Orbit Determination for a Microsatellite Rendezvous with a Non-Cooperative Target

Brian L. Foster

ORBIT DETERMINATION FOR A MICROSATELLITE RENDEZVOUS WITH A NON-COOPERATIVE TARGET

THESIS

Brian L. Foster, Captain, USAF

AFIT/GAI/ENY/03-2

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAI/ENY/03-2

ORBIT DETERMINATION FOR A MICROSATELLITE RENDEZVOUS WITH A NON-COOPERATIVE TARGET

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Aerospace and Information Operations

Brian L. Foster, B.S. AsE.

Captain, USAF

March 2003

AFIT/GAI/ENY/03-2

ORBIT DETERMINATION FOR A MICROSATELLITE RENDEZVOUS WITH A NON-COOPERATIVE TARGET

Brian L. Foster, B.S.AsE.

Captain, USAF

Approved:

Table of Contents

Page

Page

List of Tables

AFIT/GAI/ENY/03-2

Abstract

This study investigated the minimum requirements to establish a satellite tracking system architecture for a hostile "parasitic microsatellite" to rendezvous with a larger, non-cooperative target satellite. Four types of tracking systems and their capabilities were reviewed with emphasis on "low-technology" level and/or mobile systems which could be used by technologically unsophisticated state or non-state adversaries. With the tracking system architecture selected, simulated tracking data was processed with a non-linear least squares orbit determination filter to determine and/or update the target satellite's state vector.

ORBIT DETERMINATION FOR A MICROSATELLITE RENDEZVOUS WITH A NON-COOPERATIVE TARGET

I. Introduction

1.1 Background Information

Since the end of the 1991 Persian Gulf War, which has been called the ‘first space war,’ the United States has become increasingly dependent on products and services derived from space borne assets, both economically and militarily. In view of this increased dependency, the 2000 Commission to Assess United States National Security Space Management and Organization recognized in their January 11, 2001 report “The political, economic, and military value of space systems makes them attractive targets for state and non-state actors hostile to the United States and its interests.” (Space Commission, 12) China is one such potentially hostile state actor. According to a report in the Hong Kong *Sing Tao* newspaper dated January 5, 2001, “The Small Satellite Institute under the Research Institute of Space Technology has developed an advanced anti-satellite weapon called ‘parasitic satellite’.” (Tung) The article further reports

> “the ‘parasitic satellite’ is a microsatellite which can be launched to stick to an enemy satellite; and in time of war, it will jam or destroy the enemy satellite according to the command it receives. As a new-concept anti-satellite weapon, ‘parasitic satellite’ can control or attack many types of satellite, including low-orbit, medium-orbit and high-orbit satellites, both military and civilian satellites, single satellite, and constellated satellites. An enemy satellite, once locked on by ‘parasitic satellite,’ cannot escape being paralyzed or destroyed instantaneously in time of war, no matter how sophisticated it is, and no matter whether it is a communications satellite, radar electronics jamming satellite, or even a space station or space-based laser gun.” (Tung)

In the concluding paragraph, the article states “Its [Beijing’s] long-term strategic objective is to establish a strategic balance among big powers, break the space monopoly by the superpower’s huge astronautical system, and weaken the superpower’s information warfare capability.” (Tung)

While the parasitic satellite report may itself be an example of information warfare whereby an adversary attempts to misinform or deceive potential adversaries, a more substantive report appearing on the SPACE.com website on October 19, 2000 details Tsinghua-1, China’s first microsatellite. According to the report, Tsinghua-1 was a joint project of Tsinghua University in Beijing and Surrey Satellite Technology Ltd. (SSTL) of Guildford, United Kingdom. Tsinghua-1 was one of three small satellites launched by a Kosmos 3M booster on June 28, 2000. A key paragraph of this article states

> “While the intent of the microsat project is purely scientific in nature, its capabilities have not been lost on military experts…And the satellite also has demonstrated the ability to maneuver and station-keep with neighboring spacecraft…” (Seitzen)

This ability of small satellites to carry out automated space rendezvous and observation of other satellites was demonstrated shortly after launch when the 6.5-kg British SNAP-1 *nano*-satellite, also built by SSTL and launched with Tsinghua-1, made the first-ever space rendezvous of microsats, closing to a range of just 30 feet (9 meters) (Seitzen). It should be noted that any potentially hostile satellite rendezvous missions will not be so easily set up for success. In this case, the three small satellites were all deployed by the same booster into roughly the same orbital conditions and were no more than a few hundred meters apart when the rendezvous was performed. For the case of a hostile microsatellite mission, the aggressor will have to first determine which of the satellites already on orbit, whether for days or years, is to be targeted then launch the microsatellite to intercept the target. Since by its very nature (i.e. micro-sized), the microsatellite will not possess an extensive propulsion system or on-board propellant supply, it is critical the microsatellite be directly launched as closely as possible into the target satellite's orbital plane. Out-of-plane, or inclination changing, maneuvers are extremely costly in terms of propellant for any satellite, regardless of size. In-plane maneuvers, on the other hand, are relatively inexpensive in terms of propellant.

Another potentially threatening implication for United States' space systems is the possibility that "Small, microsat satellites used in future reconnaissance roles could be quickly built and launched aboard Chinese space boosters in a "pop-up" capability as needed for military assignments." (Seitzen) Although, first mentioned in the October 19, 2000 article above, further reference to a launch-on-demand system was publicly made at a space symposium held in Shanghai on 17-20 April 2001. According to the article "China Plans Rapid-Response, Mobile Rocket, Nanosatellite Next Year," which appeared on the SpaceDaily website on May 1, 2001, Chinese speakers discussed "the need for 300-500 kilogram-class satellites to be put in orbit within hours upon request from a customer…along with scientific, economic, and national security needs." To meet this requirement, the Chinese engineers and scientists envision a mobile, truck-based platform that would be capable of launching from "anywhere in the country." The article further quoted Yin Xingliang, vice president of a Chinese company called CAMEC, regarding the mobile launch system, "the tracking, telemetry, and command (TT&C) method and the TT&C system must conform to features of mobile launch." (Cosyn)

For this thesis, the initial conditions to be tested for rendezvous placed the microsatellite in the same orbit as the target but trailing the target by 1,000 km. The first rendezvous maneuver control thrust calculations will be based on orbit positions determined by ground sensors. When the microsatellite is within range of the target to track it with its on-board sensor, then the control thrust calculations will be based on orbit positions based on those observations.

1.2 Problem Description/Objectives

The fundamental issue to be investigated regarding the Chinese "parasitic satellite" was the overall feasibility of such a system. To that end, the work related to this topic was divided among three students in the Air Force Institute of Technology Department of Aeronautics and Astronautics (AFIT/ENY) 03M class. The areas of responsibility included selecting a tracking and orbit determination architecture for both the hostile microsatellite and the larger target satellite; establishing a rendezvous control algorithm; and modeling the larger target satellite's dynamics for detection of a covert microsatellite docking.

The objective of this thesis was to develop a tracking system architecture concept and a set of orbit determination routines for three different tracking phases for both the microsatellite and the target satellite. These phases include: 1) initial orbit determination such as following the launch of the microsatellite at the beginning of its rendezvous mission or the activation of a new tracking sensor that has no *a priori* knowledge of the target's state (orbital elements); 2) orbit determination (orbital element update) from ground sensor data using an initial estimate of the target's state to start the orbit determination filter; and finally, 3) determination of the target satellite's orbit from the perspective of the microsatellite's space-borne sensor using an initial estimate of the target's state based on the orbit determination from the ground sensor(s) to start the on-orbit determination filter.

The initial orbit determination phase utilizes methods developed by Gibbs and Herrick (Vallado: 414, 420). The Gibbs Method uses three sequential, non-zero, coplanar position vectors to determine the velocity associated with the second (middle) position vector. Thus, having the three components for position vector and the three components for the velocity vector give the six total quantities needed to define the satellite's state. Vallado (1998) offers two warnings when implementing the Gibbs Method. First, although the problem formulation assumes the vectors are coplanar, real world data may produce position vectors that are slightly out of plane. Therefore, the user must choose an error tolerance level when checking whether the vectors are coplanar. Vallado suggests a tolerance of 2^o to 3^o (Vallado, 410). Second, even if the position vectors are coplanar, the Gibbs Method will suffer numerical instability if they are too closely spaced together along the orbital path. Vallado states the Gibbs Method is robust and works with angles separated by as little as 1^o, but degrades quickly with smaller angles (Vallado, 413).

The Herrick-Gibbs Method (Vallado, 420) is a variation of the basic Gibbs Method which uses a Taylor-series approximation to obtain the velocity vector associated with the second of three sequential position vectors. Whereas the Gibbs Method becomes unstable when the three position vectors are closely spaced in-plane, the Herrick-Gibbs Method is better suited for such conditions. Regarding the suitability of the Gibbs versus

Herrick-Gibbs methods, Vallado says Herrick-Gibbs is superior below angular separations of 1^{o} while Gibbs is superior with angular separations over 5^{o} (Vallado, 421).

The second orbit determination phase is the updating of the target satellite's orbital elements using observations from a ground-based sensor and implementing a non-linear least squares orbit determination filter. The non-linear least squares filter was modeled after FORTRAN code developed by Dr. William Wiesel for use in his class MECH 731 Modern Methods of Orbit Determination at the Air Force Institute of Technology. His original FORTRAN code was set up for orbit determination of a spacecraft on an interplanetary trajectory to rendezvous with Mars and included perturbation modeling for the third-body gravitational effects of the sun and the moon. The FORTRAN code was translated by the author, with Dr. Wiesel's permission, to MATLAB ® and updated to include perturbation modeling for atmospheric drag for orbits below 1,000 km altitude and the gravitational effect of the Earth's oblateness. The filter is *not* self-starting and must use an *a priori* estimate of the target satellite's state to begin calculations. The initial estimate of the target's state could be the initial orbit as determined using the Gibbs or Herrick-Gibbs Methods results of the first phase or North American Aerospace Defense Command (NORAD) two-line element (TLE) sets obtained through other sources such as amateur satellite tracking bulletin boards on the Internet such as Celestrak (http://celestrak.com).

The third orbit determination phase is the updating of the target satellite's orbit elements using observations from a space-borne sensor on-board the chase satellite, which is assumed to be the microsatellite, and involves implementing a non-linear least squares orbit determination filter. In this case, the initial estimate of the target satellite's state is the orbit as determined from ground sensor(s) in phase two.

The overarching approach used for this thesis was to go as "low-tech" as possible in the development of the tracking architecture and orbit determination routines. However, space missions do require a substantial investment in terms of hardware such as the satellites and tracking systems, engineers and technicians knowledgeable in space-systems engineering and integration, and perhaps, most importantly, funding. The rationale behind this "low-tech" approach was to determine if a relatively unsophisticated potential adversary such as a terrorist group or developing nation or state could reasonably pose a threat to satellites in orbit.

II. Tracking Systems Architecture Background Information

An open-source literature review was conducted to investigate the types of ground-based satellite tracking systems available to a potential adversary and the capabilities of those systems. Emphasis was placed on identifying foreign systems, whether they were permanently fixed tracking sites or portable systems which might be more favorable to a terrorist-type organization; however, open source literature was determined to be extremely lacking. Even the most authoritative open source, *Jane's Radar and Electronic Warfare Systems*, had few entries on space tracking systems of origin other than Russia or the United States. Where possible, the capabilities of foreign systems are described, but the discussion defaults to describing US systems in order to establish a baseline reference for the type of system being reviewed. The inference is that if the United States, which the author assumes has the most well established combination of tracking systems, has technical difficulty with certain systems, then an adversary with less technical capability or resources will have even greater difficulty. The most likely candidate space tracking systems include radar, Global Positioning System (GPS), satellite laser ranging (SLR), and optical tracking. Each of these systems is discussed in the following sections. The literature review also searched for information on microsatellite space borne tracking systems specifically for rendezvous and docking of *non-cooperating* vehicles. Again, open source literature was found to be non-existent.

2.1 Radar

Radar is the most likely satellite tracking system to be used by an adversary. The main advantages are its ability to deliver accurate range (distance from the radar to the satellite) information, its 24-hour availability (day and night), and its ability to penetrate weather such as clouds and rain. Although orbit determination methods which use angles-only (observations of azimuth and elevation or right ascension and declination) do exist, Vallado states that "range information allows us to analyze data faster, more simply, and more accurately." (Vallado, 379)

The primary disadvantage to using radar is that the adversary is typically bound to the radar site's geographic location and thus may not be able to track all targets of interest to the adversary depending on the mission orbits of the desired targets. To counter this situation, an adversary would need to operate a worldwide tracking system such as the United States Air Force's Space Surveillance Network (SSN) and/or have mobile radar space track systems. Since no other state or non-state entity possesses a worldwide network, the need for a mobile system becomes obvious if the adversary intends to have the ability to track any desired target.

Only three references on mobile space track radars were found during the literature review. In his background paper for the 2000 Commission to Assess United States National Security Space Management and Organization, "Threats to United States Space Capabilities," author Tom Wilson states

> "The proliferation of air and theater missile defense radars, such as those associated with the SA-10, have enabled many countries, such as China (who purchase these radars from Russia), to field *space-based* tracking systems capable of accurately locating objects in LEO. These mobile radars were originally designed to track reentry vehicles but, due to their

low-cost and mobility, are attractive as *space-based object* trackers as well." (Wilson, 7)

Although Wilson appears to say the SA-10 radars are space-based, he means the *mobile, ground-based* radars are used to track *space-based objects*. Regrettably, Wilson does not provide a reference for his statement regarding the SA-10. According to *Missile Systems of the World,* the radar used with the SA-10A "Grumble" is the 10-GHz 36N6 (NATO Flap Lid) phased-array radar (Missile Systems, 104). There is also a SA-10C/D "Grumble" variant whose associated fire-control radar is the improved three-dimensional Tombstone surveillance radar (Missile Systems, 106). A review of *Jane's Radar and Electronic Warfare Systems 2001-2002* does not show the 36N6 Flap Lid but does list the 30N6 Flap Lid B radar. According to *Jane's*, the 30N6's detection range is only 90 km (Jane's, 96). Similarly, for the Tombstone radar (64N6E), the detection range is only slightly better at 260 km for a target the size of a MiG-21 aircraft (Jane's, 98). Its listed accuracies are 30 minutes of arc in azimuth, 35 minutes of arc in elevation, and 200 meters in range.

