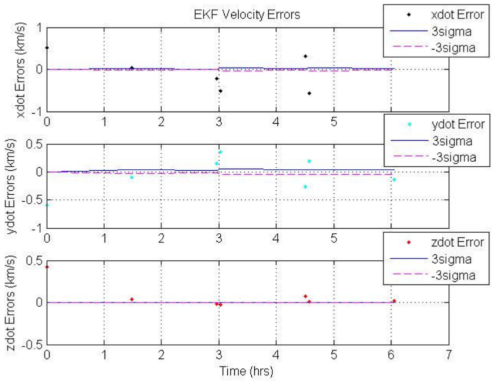


Figure C.36: EKF Socorro Velocity Errors (Perturbations Batch 240)

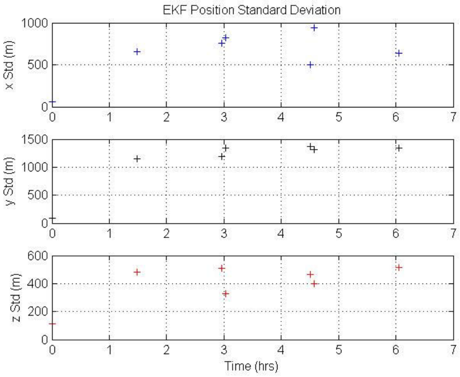


Figure C.37: EKF Socorro Position Standard Deviation (Perturbations Batch 24)

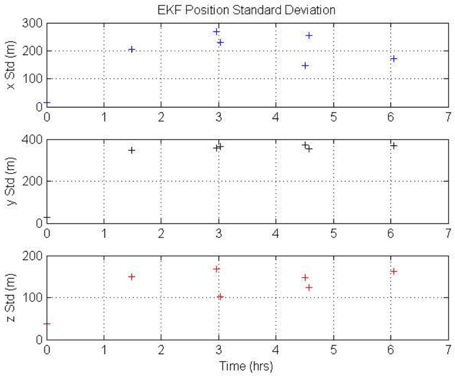


Figure C.38: EKF Socorro Position Standard Deviation (Perturbations Batch 240)

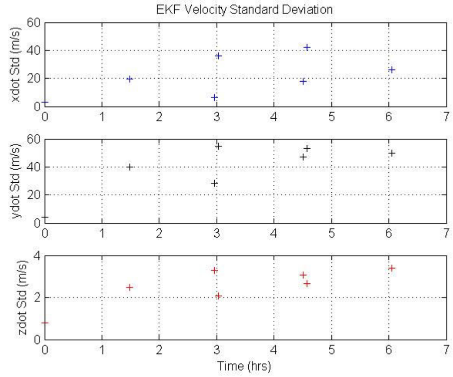


Figure C.39: EKF Socorro Velocity Standard Deviation (Perturbations Batch 24)

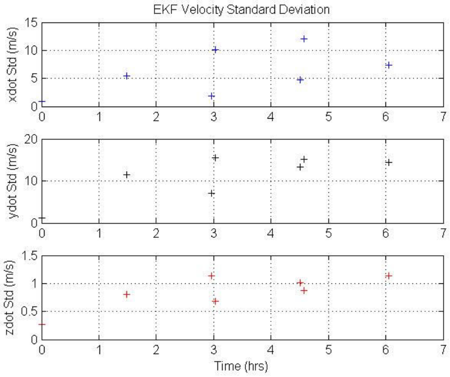


Figure C.40: EKF Socorro Velocity Standard Deviation (Perturbations Batch 240)

REFERENCES

[1] United Nations. "Technical Report on Space Debris." A/AC.105/720. ISBN 92-1-100813-1. New York, 1999. <http://www.unoosa.org/pdf/reports/ac105/AC105_720E.pdf>.

[2] Grahn, Sven. "The US-A program (Radar Ocean Reconnaissance Satellites - RORSAT) and radio observations thereof." Sven's Space Place: Space Tracking Notes. 1996. <http://www.svengrahn.pp.se/trackind/RORSAT/RORSAT.html>.

[3] NASA. "Orbital Debris and Micrometeoroids" 9 September 2002. <http://www.wstf.nasa.gov/Hazard/Hyper/debris.htm>.

[4] Wilkins, Matthew. P., "Characterizing Orbit Uncertainty Due To Atmospheric Uncertainty." *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, AIAA 2000-3931, Denver, CO, August 14-17, 2000.

[5] NationalGeographic.com. "Space Junk Cleanup Needed, NASA Experts Warn".19962006. <http://news.nationalgeographic.com/news/bigphotos/11534070.html>.

[6] Foust, Jeff**. "**The Dangers of Orbital Debris". March 1997. <http://see.msfc.nasa.gov/sparkman/Section_Docs/article_1.htm>.

[7] "Orbital Debris Frequently Asked Questions" 29 April 2005. <http://www.orbitaldebris.jsc.nasa.gov/faqs.html>.