Only one other reference for a mobile space tracking system was found. The Chinese HN-C03-M precision instrumentation radar is listed as having a range of 300 km (for a reflecting target of unspecified size). It operates in the G-band (5.5 – 5.7 GHz) with a peak power of 1 megawatt (MW). Its tracking accuracies are 0.2 min (0.00333 deg) in both azimuth and elevation and 5 meters in range (Jane's, 288).

In contrast to mobile radar tracking systems, an example of a foreign fixed-base radar is the Russian Don-2N Anti-Ballistic Missile (ABM) and space vehicle tracking radar. *Jane's* lists its capabilities as full-hemispherical coverage (360^o in azimuth and 90^o

in elevation), detection range of 600 – 1,000 km for a 5-cm space object, and accuracies of 0.02-0.04^{o} angular position and 200 meters in range (Jane's, 37).

If a potential adversary is to threaten all of the mission orbits of US space systems then it must have the capability to track satellites as far as the geosynchronous belt at a range of 22,236 miles (35,786 km). Obviously, the mobile systems and the one Russian fixed-base radar discussed here do not have that capability. An example of a US radar that is capable of ranging to geosynchronous is the AN/FPS-85 Spacetrack radar at Eglin AFB, FL. Built in the 1960s, Spacetrack consists of a single receiver and a single transmitter sitting side-by-side. The receiver face is 192 feet long, 143 feet deep, and 143 feet high. The transmitter face is 126 feet long, 95 feet deep, and 95 feet high (http://www.globalsecurity/org/space/systems/an-fps-85.htm). Spacetrack reportedly has the capability to track an object the size of a basketball, approximately 457 cm^2, at geosynchronous range.

2.2 Global Positioning System (GPS)

Due to the non-cooperative nature of the target satellite, GPS cannot be used for the determination of its orbit in order to pass on to the rendezvous control algorithm for the microsatellite. However, GPS can be used for determining the orbit of the microsatellite. In his paper "Satellite Orbit Determination Using a Single-Channel Global Positioning System Receiver," Mark Psiaki describes the use of single-channel GPS receiver intended as a method of reducing the electrical power required in situations where the power budget is limited as in the case of a micro- or nano-satellite. Typically, a GPS user's position is determined by simultaneously evaluating pseudoranges from a minimum of four different GPS satellites with each satellite representing a separate channel. Some receivers may have as many as 12 channels. Obviously, the more channels a given receiver has, the more electrical power is consumed in processing those channels. The single-channel GPS receiver Psiaki describes processes data from four or more GPS satellites, but does so sequentially. This design trades off power with performance.

In terms of performance, Psiaki states that typical multi-channel receivers could determine instantaneous position with an accuracy on the order of 10 meters up to altitudes of 3,200 km. For his simulated LEO case, the single-channel receiver had peak steady-state errors of 64-m along track, 128-m across track, and 72-m in altitude (Psiaki, 141). By comparison, a 12-channel receiver for this case, had peak errors of 5-m along track, 5-m across track, and 13-m in altitude (Psiaki, 142). Other cases were tested such as a highly elliptical orbit and geosynchronous (GEO). For the GEO case, the peak position error was 7 km. The main cause for error growth for altitudes above 3,200 km is the increasing gaps in the receiver's visibility of GPS satellites with the increase in altitude. This single-channel receiver is mentioned simply as an example of the types of equipment that could be placed on a microsatellite. For the purposes of this thesis, the hostile microsatellite is assumed to be equipped with a suitable multi-channel space Global Positioning System (SGPS) receiver and its position will be considered perfectly known.

2.3 Satellite Laser Ranging (SLR)

Although satellite laser ranging (SLR) is a technique that allows range measurement with an absolute accuracy on the order of $\pm$ 1 cm, the tracked satellite must be specially equipped with retroreflectors, which are sometimes called corner reflectors. The retroreflectors are designed so as to reflect the illuminating laser pulse back to the transmitting source regardless of the angle of incidence on the reflector, thus allowing precise measurements to be made of the returned pulse's phase and round-trip time of flight (NASA Instrument and Sensing Technology: Satellite Laser Technologies webpage). Jon Schwartz, in his paper "Pulse Spreading and Range Correction Analysis for Satellite Laser Ranging" further explains the laser retroreflector array (LRA) with the following

> An LRA is a passive device used as the lidar target for ground-based laser ranging stations. The LRA is composed of a set of retroreflectors precisely located in position and orientation (generally to within 1 mm and 1^{o}, respectively) relative to some fixed point or axis. It is the precision of the location of the cube corner retroreflectors (CCRs) in the LRA that allow ranging measurements to be made to the centimeter level. (Schwartz, 3597)

According to information on the International Laser Ranging Service's website (http://ilrs.gsfc.nasa.gov/), only 75 past or current satellites/space missions have been equipped for laser ranging. The majority of these missions are dedicated to Earth observation and geophysical research; however, interestingly, thirty of these missions could be considered military related. Twenty-eight of these missions are Russian Global Navigation Satellite System (GLONASS) satellites and two are United States Global Positioning System satellites (GPS 35 and GPS 36). Another interesting fact is three of the missions listed are Apollo 11, 14, and 15. These missions left reflector equipment on the surface of the Moon so ranging tests could be performed from stations on Earth. Apparently, power generation at ground-based stations for transmitting the laser pulse great distances is not a limiting factor, as the distance from the Earth to the Moon is 356,400 km.

At the other end of the satellite laser ranging spectrum from large, powerful laser ground stations is a portable system. Engineers have developed the French Transportable Laser Ranging Station (FTLRS) system whose total mass is approximately 300 kg. The optical instrument is a 13-cm diameter telescope installed on a motorized mount. FTLRS can track satellites at altitudes of as much as 3,000 km and is designed to range to the Laser Geodynamic Earth Orientation Satellite (LAGEOS) at 6,000 km in another planned upgrade. The standard error of individual measurements during the first observation campaign were estimated to be on the order of 2-3 cm (Nicolas, 402). The laser is an Nd:YAG with a double-pass amplifier. Its wavelength is 532 nm and its energy is 100-mJ at 1,064 nm. The laser pulse-width is 100 ps. Despite the high precision ranging measurements, one must keep in mind that satellite laser ranging in this manner assumes a cooperative target equipped with retroreflectors and thus a system such as this is not likely to be used by a "low-tech" adversary.

Since the most probable target satellites for the parasitic satellite will not, in general, be equipped with retroreflectors, then if a laser system is to be used for tracking, it will have to be in a more traditional radar mode whereby the laser illuminates the target satellite's skin and produces a return. A quick survey of the United States Air Force's Maui Space Surveillance System (MSSS) shows the state-of-the-art for such a system. In the paper "HI-CLASS on AEOS: A Large Aperture Laser Radar for Space

Surveillance/Situational Awareness Investigations," authors Kovacs, et al, report that the Air Force Research Laboratory's Directed Energy Directorate (AFRL/DE) installed in late 2000, a wideband, 12 Joule, 15-Hz, CO_2 laser radar on the 3.67-meter aperture Advanced Electro-Optics System (AEOS) telescope (Kovacs, 298). MSSS also has the HIgh-Performance CO_2 Ladar Surveillance Sensor (HI-CLASS) on the 0.6-meter aperture Laser Beam Director (LBD). The article further states "the moderate power (~180 watts) HI-CLASS/AEOS system generates multiple, coherent waveforms for precision satellite tracking and characterization of space objects for 1-m^2 targets at ranges out to 10,000 km. This system also will be used to track space objects smaller than 30-cm at ranges to 2,000 km." (Kovacs, 298) Authors Hasson, et al, give more specific HI-CLASS/AEOS performance parameters in their paper "Use of Laser Radar for Small Space Object Experiments." According to them, the HI-CLASS LBD can perform precision 1-m^2 satellite tracking to ranges of 2,000 km with accuracies of $\pm$ 5 m in range and $\pm$ 5 m/s in range rate. HI-CLASS can also track 5-cm^2 objects to 1,000 km. In comparison, the larger AEOS telescope can perform precision 1-m^2 satellite tracking to ranges of 10,000 km with accuracies of $\pm$ 1-3 m in range and $\pm$ 1 m/s in range rate. AEOS can also perform sub-cm^2 object tracking up to 1,000 km (Hasson, 366). To put these performance capabilities in perspective in terms of a microsatellite, the Tsinghua-1's physical parameters were 0.07-m^3 volume with a mass of 50 kg, according to a report posted on the SpaceDaily website by reporter Wei Long July 11, 2000. Assuming a simple cube shape for the Tsinghua-1 satellite, 0.07-m^3 volume translates to a length of 41.21 cm per side or an area of 1,698 cm^2 (0.1698 m^2). The implication of this area is even the powerful AEOS telescope cannot track Tsinghua-1 all the way to 10,000 km.

Montenbruck and Gill point out additional limitations of SLR in their book, *Satellite Orbits: Models, Methods, and Applications,*

> It is noted that laser tracking (other than radar tracking) does not allow auto-tracking of satellites, but depends on the availability of high-precision a priori orbit elements for antenna pointing. Furthermore, the use of SLR for regular tracking is restricted due to its dependence on the weather at the laser stations…(Montenbruck, 203)

2.4 Optical Tracking

Once again, the scarcity of open-source information on tracking systems of foreign countries forces one to look at the capabilities of the United States Air Force. At the large end of the size spectrum for optical tracking systems, the USAF operates the Ground Based Electro-Optical Deep Space Surveillance (GEODSS) system. At the small end of the size spectrum is the Raven automated small telescope system.

The GEODSS system has four operational sites located at Socorro, New Mexico; Maui, Hawaii; Diego Garcia, British Indian Ocean Territories; and Moron, Spain. The main telescope at each of these sites has a 40-inch aperture telescope which has the capability to track an object the size of basketball (457.303 cm^2) at geosynchronous range (http://www.globalsecurity.org/space/systems/geodss.htm).

The Raven automated small telescope system grew out of a program initially set up in 1997 to use small (diameter < 0.5 meters) telescopes to track near Earth asteroids. The extension to tracking man-made satellites was a natural progression. As described by Paul Sydney, et al, “the Raven system is a design paradigm, not a specific configuration of components. Depending on the mission of the particular telescope, the design will be modified using commercial hardware and software, to optimize the configuration for that mission" (Sydney, 237). The design paradigm is to use commercial off-the-shelf (COTS) equipment originally designed for amateur astronomy as the complete Raven system. The Raven at the Maui Space Surveillance System consists of 14.5-inch diameter f/3 Torus Optics Newtonian telescope on a Paramount GT 1100 German Equatorial mount. The imaging device is a charge-coupled device (CCD). Activating the CCD shutter triggers a PC-based GPS receiver and timing card for accurate time tagging of each image. The Raven system is controlled by two computers, one for controlling the telescope and one for data processing. The system also has suite of weather monitoring equipment (Sydney, 238).

In Section 3 of their paper, Autonomous Operations, Sydney, et al, discuss the criteria the Raven control system uses to select satellites for tracking. Criterion number five, rate through the telescope's field-of-view (FOV), may be the limiting factor in using Raven, or a similar system, for tracking low earth orbiting satellites. The Raven control software does not allow tracking of objects whose angular velocity exceeds 45 arcminutes/minute (0.75 degree/minute). This restriction is described as relating to the CCD imaging operation and does not appear to be a physical limitation in terms of telescope slewing rate (Sydney, 238). Even so, this angular velocity limitation prohibits the tracking of satellites below altitudes of approximately 13,930 km according to the following equation from *Space Mission Analysis and Design, 3rd Ed.*:

$$\text{orbital angular velocity (deg/min)} = 2.170415\times10^{6}\,r^{-3/2} \tag{1}$$

where r is the distance from the center of the central body (Earth) to the satellite in kilometers. To determine the altitude, subtract the Earth's radius, 6378.135 km from *r*.

Thus, a Raven-type system would most likely not be used in a microsatellite rendezvous mission below altitudes of 13,930 km. In terms of measurement accuracy, the Raven system “demonstrated the ability to produce topocentric right ascension and declination observations of GEO satellites with RMS errors under two arcseconds (one standard deviation).” (Sydney, 241)

At approximately the same time the Air Force Research Laboratory was developing the Raven system, engineers at the Rocketdyne Division of Boeing North American were experimenting with a slightly smaller telescope with emphasis on portability. In their paper “Description and Experimental Results of a 58-lb Portable LEO Satellite Tracker,” authors Tansey, Campbell, and Koumvakalis outline their use of an 8-inch diameter f/10 telescope on a T-Point mount and controlled by commercial software. This system is reportedly capable of adjustable slew rates to six degrees per second (360 degrees/minute) (Tansey, 78). This means the system can track at all altitudes as the angular velocity at the surface of the Earth is 4.261 degrees/minute. Tansey, et al report “typical tracks at 600 km to 1,000 km are routine with track errors less than 50 μrad [0.00286 deg] peak to valley for the duration of the pass” (Tansey, 83).

2.5 Selection of Tracking System Architecture

2.5.1 Ground Tracking Systems

Having reviewed the candidate tracking systems, their capabilities and their limitations, it is evident that no single system is sufficient to cover all possible mission orbits (LEO, MEO, and GEO). The space tracking systems that have evolved have done so based on those very capabilities and limitations for their type. Thus, those countries or persons involved in satellite tracking must use a ‘system of systems’ in order to cover all orbits. While this does complicate matters in terms of the number of different systems that must be employed, it simplifies matters in that one can choose the simplest system within each type. Thus, for altitudes below approximately 14,000 km, radar is the most likely system to be used with a relatively modest ground station since a suitable mobile system was not found. Laser ranging could also be used for LEO orbits below 10,000 km, but it must be remembered that even the AEOS system could not track Tsinghua-1 all the way to 10,000 km. For orbits higher than 14,000 km, the Raven small telescope can be used.

2.5.2 Microsatellite On-board Tracking Sensor

A literature review was also conducted to find background information on sensors for satellite rendezvous. Again, great difficulty was encountered in trying to find information suitable for a microsatellite rendezvous mission with a non-cooperative target. Several articles were found that described video systems for terminal control; laser ranging between cooperative targets equipped with retroreflectors; relative GPS; and rendezvous radars for larger spacecraft such as the Space Shuttle and the Orbital Maneuvering Vehicle. The *Space-based Radar Handbook* describes both of these systems, but regrettably has no information on microsatellites as its publication date is 1989. For background purposes, the performance of the Space Shuttle’s rendezvous radar and the Orbital Maneuvering Vehicle’s radar are shown in Table 1.

Table 1 Examples of Space Rendezvous Radar Parameters

	Space Shuttle Rendezvous Radar	**OMV Radar**
Range	12 nmi	4.5 nmi
Angle accuracy (3σ)	8 mrad	20 mrad
Angle rate (3σ)	0.14 mrads/s	N/A
Range accuracy (3σ)	80 ft, R < 1.3 nmi 1% of R, 1.3 < R < 4.9 nmi 300 ft, 4.9 < R < 12 nmi	Greater of 20 ft or 2% of range
Range rate accuracy (3σ)	1 ft/s, R < 10 nmi	Greater of 0.1 ft/s or 2% of range rate

Space Shuttle Rendezvous Radar data (Cantafio, 201); OMV data (Cantafio, 210)

An example of a range measuring system that might be suitable for a microsatellite mission is the laser rangefinder on the Near-Earth Asteroid Rendezvous (NEAR) spacecraft. The NEAR laser rangefinder (NLR) was developed at the Johns Hopkins Applied Physics Laboratory in the early 1990s. The complete NLR system has a mass of only 4.9 kg and a volume of 14.75" x 9" x 8.5." NLR has a maximum range of just over 100 km and a range accuracy of 2 meters (Cole, 124). Since the NLR is a laser rangefinder, no value for range rate measurement was listed. For the MATLAB simulation, it is assumed that a laser radar of comparable size and range and range rate measuring capability is available. The range rate accuracy is assumed to be 2 m/s.

III. Methodology

For the problems of initial orbit determination and orbit updating, various combinations of observation data must be processed using a suitable solution method. In Table 6-1 of *Fundamentals of Astrodynamics and Applications*, Vallado lists those data combinations and solution methods along with any restrictions in terms of minimum sets of a particular observation combination. Vallado's Table 6-1 is reproduced below.

Table 2 Types of Tracking Data for Initial Orbit Determination and Orbit Updating

Data Type	**Restrictions**	**Solution Method**
Range rate ($\dot{\rho}$)	None	Estimation
Azimuth (β), elevation (el)	3 sets minimum	***Laplace, Gauss, Double-r***
Range (ρ), azimuth (β), elevation (el)	2 sets minimum	***SITE-TRACK***, then ***Lambert*** (2) or ***GIBBS/HGIBBS***
Range (ρ), azimuth (β), elevation (el), range rate ($\dot{\rho}$)	2 or 3 sets minimum	***SITE-TRACK***
Range (ρ), azimuth (β), elevation (el), range rate ($\dot{\rho}$), azimuth rate ($\dot{\beta}$), elevation rate ($\dot{el}$)	None	***SITE-TRACK***
Topocentric right ascension, α_t, and declination, δ_t	3 sets minimum	***Laplace, Gauss, Double-r***
Range (ρ)	6 simultaneous, None	Trilateration, Estimation

(Vallado, 378)

For Table 2 above, the solution methods in bold italics are algorithms Vallado has outlined in his book. Several of those algorithms, or pieces thereof, were implemented in this thesis; however, portions of those algorithms using rate information other than range rate were not utilized.

3.1 Simulation Data Generation

Since "real" observations were not taken with "real" sensors to process through the non-linear least squares orbit determination filter, simulated data for range, range rate, azimuth, elevation, right ascension, and declination were generated. The following equations were used to generate the simulated data.

The satellite's state vector, $\vec{X}$, is represented with the Earth-centered Inertial (ECI) position vector, $\vec{r}$, and velocity vector, $\vec{v}$. $\vec{X}$ is written

$$\vec{X} = \begin{bmatrix} r_I \\ r_J \\ r_K \\ v_I \\ v_J \\ v_K \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{2}$$

The equations of motion for the two-body problem are written

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

where $\ddot{\vec{r}}$ is the satellite's acceleration vector, $\mu = 398{,}600.4415 \, {km^3}/{s^2}$ is the Earth's gravitational parameter, and $r = \sqrt{x^2 + y^2 + z^2}$ is the magnitude of the position vector. Using this formulation, the equations of motion for the satellite's state vector in first order form are

$$\frac{d\vec{X}}{dt} = \dot{\vec{X}} = \begin{bmatrix} \vec{v} \\ \dfrac{-\mu\vec{r}}{r^3} \end{bmatrix} \tag{4}$$

The equations of motion were integrated using MATLAB's built-in ordinary differential equation solver function, *ode45*, to obtain the satellite's state at specified time intervals along the trajectory. Then using equations from Vallado's Algorithm 15: *RAZEL* (Vallado, 173), range ρ, azimuth β, and elevation el were calculated.

To begin the calculations for range, the tracking site's ECI position vector must be determined. First, two auxiliary terms associated with the Earth's shape are calculated. The first auxiliary term is

$$C_{\oplus} = \frac{R_{\oplus}}{\sqrt{1 - e_{\oplus}^2 \sin^2(\phi_{gd})}} \tag{5}$$

where $R_{\oplus} = 6{,}378.1363 km$ is the Earth's equatorial radius, $e_{\oplus} = 0.081819221456$ is the Earth's shape eccentricity (not its orbital eccentricity), and ϕ_{gd} is the tracking site's geodetic latitude. The second auxiliary term is

$$S_{\oplus} = C_{\oplus}(1 - e_{\oplus}^2) \tag{6}$$

Using these two auxiliary terms, the horizontal (in the plane of the Earth's equator) and the vertical (towards the North Pole for a positive (northern) latitude and towards the South Pole for a negative (southerly) latitude) components of the tracking site's position vector are determined next. The horizontal component is

$$r_{\delta} = (C_{\oplus} + h_{ellp})\cos(\phi_{gd}) \tag{7}$$

where h_{ellp} is the tracking site's height in kilometers (or other consistent units) above the reference geoid. The vertical component is

$$r_{K} = (S_{\oplus} + h_{ellp})\sin(\phi_{gd}) \tag{8}$$

The tracking site's ECI position vector is then

$$\vec{r}_{siteIJK} = \begin{bmatrix} r_\delta \cos(\theta_{LST}) \\ r_\delta \sin(\theta_{LST}) \\ r_K \end{bmatrix} \tag{9}$$

where θ_{LST} is the Local Sidereal Time (LST) at the tracking site. Local Sidereal Time can be calculated using Vallado's Algorithm 1: *LSTIME*. First, calculate T_{UT1}, the number of Julian centuries elapsed from the epoch J2000,

$$T_{UT1} = \frac{JD_0 - 2{,}451{,}545.0}{36{,}525} \tag{10}$$

where JD_0 is the Julian day number for the calendar date of interest and 2,451,545.0 is the Julian day of January 1, 2000. Second, calculate the Greenwich mean sidereal time at midnight 0000 Universal Time, θ_{GST0}, in degrees, for the date of interest

$$\theta_{GST0} = 100.4606184° + 36{,}000.77005361 T_{UT1} + 0.00038793 T_{UT1}^2 - 2.6 \times 10^{-8} T_{UT1}^3 \tag{11}$$

Third, calculate the Greenwich sidereal time for the specific time of the day by

$$\theta_{GST} = \theta_{GST0} + \omega_{\oplus} UT1 \tag{12}$$

where $\omega_{\oplus}$ is the magnitude of the Earth's rotational (angular) velocity and *UT1* is the elapsed time since midnight in seconds. Finally, local sidereal time is given by

$$\theta_{LST} = \theta_{GST} + \lambda \tag{13}$$

where λ is the tracking site's longitude (east longitude is positive and west longitude is negative).

With both the satellite and the tracking site positions known, range in ECI coordinates from the tracking station to the satellite may be calculated as

$$\vec{\rho}_{IJK} = \vec{r}_{IJK} - \vec{r}_{siteIJK} \tag{14}$$

The relative velocity vector from the tracking station to the satellite in ECI coordinates is calculated next by

$$\dot{\vec{\rho}} = \vec{v}_{IJK} - \vec{\omega}_{\oplus} \times \vec{r}_{siteIJK} \tag{15}$$

where $\vec{\omega}_{\oplus}$ is the Earth's rotational (angular) velocity vector in radians/sec given by

$$\vec{\omega}_{\oplus} = \begin{bmatrix} 0 \\ 0 \\ 0.000072921158553 \end{bmatrix} \tag{16}$$

Once the ECI range and relative velocity vectors are determined, they must be transformed (rotated) from the ECI coordinate system to the topocentric horizon SEZ (South-East-Zenith) coordinate system. The combined transformation matrix

$$IJKtoSEZ = \begin{bmatrix} \sin(\phi_{gd})\cos(\theta_{LST}) & \sin(\phi_{gd})\sin(\theta_{LST}) & -\cos(\phi_{gd}) \\ -\sin(\theta_{LST}) & \cos(\theta_{LST}) & 0 \\ \cos(\phi_{gd})\cos(\theta_{LST}) & \cos(\phi_{gd})\sin(\theta_{LST}) & \sin(\phi_{gd}) \end{bmatrix} \tag{17}$$

leads to the rotations

$$\vec{\rho}_{SEZ} = [IJKtoSEZ]\vec{\rho}_{IJK} \tag{18}$$

$$\dot{\vec{\rho}}_{SEZ} = [IJKtoSEZ]\dot{\vec{\rho}}_{IJK} \tag{19}$$

The range (a scalar) is simply the magnitude (vector norm) of the SEZ range vector

$$\rho = |\vec{\rho}_{SEZ}| \tag{20}$$

The elevation angle from the tracking station's horizon to the satellite is given by

$$el = \sin^{-1}\left(\frac{\rho_Z}{\rho}\right) \tag{21}$$

and the azimuth angle, β, is given by

$$\sin(\beta) = \frac{\rho_E}{\sqrt{\rho_S^2 + \rho_E^2}} \tag{22}$$

$$\cos(\beta) = \frac{-\rho_S}{\sqrt{\rho_S^2 + \rho_E^2}} \tag{23}$$

$$\beta = \tan^{-1}\left(\frac{\sin(\beta)}{\cos(\beta)}\right) \tag{24}$$

The final equation used from Vallado's *RAZEL* algorithm yields the range rate, which is given by

$$\dot{\rho} = \frac{\vec{\rho}_{SEZ} \cdot \dot{\vec{\rho}}_{SEZ}}{\rho} \tag{25}$$

In addition to developing equations for range, range rate, azimuth, and elevation, equations were also developed for topocentric right ascension and declination using Vallado's Algorithm 14: *Topocentric* (Vallado, 168). This algorithm is analogous to the *RAZEL* algorithm since the range vector in ECI coordinates is calculated but is not transformed to SEZ coordinates. The declination angle (positive above the celestial equator and negative below) is determined similar to elevation by

$$\delta_t = \frac{\rho_K}{\rho} \tag{26}$$

and the right ascension is determined similar to azimuth by

$$\sin(\alpha_t) = \frac{\rho_J}{\sqrt{\rho_I^2 + \rho_J^2}} \tag{27}$$

$$\cos(\alpha_t) = \frac{-\rho_I}{\sqrt{\rho_I^2 + \rho_J^2}} \tag{28}$$

$$\alpha_t = \tan^{-1}\left(\frac{\sin(\alpha_t)}{\cos(\alpha_t)}\right) \tag{29}$$

3.2 Initial Orbit Determination

With radar (or other sensor) providing range, azimuth, and elevation data, the satellite's position and velocity vectors in ECI coordinates can be calculated. First, the satellite's SEZ range coordinates are found using the radar data directly

$$\vec{\rho}_{SEZ} = \begin{bmatrix} -\rho\cos(el)\cos(\beta) \\ \rho\cos(el)\sin(\beta) \\ \rho\sin(el) \end{bmatrix} \tag{30}$$

The range rate in SEZ coordinates is found by taking the derivatives of the range component equations with respect to all three variables by the chain rule to yield

$$\dot{\vec{\rho}}_{SEZ} = \begin{bmatrix} -\dot{\rho}\cos(el)\cos(\beta) + \rho\sin(el)\cos(\beta)\dot{el} + \rho\cos(el)\sin(\beta)\dot{\beta} \\ \dot{\rho}\cos(el)\sin(\beta) - \rho\sin(el)\sin(\beta)\dot{el} + \rho\cos(el)\cos(\beta)\dot{\beta} \\ \dot{\rho}\sin(el) + \rho\cos(el)\dot{el} \end{bmatrix} \tag{31}$$

Next, the SEZ coordinate values must be transformed (rotated) to the IJK coordinate frame by the combined transformation matrix

$$SEZtoIJK = \begin{bmatrix} \sin(\phi_{gd})\cos(\theta_{LST}) & -\sin(\theta_{LST}) & \cos(\phi_{gd})\cos(\theta_{LST}) \\ \sin(\phi_{gd})\sin(\theta_{LST}) & \cos(\theta_{LST}) & \cos(\phi_{gd})\sin(\theta_{LST}) \\ -\cos(\phi_{gd}) & 0 & \sin(\phi_{gd}) \end{bmatrix} \tag{32}$$

which then leads to the rotations

$$\vec{\rho}_{IJK} = [SEZtoIJK]\vec{\rho}_{SEZ} \tag{33}$$

$$\dot{\vec{\rho}}_{IJK} = [SEZtoIJK]\dot{\vec{\rho}}_{SEZ} \tag{34}$$

The ECI position and velocity vectors are then determined by

$$\vec{r}_{IJK} = \vec{\rho}_{IJK} + \vec{r}_{siteIJK} \tag{35}$$

$$\vec{v}_{IJK} = \dot{\vec{\rho}}_{IJK} + \vec{\omega}_{\oplus} \times \vec{r}_{siteIJK} \tag{36}$$

Since radar sites do not always gather angular rate data, then the Gibbs and Herrick-Gibbs Methods can be used to determine the initial velocity vector, $\vec{v}_2$, associated with the second of three sequential position vectors, $\vec{r}_2$.