[8] Paul, Vergez L. "Lyapunov Stability Analysis of an Orbit Determination Problem". *The Journal of the Astronautical Sciences*, Vol. 45, No. 2, April-June 1997, pp. 233-245.

[9] Vergez, Paul., Sauter, Luke., and Dahlke, Scott. "An Improved Kalman Filter for Satellite Orbit Predictions". *The Journal of the Astronautical Sciences*, Vol. 52, No. 3, July-September 2004, pp. 359-380.

[10] Julier, simon J., and Uhlmann, Jeffery K. "A New Extension of the Kalman Filter to Nonlinear Systems." The Robotics Research Group, Department of Engineering Science, The University of Oxford. 1997. <http://www.cs.unc.edu/~welch/kalman/media/pdf/Julier1997_SPIE_KF.pdf>.

[11] Vallado, David A. "A Future Look at Space Surveillance and Operations". *Space Surveillance Workshop*, -98-,Washington, DC, October 20-23, 1998. <http://www.centerforspace.com/downloads/files/pubs/USNO_1998_A_future_Look_at_Space_Surveillance_and_Operations.pdf>.

[12] Vallado, David A., and Carter, Scott S. “Accurate Orbit Determination from Short-Arc Dense Observational Data”. *The Journal of the Astronautical Sciences*, Vol. 46, No. 2, April-June 1998, pp. 195-213.

[13] NASA. “Probability Of A Failing Tether”. <http://media.nasaexplores.com/lessons/03-054/9-12_2.pdf>.

[14] European Space Agency. “Space Debris: assessing the risk”. 21 March 2005. <http://www.esa.int/esaCP/SEMH5KRMD6E_index_2.html>.

[15] “Frequently Asked Questions: International Space Station” 14 November 2005. <http://spaceflight.nasa.gov/cgi-bin/comment.cgi>.

[16] “ISS TRAJECTORY DATA” 2 January 2004. <http://www.spaceflight.nasa.gov/realdata/sightings/SSapplications/Post/JavaSSOP/orbit/ISS/SVPOST.html>.

[17] Lunde, Alfred, and Foster, James Lee. “A 160-Day Simulation of Space Station Debris Avoidance Operations with the United States Space Command (USSPACECOM).” NASA/TM-2005-213166. September 2005.

[18] Vallado, David A., Fundamentals of Astrodynamics and Applications. El Segundo: Microcosm Press, 2004.

[19] Air University. Space Primer. Chapter 21. “Space Surveillance Theory and Network.” September 2002. <http://www.au.af.mil/au/awc/space/primer/space_surveillance_network.pdf>.

[20] National Security Space Road Map. “Dedicated Sensors”. 12 July 1999. <http://www.fas.org/spp/military/program/nssrm/categories/scsssen.htm>.

[21] National Security Space Road Map. “Collateral Sensors”. 12 July 1999. <http://www.fas.org/spp/military/program/nssrm/categories/collsen.htm>.

[22] National Security Space Road Map. “Contributing Sensors”. 12 July 1999. <http://www.fas.org/spp/military/program/nssrm/categories/scssss.htm>.

[23] Space Handbook. “Chapter 3: Space Support to the War Fighters”. <http://cryptome.quintessenz.at/mirror/sh/sh3.htm>.

[24] Answers.com. “Phased Array”. 2006. <http://www.answers.com/topic/phased-array>.

[25] FAS Space Policy Project. “ARMY SPACE REFERENCE TEXT: Section VI: SPACE SURVEILLANCE”. 10 January 1999. <http://www.fas.org/spp/military/docops/army/ref_text/chap07f.htm>.

[26] Air University. Space Primer. Chapter 3. “Space Operations and Tactical Application-U.S. Navy.” 24 July 2003. <http://space.au.af.mil/primer/navy_space.pdf>.

[27] Kneisel, Tom. “Detecting Radar Signals from Satellites Sites Illuminated by NAVSPASUR (Naval Space Surveillance System). 21 October 1998. <http://www.k4gfg.us/navspasur/navspasur5.html>.

[28] United States Air Force Scientific Advisory Board. “Report on: Space Surveillance Asteroids and Comets, and Space Debris.” Vol. 1: Space Surveillance, SAB-TR-96-04, June 1997. <http://www.fas.org/spp/military/docops/usaf/sab-tr-96-04-v1.pdf>.

[29] Kelso, Dr. T.S. “Orbital Coordinate Systems, Part I” 16 December 2000. <http://celestrak.com/columns/v02n01/>.

[30] Wikipedia. “Meridian (geography)” 23 June 2006. <http://en.wikipedia.org/wiki/Meridian_%28geography%29>.

[31] AU Space Primer. “Chapter 8: Orbital Mechanics”. 23 July 2003. <http://space.au.af.mil/primer/orbital_mechanics.pdf>.