3.2.1 Gibbs Method for Initial Orbit Determination

The Gibbs Method in Vallado's algorithm 48 (Vallado, 414) is outlined below. First, assuming three sequential position vectors, in ECI coordinates are available, then form the vectors

$$\vec{Z}_{12} = \vec{r}_1 \times \vec{r}_2 \tag{37}$$

$$\vec{Z}_{23} = \vec{r}_2 \times \vec{r}_3 \tag{38}$$

$$\vec{Z}_{31} = \vec{r}_3 \times \vec{r}_1 \tag{39}$$

Next, test that the input vectors are coplanar by calculating the angle

$$\alpha_{cop} = 90^\circ - \cos^{-1}\left(\frac{\vec{Z}_{23} \cdot \vec{r}_1}{|\vec{Z}_{23}||\vec{r}_1|}\right) \tag{40}$$

If the vectors are exactly coplanar, then $\alpha_{cop} = 0$. If the vectors are not exactly coplanar, then the user must determine an acceptable error tolerance and proceed. Vallado recommends no more than 2 or 3 degrees. The vectors must also have some angular separation within their common plane. The Gibbs Method works with at least 1^{o}

separation and is superior to the Herrick-Gibbs Method when the separation is greater than 5^{o}. The angular separation can be tested by calculating the two angles

$$\cos(\alpha_{12}) = \frac{\vec{r}_1 \cdot \vec{r}_2}{|\vec{r}_1||\vec{r}_2|} \tag{41}$$

$$\cos(\alpha_{23}) = \frac{\vec{r}_2 \cdot \vec{r}_3}{|\vec{r}_2||\vec{r}_3|} \tag{42}$$

where α_{12} is the angle between vectors $\vec{r}_1$ and $\vec{r}_2$ and α_{23} is the angle between vectors $\vec{r}_2$ and $\vec{r}_3$. If the angular separation is sufficient, then four intermediate vectors can be calculated

$$\vec{N} = r_1\left(\vec{Z}_{23}\right) + r_2\left(\vec{Z}_{31}\right) + r_3\left(\vec{Z}_{12}\right) \tag{43}$$

$$\vec{D} = \vec{Z}_{12} + \vec{Z}_{23} + \vec{Z}_{31} \tag{44}$$

$$\vec{S} = (r_2 - r_3)\vec{r}_1 + (r_3 - r_1)\vec{r}_2 + (r_1 - r_2)\vec{r}_3 \tag{45}$$

$$\vec{B} = \vec{D} \times \vec{r}_2 \tag{46}$$

Using one final scalar given by

$$L_g = \sqrt{\frac{\mu}{ND}} \tag{47}$$

calculate the velocity vector associated with $\vec{r}_2$

$$\vec{v}_2 = \frac{L_g}{r_2}\vec{B} + L_g\vec{S} \tag{48}$$

With both position and velocity known, the orbit is considered determined.

3.2.2 Herrick-Gibbs Initial Orbit Determination

The Herrick-Gibbs Method, which also determines the velocity vector associated with the second of three sequential position vectors, is used when the angular separation is less than 1°. The Julian Dates associated with the three position vectors are also used in this algorithm. First, calculate the time differences between the position vectors

$$\Delta t_{31} = JD_3 - JD_1 \tag{49}$$

$$\Delta t_{32} = JD_3 - JD_2 \tag{50}$$

$$\Delta t_{21} = JD_2 - JD_1 \tag{51}$$

From this point, the Herrick-Gibbs Method is similar to the Gibbs Method by also testing whether the position vectors are coplanar and checking the angular separation

$$\vec{Z}_{23} = \vec{r}_2 \times \vec{r}_3 \tag{52}$$

$$\alpha_{cop} = 90^{\circ} - \cos^{-1}\left(\frac{\vec{Z}_{23} \cdot \vec{r}_1}{|\vec{Z}_{23}||\vec{r}_1|} \right) \tag{53}$$

$$\cos(\alpha_{12}) = \frac{\vec{r}_1 \cdot \vec{r}_2}{|\vec{r}_1||\vec{r}_2|} \tag{54}$$

$$\cos(\alpha_{23}) = \frac{\vec{r}_2 \cdot \vec{r}_3}{|\vec{r}_2||\vec{r}_3|} \tag{55}$$

If the degree of coplanarness and separation are acceptable, then the velocity vector is

$$\vec{v}_2 = -\Delta t_{32}\left(\frac{1}{\Delta t_{21}\Delta t_{31}} + \frac{\mu}{12r_1^3} \right)\vec{r}_1 + (\Delta t_{32} - \Delta t_{21})\left(\frac{1}{\Delta t_{21}\Delta t_{32}} + \frac{\mu}{12r_2^3} \right)\vec{r}_2$$

$$+ \Delta t_{21}\left(\frac{1}{\Delta t_{32}\Delta t_{31}} + \frac{\mu}{12r_3^3} \right)\vec{r}_3 \tag{56}$$

and, again, with both the position and velocity known, the orbit is considered determined. The orbit determined by the Gibbs or Herrick-Gibbs methods can then be used as the estimate for the satellite's reference trajectory in the non-linear least squares filter.

3.3 Non-linear Least Squares Orbit Determination Filter

Wiesel's non-linear least squares algorithm from his book *Modern Methods of Orbit Determination* is described in this section. First, assuming there are multiple observations, then for each observation time t_i , propagate the state vector to the observation time t_i and obtain the state transition matrix $\Phi\left(t_i, t_0\right)$. With the satellite's state vector written as

$$\vec{X} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{57}$$

the state transition matrix is a 6 x 6 matrix whose components are the partial derivatives of each state component with respect to each component of the state itself

$$\Phi = \begin{bmatrix} \frac{\partial x}{\partial x} & \frac{\partial x}{\partial y} & \frac{\partial x}{\partial z} & \frac{\partial x}{\partial \dot{x}} & \frac{\partial x}{\partial \dot{y}} & \frac{\partial x}{\partial \dot{z}} \\ \frac{\partial y}{\partial x} & \frac{\partial y}{\partial y} & \frac{\partial y}{\partial z} & \frac{\partial y}{\partial \dot{x}} & \frac{\partial y}{\partial \dot{y}} & \frac{\partial y}{\partial \dot{z}} \\ \frac{\partial z}{\partial x} & \frac{\partial z}{\partial y} & \frac{\partial z}{\partial z} & \frac{\partial z}{\partial \dot{x}} & \frac{\partial z}{\partial \dot{y}} & \frac{\partial z}{\partial \dot{z}} \\ \frac{\partial \dot{x}}{\partial x} & \frac{\partial \dot{x}}{\partial y} & \frac{\partial \dot{x}}{\partial z} & \frac{\partial \dot{x}}{\partial \dot{x}} & \frac{\partial \dot{x}}{\partial \dot{y}} & \frac{\partial \dot{x}}{\partial \dot{z}} \\ \frac{\partial \dot{y}}{\partial x} & \frac{\partial \dot{y}}{\partial y} & \frac{\partial \dot{y}}{\partial z} & \frac{\partial \dot{y}}{\partial \dot{x}} & \frac{\partial \dot{y}}{\partial \dot{y}} & \frac{\partial \dot{y}}{\partial \dot{z}} \\ \frac{\partial \dot{z}}{\partial x} & \frac{\partial \dot{z}}{\partial y} & \frac{\partial \dot{z}}{\partial z} & \frac{\partial \dot{z}}{\partial \dot{x}} & \frac{\partial \dot{z}}{\partial \dot{y}} & \frac{\partial \dot{z}}{\partial \dot{z}} \end{bmatrix} \tag{58}$$

At $t_i = t_0$ this results in the 6 x 6 identity matrix, I_{6x6},

$$\Phi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{59}$$

Next, obtain the residual vector $\mathbf{r}_i = \mathbf{z}_i - \mathbf{G}(\vec{X})$. $\mathbf{z}_i$ is the n x 1 measured data vector for this observation time, where n is the number of types of observations being taken. For example, if at observation t_i, observations were taken for range, azimuth, and elevation, then $n = 3$. $\mathbf{G}(\vec{X})$ is the predicted data vector as a function of the current state vector $\vec{X}$. The form of $\mathbf{G}(\vec{X})$ depends on what predicted data is needed. For example, assume a radar, whose position vector is known, is measuring range, azimuth, and elevation to a target satellite, then the predicted range, azimuth, and elevation based on the propagated reference trajectory would be calculated using the same equations that were used to generate the simulation data described in section 3.1. These predicted data are then subtracted from the corresponding measured data to form the residual vector

$$r_i = z_{measured} - z_{predicted} \tag{60}$$

which would take the form

$$r_i = \begin{pmatrix} \rho_{measured} - \rho_{predicted} \\ \beta_{measured} - \beta_{predicted} \\ el_{measured} - el_{predicted} \end{pmatrix} \tag{61}$$

It should be noted that noise was added to the generated data using a Gaussian random number in order to give "realistic" measurement data. Next, calculate H_i for this particular data point. H_i, the linear observations model, is the *n* x 6 matrix of partial derivatives of the **G** vector with respect to the state evaluated on the reference trajectory where *n* is still the number of types of observation being taken

$$H_{ij} = \frac{\partial G_i}{\partial X_j} |_{X_{ref}} \tag{62}$$

where i = 1 to n is the **G** vector component being differentiated and j = 1 to 6 is the state vector component **G** is the differentiation variable. Together, i and j, are the row and column indices of the *n* x 6 *H* matrix.

The **G** vectors and their associated *H* matrices for the data types coded in the simulation program are shown next. Since *H* is an *n* x 6 matrix, if a specific value for an element of *H* is not shown, it is assumed to be 0.

$$\mathbf{G}_i(\vec{X}) = Range, \rho = \sqrt{x^2 + y^2 + z^2} \tag{63}$$

$$H_{i1} = \frac{x}{\sqrt{x^2 + y^2 + z^2}} = \frac{x}{r} \tag{64}$$

$$H_{i2} = \frac{y}{\sqrt{x^2 + y^2 + z^2}} = \frac{y}{r} \tag{65}$$

$$H_{i3} = \frac{z}{\sqrt{x^2 + y^2 + z^2}} = \frac{z}{r} \tag{66}$$

$$\mathbf{G}_i\,(\vec{X}\,) = \ Azimuth, \beta = \tan^{-1}\left(\frac{y}{x}\right) \tag{67}$$

$$H_{i1} = \frac{\frac{-y}{x^2}}{1 + \left(\frac{y}{x}\right)^2} \tag{68}$$

$$H_{i2} = \frac{\frac{1}{x}}{1 + \left(\frac{y}{x}\right)^2} \tag{69}$$

$$H_{i3} = 0 \tag{70}$$

$$\mathbf{G}_i\,(\vec{X}\,) \ = elevation, el = \tan^{-1}\left(\frac{z}{\sqrt{x^2 + y^2}}\right) \tag{71}$$

$$H_{i1} = \frac{\frac{-xz}{\left(x^2 + y^2\right)^{3/2}}}{\left(1 + \frac{z^2}{\left(x^2 + y^2\right)}\right)} \tag{72}$$

$$H_{i2} = \frac{\frac{-yz}{\left(x^2 + y^2\right)^{3/2}}}{\left(1 + \frac{z^2}{\left(x^2 + y^2\right)}\right)} \tag{73}$$

$$H_{i3} = \frac{\frac{1}{\left(x^2 + y^2\right)^{1/2}}}{\left(1 + \frac{z^2}{\left(x^2 + y^2\right)}\right)} \tag{74}$$

$$\mathbf{G}_i\,(\vec{X}) = \text{Range rate, } \dot{\rho} = \frac{\vec{\rho}\cdot\dot{\vec{\rho}}}{\rho} \tag{75}$$

$\dot{\vec{\rho}}$ is the relative velocity vector in ECI coordinates from the tracking site to the satellite for the case of the ground-based orbit determination filter or the relative velocity vector in ECI coordinates from the microsatellite to the target satellite for the on-orbit version of the orbit determination filter

$$\dot{\vec{\rho}} = \begin{bmatrix} \dot{x}_{sat,IJK} - \dot{x}_{site,IJK} \\ \dot{y}_{sat,IJK} - \dot{y}_{site,IJK} \\ \dot{z}_{sat,IJK} - \dot{z}_{site,IJK} \end{bmatrix} \tag{76}$$

$$\dot{\vec{\rho}} = \begin{bmatrix} \dot{x}_{t\arg et,IJK} - \dot{x}_{micro,IJK} \\ \dot{y}_{t\arg et,IJK} - \dot{y}_{micro,IJK} \\ \dot{z}_{t\arg et,IJK} - \dot{z}_{micro,IJK} \end{bmatrix} \tag{77}$$

When the dot product and division have been performed the resulting equation for range rate is then

$$\dot{\rho} = \frac{(x_{sat} - x_{site})(\dot{x}_{sat} - \dot{x}_{site}) + (y_{sat} - y_{site})(\dot{y}_{sat} - \dot{y}_{site}) + (z_{sat} - z_{site})(\dot{z}_{sat} - \dot{z}_{site})}{\sqrt{(x_{sat} - x_{site})^2 + (y_{sat} - y_{site})^2 + (z_{sat} - z_{site})^2}} \tag{78}$$

and similarly for the on orbit case from microsatellite to the target satellite. The 1 x 6 *H* matrix for range rate is

$$H_{i1} = \frac{\dot{\rho}_x}{\rho} - \frac{\dot{\rho}\rho_x}{\rho^2} \tag{79}$$

$$H_{i2} = \frac{\dot{\rho}_y}{\rho} - \frac{\dot{\rho}\rho_y}{\rho^2} \tag{80}$$

$$H_{i3} = \frac{\dot{\rho}_z}{\rho} - \frac{\dot{\rho}\rho_z}{\rho^2} \tag{81}$$

$$H_{i4} = \frac{\rho_x}{\rho} \tag{82}$$

$$H_{i5} = \frac{\rho_y}{\rho} \tag{83}$$

$$H_{i6} = \frac{\rho_z}{\rho} \tag{84}$$

$$\mathbf{G}_i(\vec{X}) = \textit{Right Ascension}, \alpha = \tan^{-1}\left(\frac{\rho_{y,IJK}}{\rho_{x,IJK}}\right) \tag{85}$$

$$H_{i1} = \frac{\dfrac{-\rho_{y,IJK}}{\rho_{x,IJK}^2}}{1 + \left(\dfrac{\rho_{y,IJK}}{\rho_{x,IJK}}\right)^2} \tag{86}$$

$$H_{i2} = \frac{\dfrac{1}{\rho_{x,IJK}}}{1 + \left(\dfrac{\rho_{y,IJK}}{\rho_{x,IJK}}\right)^2} \tag{87}$$

$$H_3 = 0 \tag{88}$$

$$\mathbf{G}_i(\vec{X}) = \textit{declination}, \delta_t = \tan^{-1}\left(\frac{\rho_{z,IJK}}{\sqrt{\rho_{x,IJK}^2 + \rho_{y,IJK}^2}}\right) \tag{89}$$

$$H_{i1} = \frac{\dfrac{-\rho_{x,IJK}\rho_{z,IJK}}{\left(\rho_{x,IJK}^2 + \rho_{y,IJK}^2\right)^{3/2}}}{\left(1 + \dfrac{\rho_{z,IJK}^2}{\left(\rho_{x,IJK}^2 + \rho_{y,IJK}^2\right)}\right)} \tag{90}$$

$$H_{i2} = \frac{\dfrac{-\rho_{y,IJK}\rho_{z,IJK}}{\left(\rho_{x,IJK}^2 + \rho_{y,IJK}^2\right)^{3/2}}}{\left(1 + \dfrac{\rho_{z,IJK}^2}{\left(\rho_{x,IJK}^2 + \rho_{y,IJK}^2\right)}\right)} \tag{91}$$

$$H_{i3} = \frac{\dfrac{1}{\left(\rho_{x,IJK}^2 + \rho_{y,IJK}^2\right)^{1/2}}}{\left(1 + \dfrac{\rho_{z,IJK}^2}{\left(\rho_{x,IJK}^2 + \rho_{y,IJK}^2\right)}\right)} \tag{92}$$

Next, calculate the observation matrix, $T_i = H_i\Phi$, and add new terms to the running sums of the matrix

$$\sum_i T_i^T Q_i^{-1} T_i \tag{93}$$

and the vector

$$\sum_i T_i^T Q_i^{-1} \vec{r}_i \tag{94}$$

where Q is the instrumental covariance (or observation covariance) matrix and Q^{-1} is its inverse. The matrix $T_i^T Q_i^{-1} T_i$ must be invertible for a new estimate of the reference trajectory to exist. Wiesel calls this the *observability condition*. When all data has been processed calculate the covariance of the correction

$$P_{\delta x} = \left(\sum_i \mathrm{T_i^T Q_i^{-1} T_i}\right)^{-1} \tag{95}$$

and the state correction vector at epoch

$$\delta\vec{x}(t_0) = P_{\delta x} \sum_i T_i^T Q_i^{-1} \vec{r}_i \tag{96}$$

Update the reference trajectory vector by adding the state correction vector

$$\vec{X}_{ref+1}(t_0) = \vec{X}_{ref}(t_0) + \delta\vec{x}(t_0) \tag{97}$$

Determine if the process has converged. If it has, then $\vec{X}_{ref+1}$ is the new estimate of the reference trajectory with covariance $P_{\delta x}$. Finally, check the residuals to see if they are of appropriate magnitude and distribution.

3.4 Perturbations

Perturbations, deviations from a normal, idealized, or unperturbed motion, which can be included within the dynamics model in the simulation include Earth oblateness effects from the J2 zonal gravity harmonic, third-body gravitational effects from the Sun and the Moon, and atmospheric drag. Each is discussed below.

3.4.1 J2

The accelerations, in ECI coordinates, resulting from the Earth's oblateness, or non-spherical shape, are

$$\vec{a}_{J2} = \begin{bmatrix} \dfrac{-3J_2\mu R_\oplus^2 r_I}{2r^5}\left(1-\dfrac{5r_K^2}{r^2}\right) \\ \dfrac{-3J_2\mu R_\oplus^2 r_J}{2r^5}\left(1-\dfrac{5r_K^2}{r^2}\right) \\ \dfrac{-3J_2\mu R_\oplus^2 r_K}{2r^5}\left(3-\dfrac{5r_K^2}{r^2}\right) \end{bmatrix} \tag{98}$$

where $J_2 = 0.0010826269$ is the dimensionless second zonal gravity harmonic coefficient, μ is the Earth's gravitational parameter, and $R_\oplus$ is the Earth's equatorial radius.

3.4.2 Third-Body Gravitational Effects

The acceleration of the satellite relative to the Earth due to the gravitational influence of a third body such as the Sun or Moon can be calculated by the following equation from page 10 of *Fundamentals of Astrodynamics*

$$\ddot{\vec{r}}_{12} = -\frac{G(m_1+m_2)}{r_{12}^3}\vec{r}_{12} - \sum_{j=3}^{n} Gm_j\left(\frac{\vec{r}_{j2}}{r_{j2}^3} - \frac{\vec{r}_{j1}}{r_{j1}^3}\right) \tag{99}$$

where the first term is the two-body equation; however, this equation includes the (m_1+m_2) term for completeness where m_2 is the satellite's mass . It is usually assumed that the satellite's mass is insignificant relative to the mass of the central body and is thus dropped, leaving $Gm_1 = \mu$, the gravitational parameter. The summation term is the third-body contribution to the acceleration. Gm_j is 1.32712428E+11 km^3/s^2 for the Sun and 4,902.799 km^3/s^2 for the Moon. $\vec{r}_{j2}$ is the vector from the third body to the satellite and $\vec{r}_{j1}$ is the vector from the third body to the central body, which for this simulation is Earth.

3.4.2.1 Sun Position Vector

The Sun's geocentric position vector to be used in the equation for third-body gravitational effects can be calculated by Vallado's Algorithm 18: *Sun* (Vallado, 183). The algorithm begins by computing

$$T_{UT1} = \frac{JD_{UT1} - 2{,}451{,}545.0}{36{,}525} \tag{100}$$

where T_{UT1} is the number of Julian centuries elapsed from the epoch J2000 and JD_{UT1} is the Julian date of the time of interest.

The mean longitude of the Sun is

$$\lambda_{M_{sun}} = 280.4606184° + 36{,}000.77005361 T_{UT1} \tag{101}$$

The Sun's mean anomaly is

$$M_{sun} = 357.5277233° + 35{,}999.05034 T_{TDB} \tag{102}$$

T_{TDB}, barycentric dynamical time, is a more precise parameter that includes more details such as relativistic effects, etc. that are not needed for the level of precision for most analyses. In this case, T_{TDB} may be assumed to be approximately equal to T_{UT1}.

Then for this algorithm, the longitude of the ecliptic is

$$\lambda_{ecliptic} = \lambda_{M_{sun}} + 1.914666471° \sin(M_{sun}) + 0.019994643 \sin(2M_{sun}) \tag{103}$$

where the ecliptic is the mean plane of the Earth's orbit about the Sun.

The magnitude of the Sun's position vector, in astronomical units, is

$$r_{sun} = 1.000140612 - 0.016708617 \cos(M_{sun}) - 0.000139589 \cos(2M_{sun}) \tag{104}$$

Also for this algorithm, ε, the obliquity of the ecliptic, which is the angle between the Earth's mean equator and the ecliptic, is given by

$$\varepsilon = 23.439291° - 0.0130042 T_{TDB} \tag{105}$$

Finally, the Sun's position vector, in astronomical units, is given by

$$\vec{r}_{sun} = \begin{bmatrix} r_{sun} \cos(\lambda_{ecliptic}) \\ r_{sun} \cos(\varepsilon) \sin(\lambda_{ecliptic}) \\ r_{sun} \sin(\varepsilon) \sin(\lambda_{ecliptic}) \end{bmatrix} \tag{106}$$

3.4.2.2 Moon Position Vector

Similarly, the Moon's geocentric position vector can also be computed by Vallado's Algorithm 19: *Moon* (Vallado,186). This algorithm also begins by computing a time parameter

$$T_{TDB} = \frac{JD_{TDB} - 2{,}451{,}545.0}{36{,}525} \tag{107}$$

The longitude of the ecliptic is

$$\begin{aligned}\lambda_{ecliptic} &= 218.32^\circ + 481{,}267.8813T_{TDB} + 6.29\sin(134.9 + 477{,}198.85T_{TDB}) \\ &- 1.27\sin(259.2 - 413{,}335.38T_{TDB}) + 0.66\sin(235.7 + 890{,}534.23T_{TDB}) \\ &+ 0.21\sin(269.9 + 954{,}397.70T_{TDB}) - 0.19\sin(357.5 + 35{,}999.05T_{TDB}) \\ &- 0.11\sin(186.6 + 966{,}404.05T_{TDB})\end{aligned} \tag{108}$$

The latitude of the ecliptic is

$$\begin{aligned}\phi_{ecliptic} &= 5.13^\circ\sin(93.3 + 483{,}202.03T_{TDB}) + 0.28\sin(228.2 + 960{,}400.87T_{TDB}) \\ &- 0.28\sin(318.3 + 6{,}003.18T_{TDB}) - 0.17\sin(217.6 - 407{,}332.20T_{TDB})\end{aligned} \tag{109}$$

The parallax is

$$\begin{aligned}\wp &= 0.9508^\circ + 0.0518\cos(134.9 + 477{,}198.85T_{TDB}) + 0.0095\cos(259.2 - 413{,}335.38T_{TDB}) \\ &+ 0.0078\cos(235.7 + 890{,}534.23T_{TDB}) + 0.0028\cos(269.9 + 954{,}397.70T_{TDB})\end{aligned} \tag{110}$$

The magnitude of the position vector in Earth radii is then

$$r_{moon} = \frac{1}{\sin(\wp)} \tag{111}$$

The position vector is then

$$\vec{r}_{moon} = r_{moon}\begin{bmatrix}\cos(\phi_{ecliptic})\cos(\lambda_{ecliptic}) \\ \cos(\varepsilon)\cos(\phi_{ecliptic})\sin(\lambda_{ecliptic}) - \sin(\varepsilon)\sin(\phi_{ecliptic}) \\ \sin(\varepsilon)\cos(\phi_{ecliptic})\sin(\lambda_{ecliptic}) + \cos(\varepsilon)\sin(\phi_{ecliptic})\end{bmatrix} \tag{112}$$

where ε, the obliquity of the ecliptic is given in radians by Equation 1-58 of Vallado

$$\varepsilon = 0.40909280 - 0.000226966 T_{TDB} - 2.86\times10^{-9} T_{TDB}^2 + 8.80\times10^{-9} T_{TDB}^3 \qquad (113)$$

To convert from Earth radii to kilometers multiply by 6,378.1363.

3.4.3 Atmospheric Drag

The acceleration due to atmospheric drag is given by Equation 7-24 from Vallado (Vallado, 498)

$$\vec{a}_{drag} = -\frac{1}{2}\frac{c_D A}{m}\rho v_{rel}^2 \frac{\vec{v}_{rel}}{|\vec{v}_{rel}|} \qquad (114)$$

where c_D is the satellite's coefficient of drag, A is the satellite's cross-sectional area normal to the satellite's velocity vector, m is the satellite's mass, ρ is the atmospheric density at the satellite's altitude, and $\vec{v}_{rel}$ is the satellite's velocity vector relative to the Earth's rotating atmosphere. The relative velocity is given by

$$\vec{v}_{rel} = \frac{d\vec{r}}{dt} - \vec{\omega}_\oplus \times \vec{r} = \begin{bmatrix} \frac{dx}{dt} + \omega_\oplus y \\ \frac{dy}{dt} - \omega_\oplus x \\ \frac{dz}{dt} \end{bmatrix} = \begin{bmatrix} \dot{x} + \omega_\oplus y \\ \dot{y} - \omega_\oplus x \\ \dot{z} \end{bmatrix} \qquad (115)$$

The atmospheric density is given by an exponential model which gives values from 0 to 1,000 km of altitude. The exponential model is

$$\rho = \rho_o e^{\frac{-(h_{ellp} - h_o)}{H}} \qquad (116)$$

where ρ_o is the reference density for the specific altitude, h_{ellp} is the actual altitude of the satellite, h_o is the reference altitude, and H is the scale height. Vallado tabulates values for these parameters in Table 7-4 of his book, *Fundamentals of Astrodynamics and Applications*.

3.5 Equations of Variation

The equations of variation are the partial derivatives of the acceleration term equations with respect to the state and are used to form the 6 x 6 *A* matrix, which in turn, is used to form the derivative of the state transition matrix, Φ, as

$$\frac{d}{dt}\Phi(t,t_0) = \dot{\Phi} = A(t)\Phi(t,t_0) \tag{117}$$

The 36 components of $\dot{\Phi}$ are combined with the 6 components of the satellite's state vector derivatives $\dot{\vec{X}}$ to form the "total" state derivative which is integrated by *ode45*.

3.5.1 Equations of Variation for the Two-Body Problem

From Wiesel (Wiesel, 78), the equations of variation for the basic two-body problem are

$$A(t) = \begin{pmatrix} \phi & I \\ A_{rr} & \phi \end{pmatrix} \tag{118}$$

where 3 x 3 ϕ is the null matrix, I is a 3 x 3 identity matrix, and A_{rr} is

$$A_{rr} = \begin{pmatrix} \frac{-\mu}{r^3} + \frac{3\mu x^2}{r^5} & \frac{3\mu xy}{r^5} & \frac{3\mu xz}{r^5} \\ \frac{3\mu xy}{r^5} & \frac{-\mu}{r^3} + \frac{3\mu y^2}{r^5} & \frac{3\mu yz}{r^5} \\ \frac{3\mu xz}{r^5} & \frac{3\mu yz}{r^5} & \frac{-\mu}{r^3} + \frac{3\mu z^2}{r^5} \end{pmatrix} \tag{119}$$

Since the acceleration terms for the two-body problem are dependent only on the satellite's position, the *A* matrix is non-zero and/or non-unity only where there are position-related component terms. Any velocity related terms would appear in the three rightmost columns, as will be seen in Section 3.5.4.