[32] Malone, Jerry. The Free Dictionary. “Prime Meridian”. 2005. <http://www.thefreedictionary.com/_/viewer.aspx?path=hm&name=A4meridi>

[33] Kelso, Dr. T.S. “Orbital Coordinate Systems, Part II” 16 December 2000. <http://celestrak.com/columns/v02n02/>.

[34] Dennis D. McCarthy. “Astronomical Time.” Proceedings of the IEEE, Vol. 79, No. 7, July 1991, pp. 915-920.< http://www.cl.cam.ac.uk/~mgk25/volatile/astronomical-time.pdf>.

[35] Crassidis, John L., and John L. Junkins. Optimal Estimation of Dynamic Systems. Boca Raton: CRC Press LLC, 2004.

[36] Answers.com. “GEOID”. 2006. <http://www.answers.com/main/ntquery;jsessionid=5ldma0a7ic2ak?tname=geoid&sbid=lc08a>.

[37] Long, Anne C. et al. 1989. “Goddard Trajectory Determination System (GTDS) Mathematical theory (Revision 1).” FDD/522-89/001 and CSC/TR-89/001. Goddard Space Flight Center: National Aeronautics and Space Administration. <http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19910072959_1991072959.pdf>.

[38] “Program NPOE: Numerical Prediction of Orbital Events” 1995-2004. <http://www.cdeagle.com/npoe/npoe.pdf>.

[39] “Spherical harmonic coefficients of EGM96 (including the standard deviations)” 24 February 2002. <ftp://cddis.gsfc.nasa.gov/pub/egm96/general_info/egm96_to360.ascii>.

[40] J. B. Calvert. “Spherical Harmonics”. 6 July 2000. <http://www.du.edu/~jcalvert/math/harmonic/harmonic.htm>.

[41] Kelso, Dr. T.S. “Orbital Coordinate Systems, Part I” 16 December 2000. <http://celestrak.com/columns/v02n01/>.

[42] Natural Resources Canada. “Canadian Spatial Reference System Glossary”. 23 September 2005. <http://www.commlinx.com.au/gps_glossary.htm>.

[43] NASA Langley Research Center . <http://asd-www.larc.nasa.gov/SCOOL/vz.sz.gif>.

[44] Heavens Above. “Azimuth”. <http://www.heavens-above.com/gloss.asp?term=azimuth>.

[45] Science Software. “Orbital Mechanic with MATLAB: Astrodynamic Coordinates”. <http://www.cdeagle.com/ommatlab/coordinates.pdf>.

[46] Wikipedia. “Ballistic Missile Early Warning System”. 13 July 2006. <http://en.wikipedia.org/wiki/Ballistic_Missile_Early_Warning_System>.

[47] Dr. Richard A. Albanese. **“**PAVE PAWS: Still a concern”. *Cape Cod Today.* 9 February 2006.<http://www.capecodtoday.com/blogs/index.php/headlines/2006/02/09/pave_paws_large_phased_array_radar_syste>.

[48] California State Military Department. “Beale Air Force Base”. <http://www.militarymuseum.org/Beale.html>.

[49] Satnam Alag, Kai Goebel, and Alice Agogino. “A Methodology for Intelligent Sensor Validation and Fusion used in Tracking and Avoidance of Objects for Automated Vehicles”. *University of California at Berkeley*. 12 May 1995. <http://best.me.berkeley.edu/~goebel/path.dir/ACC.dir/ACC.final.html>.

[50] PlanetMath.org. “Kronecker Product”. 11 October 2004. <http://planetmath.org/encyclopedia/KroneckerProduct.html>.

[51] PlanetMath.org. “Partitioned Matrix”. 7 July 2006. <http://planetmath.org/encyclopedia/BlockMatrix.html>.

[52] Schaub, Hanspeter, and John L. Junkins. “Kalman Filter Study of A Micro-Robot With Track Slippage.” Tech. Rep., Texas A&M University, 2003.

[53] “The Kronecker Tensor Product” 1994-2006. <http://www.mathworks.com/access/helpdesk/help/techdoc/math/mat_linalg9.html>.

BIOGRAPHICAL INFORMATION

Jolanta Matusewicz began her studies in Mechanical Engineering at the University of Toronto in Ontario, Canada. Her interests in the space industry and specifically the field of orbital mechanics prompted her to pursue a degree in Aerospace Engineering. Jolanta Matusewicz earned a Bachelor of Science in Aerospace Engineering from the University of Texas at Arlington in May of 2004. She will earn a Master of Science in Aerospace Engineering from the University of Texas at Arlington in August of 2006. She will then pursue a Ph.D. in Aerospace Engineering at the University of Texas at Arlington. Her research interest is in the orbit determination problem as it relates to asteroids and comet trajectories. Following the Ph.D program, Jolanta hopes to pursue fulltime research in her field of interest in the space industry or at a related university.