3.5.2 Equations of Variation for J₂

Similar to the basic two-body problem, the equations of variation for J_2 are dependent only on position-related terms and thus populate only the lower 3 x 3 corner of the *A* matrix

$$A_{rr,J2} = \begin{pmatrix} \phi & \phi \\ A_{J2} & \phi \end{pmatrix} \tag{120}$$

The equations for the individual components are

$$A_{J2,41} = -\frac{3}{2} J_2 R_\oplus^2 \left[\left(1 - \frac{5z^2}{r^2}\right)\left(\frac{1}{r^5} - \frac{5x^2}{r^7}\right) + \frac{10x^2z^2}{r^9} \right] \tag{121}$$

$$A_{J2,42} = -\frac{3}{2} J_2 R_\oplus^2 x \left[\left(1 - \frac{5z^2}{r^2}\right)\left(-\frac{5y}{r^7}\right) + \frac{10yz^2}{r^9} \right] \tag{122}$$

$$A_{J2,43} = -\frac{3}{2} J_2 R_\oplus^2 x \left[\left(1 - \frac{5z^2}{r^2}\right)\left(-\frac{5z}{r^7}\right) + \left(\frac{10z}{r^7}\right)\left(\frac{z^2}{r^2} - 1\right) \right] \tag{123}$$

$$A_{J2,51} = -\frac{3}{2} J_2 R_\oplus^2 y \left[\left(1 - \frac{5z^2}{r^2}\right)\left(-\frac{5x}{r^7}\right) + \frac{10xz^2}{r^9} \right] \tag{124}$$

$$A_{J2,52} = -\frac{3}{2} J_2 R_\oplus^2 \left[\left(1 - \frac{5z^2}{r^2}\right)\left(\frac{1}{r^5} - \frac{5y^2}{r^7}\right) + \frac{10y^2z^2}{r^9} \right] \tag{125}$$

$$A_{J2,53} = -\frac{3}{2} J_2 R_{\oplus}^2 y \left[\left(1 - \frac{5z^2}{r^2}\right)\left(-\frac{5z}{r^7}\right) + \left(\frac{10z}{r^7}\right)\left(\frac{z^2}{r^2} - 1\right) \right] \tag{126}$$

$$A_{J2,61} = -\frac{3}{2} J_2 R_{\oplus}^2 z \left[\left(3 - \frac{5z^2}{r^2}\right)\left(-\frac{5x}{r^7}\right) + \frac{10xz^2}{r^9} \right] \tag{127}$$

$$A_{J2,62} = -\frac{3}{2} J_2 R_{\oplus}^2 z \left[\left(3 - \frac{5z^2}{r^2}\right)\left(-\frac{5y}{r^7}\right) + \frac{10yz^2}{r^9} \right] \tag{128}$$

$$A_{J2,63} = -\frac{3}{2} J_2 R_{\oplus}^2 \left[\left(3 - \frac{5z^2}{r^2}\right)\left(\frac{1}{r^5} - \frac{5z^2}{r^7}\right) + \left(\frac{10z^2}{r^7}\right)\left(\frac{z^2}{r^2} - 1\right) \right] \tag{129}$$

3.5.3 Equations of Variation for Third-Body Gravitational Effects

The equations of variation for third-body gravitational effects are also only position-dependent. In this case, care must be taken to use the appropriate gravitational parameter for the third-body of interest, μ_{3-body}, and not that of Earth. The individual component equations are

$$A_{3-body,41} = -\mu_{3-body}\left(\frac{1}{r^3} - 3\frac{x^2}{r^5}\right) \tag{130}$$

$$A_{3-body,42} = \frac{3\mu_{3-body}xy}{r^5} \tag{131}$$

$$A_{3-body,43} = \frac{3\mu_{3-body}xz}{r^5} \tag{132}$$

$$A_{3-body,51} = \frac{3\mu_{3-body}xy}{r^5} \tag{133}$$

$$A_{3-body,52} = -\mu_{3-body}\left(\frac{1}{r^3} - 3\frac{y^2}{r^5}\right) \tag{134}$$

$$A_{3-body,53} = \frac{3\mu_{3-body} yz}{r^5} \tag{135}$$

$$A_{3-body,61} = \frac{3\mu_{3-body} xz}{r^5} \tag{136}$$

$$A_{3-body,62} = \frac{3\mu_{3-body} yz}{r^5} \tag{137}$$

$$A_{3-body,63} = -\mu_{3-body}\left(\frac{1}{r^3} - 3\frac{z^2}{r^5}\right) \tag{138}$$

3.5.4 Equations of Variation for Atmospheric Drag

For the models incorporated in this simulation, only atmospheric drag has terms that are velocity-dependent. For this reason, the equations of variation for atmospheric drag also populate the lower right 3 x 3 corner of the *A* matrix in addition to the lower left 3 x 3 corner. The eighteen equations of variation for atmospheric drag are

$$A_{drag,41} = -\frac{1}{2}\frac{c_D A}{m}\rho(\dot{x} + \omega_\oplus y)\left[-\frac{x v_{rel}}{Hr} - \frac{\omega_\oplus(\dot{y} - \omega_\oplus x)}{v_{rel}}\right] \tag{139}$$

$$A_{drag,42} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[-\frac{y v_{rel}(\dot{x} + \omega_\oplus y)}{Hr} + \frac{\omega_\oplus(\dot{x} + \omega_\oplus y)^2}{v_{rel}} + v_{rel}\omega_\oplus\right] \tag{140}$$

$$A_{drag,43} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[-\frac{z v_{rel}(\dot{x} + \omega_\oplus y)}{Hr}\right] \tag{141}$$

$$A_{drag,44} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[\frac{(\dot{x} + \omega_\oplus y)^2}{v_{rel}} + v_{rel}\right] \tag{142}$$

$$A_{drag,45} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[\frac{(\dot{x} + \omega_\oplus y)(\dot{y} - \omega_\oplus x)}{v_{rel}}\right] \tag{143}$$

$$A_{drag,46} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[\frac{\dot{z}\left(\dot{x}+\omega_\oplus y\right)}{v_{rel}}\right] \tag{144}$$

$$A_{drag,51} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[-\frac{x v_{rel}\left(\dot{y}-\omega_\oplus x\right)}{Hr}-\frac{\omega_\oplus\left(\dot{y}-\omega_\oplus x\right)^2}{v_{rel}}-v_{rel}\omega_\oplus\right] \tag{145}$$

$$A_{drag,52} = -\frac{1}{2}\frac{c_D A}{m}\rho\left(\dot{y}-\omega_\oplus x\right)\left[-\frac{y v_{rel}}{Hr}+\frac{\omega_\oplus\left(\dot{x}+\omega_\oplus y\right)}{v_{rel}}\right] \tag{146}$$

$$A_{drag,53} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[-\frac{z v_{rel}\left(\dot{y}-\omega_\oplus x\right)}{Hr}\right] \tag{147}$$

$$A_{drag,54} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[\frac{\left(\dot{x}+\omega_\oplus y\right)\left(\dot{y}-\omega_\oplus x\right)}{v_{rel}}\right] \tag{148}$$

$$A_{drag,55} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[\frac{\left(\dot{y}-\omega_\oplus x\right)^2}{v_{rel}}+v_{rel}\right] \tag{149}$$

$$A_{drag,56} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[\frac{\dot{z}\left(\dot{y}-\omega_\oplus x\right)}{v_{rel}}\right] \tag{150}$$

$$A_{drag,61} = -\frac{1}{2}\frac{c_D A}{m}\rho\dot{z}\left[-\frac{x v_{rel}}{Hr}-\frac{\omega_\oplus\left(\dot{y}-\omega_\oplus x\right)}{v_{rel}}\right] \tag{151}$$

$$A_{drag,62} = -\frac{1}{2}\frac{c_D A}{m}\rho\dot{z}\left[-\frac{y v_{rel}}{Hr}+\frac{\omega_\oplus\left(\dot{x}+\omega_\oplus y\right)}{v_{rel}}\right] \tag{152}$$

$$A_{drag,63} = -\frac{1}{2}\frac{c_D A}{m}\rho\dot{z}\left[-\frac{z v_{rel}}{Hr}\right] \tag{153}$$

$$A_{drag,64} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[\frac{\dot{z}\left(\dot{x}+\omega_\oplus y\right)}{v_{rel}}\right] \tag{154}$$

$$A_{drag,65} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[\frac{\dot{z}\left(\dot{y}-\omega_\oplus x\right)}{v_{rel}}\right] \tag{155}$$

$$A_{drag,66} = -\frac{1}{2}\frac{c_D A}{m}\rho\left[\frac{\dot{z}^2}{v_{rel}} + v_{rel}\right] \tag{156}$$

IV. Results and Analysis

4.1 Gibbs and Herrick-Gibbs Initial Orbit Determination Methods

Although a possible satellite tracking architecture employing a system of systems that allowed satellite tracking at all ranges from low-Earth orbit (LEO) to geosynchronous (GEO) was suggested in Section 2.5, the most probable orbital region for a covert microsatellite rendezvous mission is low-Earth orbit (LEO). With this fact in mind, several cases based on a Defense Meteorological Satellite Program orbit at 830-km sun-synchronous altitude were studied.

First, the Gibbs and Herrick-Gibbs initial orbit determination methods were compared for performance accuracy for their common application area of in-plane angular separations between 1^{o} and 5^{o}. As previously stated in Section 1, the Gibbs method is preferred for larger angular separations, especially over 5^{o}, while the Herrick-Gibbs method is preferred for smaller angular separations. Simulated tracking data consisting of range, azimuth, and elevation measurements was generated using the accuracy numbers of both the Eglin Spacetrack radar and the Russian Don-2M anti-ballistic missile radar and corrupted with a random number generator to simulate process noise. The initial state vector used as the truth model had an epoch of April 5, 2003 at 00:00:00.00 UCT given by

$$\vec{X}_{TRUTH} = \begin{bmatrix} 1602.648663km \\ -7027.713446km \\ 0.0km \\ -1.101099km/s \\ -0.251102km/s \\ 7.350048km/s \end{bmatrix}$$

The simulated data for the Don-2M radar assumed a range accuracy of 200 meters and 0.03° in both azimuth and elevation. Using equation (30) from Section 3, this data was used to calculate x, y, and z position coordinates in the SEZ-frame and then transformed to the IJK-frame. For the case of 1° of angular separation, the three position vectors used in both the Gibbs and Herrick-Gibbs methods are given in Table 3.

Table 3 Position Vectors from the Don-2M Radar Separated by 1°

	x (km)	y (km)	z (km)
r_1	1,630.53547	-7,019.29836	-119.05177
r_2	1,599.46588	-7,028.79482	15.66486
r_3	1,571.1354	-7,034.33703	147.6602

Remembering that both the Gibbs and Herrick-Gibbs methods both determine the velocity associated with the second position vector, r_2, their results are shown in Table 4.

Table 4 State Vectors Separated by 1° from Don-2M Radar Data

	x (km)	y (km)	z (km)	$\dot{x}$ (km/s)	$\dot{y}$ (km/s)	$\dot{z}$ (km/s)
Gibbs	1,599.46588	-7,028.79482	15.66486	-1.02787	-0.25881	4.61946
Herrick-Gibbs	1,599.46588	-7,028.79482	15.66486	-1.65012	-0.41776	7.40921

The same type of data was generated for the Eglin Spacetrack radar with assumed accuracies of 5 meters in range and 0.0154° in azimuth and 0.0147° in elevation. The position vectors generated from simulated Eglin data are given in Table 5 and the results of the Gibbs and Herrick-Gibbs methods using this data are shown in Table 6.

Table 5 Position Vectors from the Eglin Spacetrack Radar Separated by 1°

	x (km)	y (km)	z (km)
r_1	1,629.78305	-7,022.51937	-116.51275
r_2	1,599.99858	-7,028.25713	14.87741
r_3	1,569.93708	-7,032.14408	146.52768

Table 6 State Vectors Separated by 1° from Eglin Spacetrack Radar Data

	x (km)	y (km)	z (km)	$\dot{x}$ (km/s)	$\dot{y}$ (km/s)	$\dot{z}$ (km/s)
Gibbs	1,599.99858	-7,028.25713	14.87741	-1.91216	-0.30763	8.40459
Herrick-Gibbs	1,599.99858	-7,028.25713	14.87741	-1.66250	-0.26738	7.30721

The differences between the truth state vector and the two radar state vectors are shown in Table 7.

Table 7 Difference Between Truth and Radar State Vectors for 1° Separation

	x (km)	y (km)	z (km)	$\dot{x}$ (km/s)	$\dot{y}$ (km/s)	$\dot{z}$ (km/s)
Don-2M						
Gibbs	-3.182783	-1.081374	15.66486	0.073229	-0.007708	-2.730588
Herrick-Gibbs	-3.182783	-1.081374	15.66486	-0.549021	-0.166658	0.059162
Eglin						
Gibbs	-2.650083	0.53769	14.87741	-0.811061	-0.056528	1.054542
Herrick-Gibbs	-2.650083	0.53769	14.87741	-0.561401	-0.016278	-0.042838

From the data in Table 7, several facts can be noted. For an angular separation of approximately 1°, the Herrick-Gibbs is more accurate than the Gibbs method for both the Don-2M and Eglin radars as expected. Although, the same r_2 position vector was used in the Gibbs and Herrick-Gibbs methods, the difference is in the velocity error magnitudes. The velocity error magnitude for theDon-2M radar using the Gibbs method is 2.731 km/s while the Herrick-Gibbs method velocity error magnitude is only 0.576 km/s, roughly 21% of the Gibbs method. For the more accurately measuring Eglin radar, the velocity error magnitude using the Gibbs method is 1.331 km/s compared to only 0.563 km/s for the Herrick-Gibbs method. The Herrick-Gibbs method, in this case, is 2.36 times better.

The same type of comparison was made between the two methods and the two radars using position vectors separated by approximately 5^o. The position vectors calculated from the simulated Don-2M data are shown in Table 8.

Table 8 Position Vectors from the Don-2M Radar Separated by 5^o

	x (km)	y (km)	z (km)
r_1	1,686.49254	-6,982.79171	-601.78566
r_2	1,599.46588	-7,028.79482	15.66486
r_3	1,501.82853	-7,019.49588	629.85516

The position vectors resulting from the Gibbs and Herrick-Gibbs methods are given in Table 9.

Table 9 State Vectors Separated by 5^o from Don-2M Radar Data

	x (km)	y (km)	z (km)	$\dot{x}$ (km/s)	$\dot{y}$ (km/s)	$\dot{z}$ (km/s)
Gibbs	1,599.46588	-7,028.79482	15.66486	-1.07891	-0.21268	7.19367
Herrick-Gibbs	1,599.46588	-7,028.79482	15.66486	-1.10055	-0.21881	7.34038

Similarly, the position vectors calculated from the simulated Eglin data are shown in Table 10 and the state vectors determined from these vectors are given in Table 11.

Table 10 Position Vectors from the Eglin Spacetrack Radar Separated by 5^o

	x (km)	y (km)	z (km)
r_1	1,684.70942	-6,982.28071	-601.80866
r_2	1,599.99858	-7,028.25713	14.87741
r_3	1,502.43363	-7,021.71168	631.71985

Table 11 State Vectors Separated by 5° from Eglin Spacetrack Radar Data

	x (km)	y (km)	z (km)	$\dot{x}$ (km/s)	$\dot{y}$ (km/s)	$\dot{z}$ (km/s)
Gibbs	1,599.99858	-7,028.25713	14.87741	-1.08650	-0.23523	7.35306
Herrick-Gibbs	1,599.99858	-7,028.25713	14.87741	-1.08633	-0.23499	7.35163

Finally, the difference between the truth model vector and the calculated state vectors is shown in Table 12.

Table 12 Difference Between Truth and Radar State Vectors for 5°

	x (km)	y (km)	z (km)	$\dot{x}$ (km/s)	$\dot{y}$ (km/s)	$\dot{z}$ (km/s)
Don-2M						
Gibbs	-3.182783	-1.081374	15.66486	0.022189	0.038422	-0.156378
Herrick-Gibbs	-3.182783	-1.081374	15.66486	0.000549	0.032292	-0.009668
Eglin						
Gibbs	-2.650083	0.53769	14.87741	0.014599	0.015872	0.003012
Herrick-Gibbs	-2.650083	0.53769	14.87741	0.014769	0.016112	0.001582

Similar to the 1° case, several facts can be noted from Table 12. Although the Gibbs method is expected to give more accurate results because of the larger angular separation, it actually produces more error in the velocity components for the vectors for the Don-2M radar. The Gibbs method velocity magnitude error for the Don-2M is 0.162 km/s while the Herrick-Gibbs method velocity error magnitude is only 0.0337 km/s, a factor of 4.82 better. For the Eglin radar, the Gibbs method is more accurate as would normally be expected but only slightly. The Gibbs method velocity error is 0.0217 km/s and the Herrick-Gibbs method error is 0.0219 km/s, less than 1% difference. Finally, the more accurately measuring Eglin radar produces the more accurate estimate of position and velocity.

4.2 Non-linear Least Squares Orbit Determination Filter

This section discusses two illustrative examples about the performance of the non-linear least squares orbit determination filter. First, Table 13 shows the effect on accuracy of the estimated state vector with an increasing number of data points. The first vector listed was used as the truth model of an 830-km altitude, sun-synchronous orbit to generate 10, 20, 30, 40, and 100 data points at 60-second intervals using only simple two-body motion and not including any perturbation forces. The second vector listed is the initial estimate of the target satellite's state vector used in the non-linear least squares orbit determination filter. The position vector components x, y, and z were each displaced 3.0 km to simulate a 5.2 km error in the knowledge of the target's position. Velocity components were not perturbed. In this rather simple case, it can be seen that the target's estimated converges closer to the truth model quickly with just 20 points but

Table 13 Comparison of Estimated State Vectors Based on Increasing Number of Data Points

	x (km)	y (km)	z (km)	$\dot{x}$ (km/s)	$\dot{y}$ (km/s)	$\dot{z}$ (km/s)
Truth (data)	1,602.648	-7,027.71	0.0	-1.101099	-0.251102	7.350048
Filter Estimate	1,599.649	-7,030.71	3.0	-1.011099	-0.251102	7.350048
10 data Points	1,586.364	-7,027.96	1.180124	-1.09513	-0.245541	7.343681
20 data Points	1,602.496	-7,027.81	-0.02949	-1.10077	-0.251194	7.350026
30 data points	1,602.63	-7,027.69	-0.00197	-1.10112	-0.251066	7.350052
40 data points	1,602.644	-7,027.74	-0.00205	-1.10105	-0.251136	7.350043
100 data points	1,602.643	-7,027.71	-0.00131	-1.1011	-0.251101	7.350048

is still approximately 184 meters in error. With 40 data points, the filter has converged within approximately 30 meters of the truth model. At 100 data points, the estimated state vector has converged to within approximately 5 meters of the truth model. Obviously, more data is always desired from a standpoint of increased accuracy; however, in a rendezvous mission, time to collect data may not be available and rendezvous maneuvers will have to be planned based on less accurate position estimates. The convergence criteria for all of these cases was each state component must be within 5 percent its variance as computed in the covariance matrix, P. Within the 6 x 6 covariance matrix, the variances , σ_i^2 , are the diagonal entries corresponding to the particular state vector component *i*. Taking the square root of the variance gives the standard deviation which then establishes an upper and lower bound on the vector component

$$(X_i - \sigma_i) < X_i < (X_i + \sigma_i)$$

Table 14 shows the variances for the 10, 20, 30, 40, and 100 data point cases above.

Table 14 Comparison of Variances for 10, 20, 30, 40, and 100 Data Points

	σ_x^2 (km^2)	σ_y^2 (km^2)	σ_z^2 (km^2)	$\sigma_{\dot{x}}^2$ (km/s)^2	$\sigma_{\dot{y}}^2$ (km/s)^2	$\sigma_{\dot{z}}^2$ (km/s)^2
10 data Points	245.6172	0.467814	22.62718	1.180 E-6	2.3883 E-5	5.076 E-5
20 data Points	0.106890	0.022710	0.001767	3.576 E-8	1.0605 E-7	2.972 E-10
30 data points	0.0018319	0.003897	0.000108	2.144 E-8	8.5994 E-9	8.247 E-11
40 data points	0.0016032	0.0011008	9.1517 E-5	1.117 E-8	1.8390E-9	1.220 E-11
100 data points	1.1489 E-5	3.7707 E-7	1.7362 E-6	5.579 E-12	1.003E-12	2.214 E-13

The final example shows a comparison between a typical GPS semi-synchronous orbit modeled with and without third-body gravitational perturbations in the dynamics. In general, this example simply illustrates that perturbations increase the uncertainty in state vector estimate. Table 15 shows the state vector used to generate simulated data, the state vector used as the initial estimate to start the filter, and the estimated state vectors with and without the third-body perturbation. Table 16 shows the variances for these two cases.

Table 15 Comparison of GPS State Vectors with & without Third-Body Perturbation

	x (km)	y (km)	z (km)	$\dot{x}$ (km/s)	$\dot{y}$ (km/s)	$\dot{z}$ (km/s)
Truth data	-5,522.5788	25,981.690	0.0	-2.173367	0.461963	3.173233
Filter Start	-5,529.0979	25,980.302	9.519979	-2.173014	0.463621	3.173232
Without 3-body Perturbation	-5,524.102	25,981.601	-1.00242	-2.173227	0.461914	3.173383
With 3-body Perturbation	-5,526.1636	25,981.523	-2.20056	-2.173041	0.461843	3.173551

Table 16 Variances for GPS Orbit with and without Third-Body Perturbation

	σ_x^2 (km^2)	σ_y^2 (km^2)	σ_z^2 (km^2)	$\sigma_{\dot{x}}^2$ (km/s)^2	$\sigma_{\dot{y}}^2$ (km/s)^2	$\sigma_{\dot{z}}^2$ (km/s)^2
Without 3-body Perturbation	4.73205	0.028302	1.861128	3.7143 E-8	4.4585 E-9	4.1236 E-8
With 3-body Perturbation	8.93414	0.121836	2.702021	8.3335 E-8	3.9442 E-9	7.1389 E-8

V. Conclusions and Recommendations

The objective of this thesis was to investigate the feasibility of a technologically unsophisticated adversary implementing a "low-tech" satellite tracking system architecture and orbit determination program to perform a covert microsatellite rendezvous with a larger uncooperative target. The open-source literature review investigated the types of tracking sensors and their representative accuracies. These "real world" accuracy values were then used in a non-linear least squares orbit determination filter. Since the basis of this thesis was a simulation experiment which involved programming a non-linear least squares orbit determination filter using simulated data, it should come as no surprise the filter converges to a solution assuming that the equations of motion, equations of variation, and other supporting subroutines were properly developed and coded in the MATLAB® program. The real test of the filter would be to deploy a real-world sensor, take satellite observations with that sensor, then process the data through the non-linear least squares filter.

The most probable orbital location for a hostile, covert, microsatellite rendezvous mission is low-Earth orbit (LEO) given the difficulty of detecting and tracking the microsatellite at mid-Earth orbit (MEO) and geosynchronous (GEO) altitudes. The current assessment, based on open-source information, is that neither China nor any other foreign country possesses an operational microsatellite anti-satellite weapon. Assuming that a potential adversary could acquire or develop the technology to design, build, and launch a microsatellite, and track it with sufficient accuracy, the overall conclusion is that someday some organization will be able to perform a microsatellite rendezvous with a non-cooperative target.

Further work which could be pursued in relation to this thesis would be to develop a Kalman filter to allow for real-time processing of observation data such as range and range rate for the orbit determination and updating processes. While the non-linear least squares orbit determination filter does converge to a solution, it performs best with large numbers of observations which take more collection time and thus delays processing. Another approach to the orbit determination problem would be to investigate an architecture where orbit determination is done solely on the ground and the maneuver commands are uplinked to the microsatellite in order for it to rendezvous with the target. This model would relieve the requirement for a precise tracking device on-board the microsatellite.

Appendix A.

```
% On_Orbit_Non_Linear_Least_Squares_Filter

% Capt Brian L. Foster

% 22 January 2003

format long g


% Data type 1 = range only; data type 2 = range and range-rate

data_type = 1;


% Open output files

fid1 = fopen('on_orbit_sat_positions_output.txt','w+');

fid2 = fopen('on_orbit_range_and_rate_residuals_output.txt','w+');

fid3 = fopen('on_orbit_state_and_state_corrections_output.txt','w+');

fid4 = fopen('target_reference_trajectory_output.txt','w+');

fid5 = fopen('on_orbit_range_only_output.txt','w+');

fid6 = fopen('on_orbit_covariance_matrix.txt','w+');


% Read in observations from data file

% Range only data

if(data_type == 1)

load('on_orbit_range_only_data.txt','-ascii');

[ob_type,order_ob,JDay,ob_time,range_ob]...
```

```
    =textread('on_orbit_range_only_data.txt','%d %d %f %f %f',-1);
end
% Range and range-rate data
if(data_type == 2)
load('on_orbit_range_and_rate_data.txt','-ascii');
[ob_type,order_ob,JDay,ob_time,range_ob,range_rate_ob]...
    =textread('on_orbit_range_and_rate_data.txt','%d %d %f %f %f %f',-1);
end


% Determine the number of observations which will determine the number
% of times through the data processing loop.
num_obs = length(ob_type);


% Initial guess (estimate) of state vector for the target satellite
r_tgt(1) = 1605.648663;
r_tgt(2) = -7030.713446;
r_tgt(3) = 3.0;
v_tgt(1) = -1.101099;
v_tgt(2) = -0.251102;
v_tgt(3) = 7.350048;


% Initial position of the microsatellite
r_micro(1) = 1597.121868;
r_micro(2) = -7028.875456;

r_micro(3) = 36.750077;

v_micro(1) = -1.109613;

v_micro(2) = -0.213701;

v_micro(3) = 7.349950;


% Initialize the state vectors for the target and the microsatellite.

% The state vector is a 6 x 1 column vector.

% X(1) = I component of position vector r in the IJK coordinate system

% X(2) = J component of position vector r in the IJK coordinate system

% X(3) = K component of position vector r in the IJK coordinate system

% X(4) = I component of velocity vector v in the IJK coordinate system

% X(5) = J component of velocity vector v in the IJK coordinate system

% X(6) = K component of velocity vector v in the IJK coordinate system


% Initial guess (estimate) of state vector for the microsat in column vector form

X_micro_ref = [r_micro(1); r_micro(2); r_micro(3); v_micro(1);...

      v_micro(2); v_micro(3)];


X_micro = X_micro_ref;


% Initial guess (estimate) of state vector for the target satellite in column vector form

X_tgt_ref = [r_tgt(1); r_tgt(2); r_tgt(3); v_tgt(1); v_tgt(2); v_tgt(3)];
```

```
% Initialize state corrections to 0
del_X_tgt = [0; 0; 0; 0; 0; 0; 0];

iteration = 0;

fprintf(fid3,'%3d %15.6f %15.6f %15.6f %15.6f %15.6f %15.6f %15.6f %15.6f %15.6f
%15.6f %15.6f %15.6f\n', iteration, X_tgt_ref(1), X_tgt_ref(2), X_tgt_ref(3),
X_tgt_ref(4), X_tgt_ref(5), X_tgt_ref(6), del_X_tgt(1), del_X_tgt(2), del_X_tgt(3),
del_X_tgt(4), del_X_tgt(5), del_X_tgt(6));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Flags to turn on ( xxxx_flag = 1) or off (xxxx_flag = 0)  perturbations.
% J2 is the second zonal gravity harmonic
% drag can be calculated for altitudes up to 1,000 km with density
%   calculated in function 'atmosphere.'
% third-body includes gravtitational effects of the Sun and Moon
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
J2_flag = 0;
drag_flag = 0;
third_body_flag = 0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Satellite parameters needed for estimating atmospheric drag.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tgt_drag_coefficient = 1.0;     % Dimensionless
tgt_sat_mass = 1000.0;          % Kilograms
tgt_sat_area = 1.0;             % Square meters
micro_drag_coefficient = 1.0;   % Dimensionless
micro_sat_mass = 100.0;         % Kilograms
micro_sat_area = 0.1698;        % Square meters


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Set up z, the total data (observation) vector.
% The "order" of z is the number of "types" of data
% associated with a single observation time.  For example,
% if processing range and range-rate then the order is 2.
% The dimension of z is (number of obs) x (order).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Set maximum number of iterations for the filter to loop through.
max_iter = 20;
iteration = 1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Begin iteration loop for Non-Linear Least Squares
while iteration <= max_iter


   % "Mode" value is the flag for deciding whether only the equations
```

```
  % of motion (EOM)(mode = 0) or EOM and equations of

  % variation (EOM + EOV)(mode = 1) are processed in subroutine "rhs"

 % which provides the differential equations to be integrated.

 mode = 1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 % Initialize the "total" state vector.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

   % "n" is the number of equations to be integrated.

  % 42 is the total number of equations. 6 for the state components

  % plus 36 for the components of the state transition matrix, phi.

  n = 42;


  % Initialize the state transition matrix for the target,

  % phi_target, to the identity matrix.

  phi_tgt = eye(6);


  % If mode not equal to 1, the totatl state vector is only the

  % target satellite's position and velocity.

  if(mode ~= 1)

      X_tgt = [X_tgt_ref(1); X_tgt_ref(2); X_tgt_ref(3); X_tgt_ref(4);...

        X_tgt_ref(5); X_tgt_ref(6)];

  end
```

```
% If mode is equal to 1, the total state vector is the target

% satellite's position and velocity and its state transition

% matrix.  Formed as 42 by 1 column vector since ode45 expects

% a column vector.


if(mode == 1)

   X_tgt = [X_tgt_ref(1); X_tgt_ref(2); X_tgt_ref(3); X_tgt_ref(4);...

      X_tgt_ref(5); X_tgt_ref(6); phi_tgt(1,1); phi_tgt(1,2);...

      phi_tgt(1,3); phi_tgt(1,4); phi_tgt(1,5); phi_tgt(1,6);...

      phi_tgt(2,1); phi_tgt(2,2); phi_tgt(2,3); phi_tgt(2,4);...

      phi_tgt(2,5); phi_tgt(2,6); phi_tgt(3,1); phi_tgt(3,2);...

      phi_tgt(3,3); phi_tgt(3,4); phi_tgt(3,5); phi_tgt(3,6);...

      phi_tgt(4,1); phi_tgt(4,2); phi_tgt(4,3); phi_tgt(4,4);...

      phi_tgt(4,5); phi_tgt(4,6); phi_tgt(5,1); phi_tgt(5,2);...

      phi_tgt(5,3); phi_tgt(5,4); phi_tgt(5,5); phi_tgt(5,6);...

      phi_tgt(6,1); phi_tgt(6,2); phi_tgt(6,3); phi_tgt(6,4);...

      phi_tgt(6,5); phi_tgt(6,6)];

end

 % Verify X is a 42 x 1 column vector.

X_tgt_size = size(X_tgt);


% Re-initialize the microsatellite's state vector to the beginning

% for each iteration or else the range will diverge with each
```

```
    % successive iteration.
    X_micro = X_micro_ref;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Initialize buffers for matrix product accumulation.
    % The matrices used in this program are:
    %   phi - state transition matrix (6 x 6)
    %   H   - observation model (order_obs x 6)
    %   T   - observation matrix; product of H * phi; (order_obs x 6)
    %   Q   - instrument covariance matrix (order_obs x order_obs)
    %   r   - residual vector
    %   P   - state covariance matrix (6 x 6)


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % For product of (T transpose) * (Q inverse) *   (r)
    % Dimensions:      (6 x n)     *   (n x n)   * (n x 1) = (6 x 1)
    T_tran_Q_inv_r = zeros(6,1);


    % Initialize state covariance matrix inverse (6 x 6)
    P_inv = zeros(6);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 % Observation (measurement data) processing loop
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 for iob = 1:num_obs
     % Write iteration and observation numbers
     % to screen for progress monitoring.
      fprintf('Iteration %d of %d\n',iteration,max_iter)
      fprintf('%d of %d observations is processing.\n', iob, num_obs)
      fprintf('\n')  % Write blank line to screen for spacing.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
     % Numerically integrate state and state transition matrix
     % derivatives to observation time.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Time "vector" to pass to integration routine ode45.
     if(iob == 1)
         time_vec = 0:ob_time(1);
     end

     if(iob > 1)
         time_step = ob_time(iob) - ob_time(iob-1);
         time_vec = 0:time_step;
```

```
end

% Also establish the Julian Date to pass on to function 'rhs' for
% third-body perturbations calculation.
JD = JDay(iob);

% Set absolute error tolerance for ode45 function for the target
% satellite.  Must match the target's state column vector size.
% IMPORTANT: Dr. Tragesser recommends the error tolerance be
% very tight, 1 x e-8 or smaller such as 1 x e-10.
abs_tol = 1e-8 * ones(42,1);

% Set options for ode45, including relative error tolerance.
options = odeset('RelTol', 1e-8, 'AbsTol', abs_tol);

% ode45 is one of MATLAB's built-in numerical integrators.  It is
% based on an explicit Runge-Kutta (4,5) formula, the Dormand-Prince
% pair.  It is a one-step solver in computing X(t), it needs only
% the solution at the immediately preceding time point, X(t n-1).

% Format of the integration routine call:
%   @rhs is the function containing the equations to be integrated.
%   time_vec is the time span to be integrated over.
```

```
        %   X is the current state of the system (initial conditions).
        %   options contain the information for absolute/relative
        %   tolerances, etc.
        % This mode statement MUST be here in order to alternate between
        % the target satellite and the microsatellite.
        mode = 1;

        % NOTE: at this point mode =1 because the state transition matrix
        % for the target must be integrated since it is the target's
        % state we are trying to estimate with the filter and not the microsatellite.
        [t,Y_tgt] = ode45(@on_orbit_rhs, time_vec, X_tgt,
options,mode,JD,third_body_flag, J2_flag, drag_flag, tgt_drag_coefficient,
tgt_sat_mass,tgt_sat_area);

        % The state of X_micro does not need to go through the equations of
        % variation since we are not estimating the microsatellites orbit.
        % Thus, mode = 0 and the state is a 6 x 1 column vector.
        mode = 0;
        abs_tol = 1e-8 * ones(6,1);
        options = odeset('RelTol', 1e-8, 'AbsTol', abs_tol);

% Propagate the microsatellite's state vector.
```

```
[t,Y_micro] = ode45(@on_orbit_rhs, time_vec, X_micro,
options,mode,JD,third_body_flag,...
   J2_flag, drag_flag,micro_drag_coefficient,micro_sat_mass,micro_sat_area);
      % ode45 returns a matrix that is of dimensions
      % (# of times steps x # of equations integrated)
      Y_tgt_ode_size = size(Y_tgt);
      Y_micro_ode_size = size(Y_micro);

      % Determine the length of the state matrices.
      last_row_tgt = Y_tgt_ode_size(1);
      last_row_micro = Y_micro_ode_size(1);

      % Extract only the last time step (row) values of state X
      % because ode45 expects a 42 component column vector
      % instead of large matrix that would be passed next time.
      % last row, the ':' means all columns associated with that row
      X_micro = Y_micro(last_row_micro,:);
      X_tgt = Y_tgt(last_row_tgt,:);

      % Write the target and microsatellite position vectors to output file
      fprintf(fid1,'%15.5f %15.6f %15.6f %15.6f %15.6f %15.6f %15.6f\n',...
      ob_time(iob),X_tgt(1),X_tgt(2),X_tgt(3),X_micro(1),X_micro(2),X_micro(3));
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Read observation data for this particular observation time.
    % These are the 'real' measured data.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if(data_type == 1)
      z_obs = [range_ob(iob)];
    end

    if(data_type == 2)
      z_obs = [range_ob(iob); range_rate_ob(iob)];
    end

   % Form the satellites' position vectors.
    r_tgt = [X_tgt(1); X_tgt(2); X_tgt(3)];
    r_micro = [X_micro(1); X_micro(2); X_micro(3)];

    % Form the satellites' velocity vectors.
    v_tgt = [X_tgt(4); X_tgt(5); X_tgt(6)];
    v_micro = [X_micro(4); X_micro(5); X_micro(6)];
```

```
      % Call to function 'obser' to get the predicted data vector,
      % which is based on the current states (position and velocity vectors).
      % zpred, H matrix, Q_inv matrix.
      [zpred,H,Q_inv] = obser(r_tgt,v_tgt,r_micro,v_micro,data_type);
       zpred_size = size(zpred);
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      % Begin the matrix calculations for this observation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      % Initialize the residual rejection flag.
      rejected = 0;

      % Calculate the residuals vector.  Residuals are the
      % difference of where we think the target satellite is
      % and where the observations say it is.
        if(data_type == 1)
        r = [z_obs(1,1) - zpred(1,1)];

      % Write residuals to screen
      fprintf('Range residual: %f kilometers.\n',r(1,1))
      fprintf('\n')  % Blank line for spacing.
```

```
    end




    if(data_type == 2)
        r = [z_obs(1,1) - zpred(1,1); z_obs(2,1) - zpred(2,1)];

    % Write residuals to screen
    fprintf('Range residual: %f kilometers.\n',r(1,1))
    fprintf('Range-rate residual: %f kilometers/second\n',r(2,1))
    fprintf('\n')  % Blank line for spacing.
    end
    residual_vector_size = size(r);
ndata = length(r);
   reject = 30000.0;

       for i = 1:ndata
           % Compare the elements of r(i) with its corresponding
           % diagonal entry of the Q_inv matrix.
           if(abs(r(i,1)) > reject/sqrt(Q_inv(i,i)))
               % Set residual rejection flag to sort/omit rejected obs.
               rejected = 1;
           end
```

```
  end  % End for i = 1:ndata



% If the observation is not rejected, process its matrices.

% Check if 'rejected' is anything other than 1 (not equal to 1).

if (rejected ~= 1)


% Extract the target satellite's phi matrix in normal form

% from the 'total' state column vector X_tgt.

phi = [X_tgt(7)  X_tgt(8)  X_tgt(9)  X_tgt(10) X_tgt(11) X_tgt(12);

       X_tgt(13) X_tgt(14) X_tgt(15) X_tgt(16) X_tgt(17) X_tgt(18);

       X_tgt(19) X_tgt(20) X_tgt(21) X_tgt(22) X_tgt(23) X_tgt(24);

       X_tgt(25) X_tgt(26) X_tgt(27) X_tgt(28) X_tgt(29) X_tgt(30);

       X_tgt(31) X_tgt(32) X_tgt(33) X_tgt(34) X_tgt(35) X_tgt(36);

       X_tgt(37) X_tgt(38) X_tgt(39) X_tgt(40) X_tgt(41) X_tgt(42)];


 % Matrix dimension statements for debugging.

 % Remove ; at end of line to write to screen.

 H_size = size(H);

 phi_size = size(phi);


 % Form matrix product  T    =    H   *  phi

 % Dimensions:       (n x 6) = (n x 6)*(6 x 6), where
 % n = ndata = order_ob

 % T is the observation matrix.

 T = H * phi;

 T_size = size(T);


 % Form product P_inv = (T transpose)*(Q inverse)*(T)

 % This product is the "observability condition."  It

 % must be invertible for an estimate to exist.

 % Dimensions:   (6 x 6)= (6 x n)*(n x n)    *(n x 6)

 P_inv = P_inv + (T' * Q_inv * T);


 % State estimate covariance, P

 P = inv(P_inv);

 position_variance = sqrt(P(1,1) + P(2,2) + P(3,3));


 % Write observed and predicted range and range rate and

% residuals to output file.

if(data_type == 1)

fprintf(fid5,'%15.5f %14.8f %14.8f %14.8f %14.8f\n',...

   ob_time(iob),z_obs(1,1),zpred(1,1),r(1,1),position_variance);

end
```

```
% Write observed and predicted range and range rate and
    % residuals to output file.
    if(data_type == 2)
    fprintf(fid2,'%15.5f %14.8f %14.8f %14.8f %10.6f %10.6f %10.6f %14.8f\n',...
       ob_time(iob),z_obs(1,1),zpred(1,1),r(1,1),z_obs(2,1),...
       zpred(2,1),r(2,1),position_variance);
    end

     % Matrix dimension statements for debugging.
     Q_inv_size = size(Q_inv);
     T_size = size(T);
     T_trans_size = size(T');
     r_size = size(r);

     % Form product (T transpose)*(Q inverse)*(r)
     % Dimensions: (6 x 1) = (6 x 1) + (6 x n)*(n x n)*(n x 1)
     T_tran_Q_inv_r = T_tran_Q_inv_r + (T' * Q_inv * r);

    end  % End to go with check of rejected ~= 1.
    % Reset rejected flag to 0 so that the next observation will be evaluated.

    rejected = 0;

end  % End of loop for iob = 1:num_obs

    % Invert matrix H transpose Q inverse H to find covariance P

    % Dimensions: (6 x 6) = inv((6 x n)*(n x n)*(n x 6))

    %P = inv(H' * Q_inv * H);

    P = inv(P_inv);


    % Multiply P by T transpose Q inverse r to get correction

    % to the state vector.

    % Initialize state correction term, dx, to zero first

    del_X_tgt = zeros(6,1);


    % Dimensions: (6 x 1) = (6 x 6)*(6 x 1)

    del_X_tgt = del_X_tgt + P * T_tran_Q_inv_r;


    if((abs(del_X_tgt(1) > 0.05*abs(P(1,1)))...

          | (abs(del_X_tgt(2) > 0.05*sqrt(abs(P(2,2)))))...

          | (abs(del_X_tgt(3) > 0.05*sqrt(abs(P(3,3)))))...

          | (abs(del_X_tgt(4) > 0.05*sqrt(abs(P(4,4)))))...

          | (abs(del_X_tgt(5) > 0.05*sqrt(abs(P(5,5)))))...

          | (abs(del_X_tgt(6) > 0.05*sqrt(abs(P(6,6))))))
```

```
        % The vertical bar(s) in the above 'if' statement is/are
        % MATLAB's logical 'or' operator.

        convergence = 0
      else
        convergence = 1
      end

    % Add in state corrections to reference state (trajectory)
    % This for the state at EPOCH only.  NOT every time step.
    X_tgt_ref = X_tgt_ref + del_X_tgt

    % Write this iterations state correction dx to output file here.
    fprintf(fid3,'%3d %15.6f %15.6f %15.6f %15.6f %15.6f %15.6f %15.6f %15.6f
%15.6f %15.6f %15.6f %15.6f\n',...

iteration,X_tgt_ref(1),X_tgt_ref(2),X_tgt_ref(3),X_tgt_ref(4),X_tgt_ref(5),X_tgt_ref(6),
    del_X_tgt(1),del_X_tgt(2),del_X_tgt(3),del_X_tgt(4),del_X_tgt(5),del_X_tgt(6));

      if(convergence == 1)
          % Just add a number to get iterations to exceed
          % maximum iteration and exit the while loop;
          fprintf('Converged on iteration %d of %d\n',iteration,max_iter)
```

```
            iteration = max_iter + 5;
        end

    % Increment iteration value
    iteration = iteration + 1

    end  % End statement for the iterations to max_iter loop

% Write values to screen
X_tgt_ref
del_X_tgt
P

% Write the final covariance matrix components to output file here.
for i = 1:6
fprintf(fid6,'%25.20f %25.20f %25.20f %25.20f %25.20f %25.20f\n',...
    P(i,1),P(i,2),P(i,3),P(i,4),P(i,5),P(i,6));
end

% Principal error axes
% Extract the 3 x 3 space and velocity covariance submatrices
% Space covariance is the upper lefthand 3 x 3 of the P matrix
% Velocity covariance is the lower righthand 3 x 3 of the P matrix
```

```
for i = 1:3
   for j = 1:3
      space_P(i,j) = P(i,j);
      velocity_P(i,j) = P(i+3,j+3);
   end
end

% Eigenvector/value analysis of the covariance matrices
[V,D] = eig(space_P)
[W,E] = eig(velocity_P)

for i = 1:3
   raxis(i) = D(i,i);
   vaxis(i) = E(i,i);
end

for i = 1:3
   if(raxis(i) < 0.0)
      negative_r_axis = 'Space covariance has a negative value!'
   else
      raxis(i) = sqrt(raxis(i));
   end
end
```

```
for i = 1:3
   if(vaxis(i) < 0.0)
      negative_v_axis = 'Velocity covariance has negative value!'
   else
      vaxis(i) = sqrt(vaxis(i));
   end
end

% Write position and velocity principal error axes to the screen.
raxis
vaxis

% Close output files.
fclose(fid1);
fclose(fid2);
fclose(fid3);
fclose(fid4);
fclose(fid5);
fclose(fid6);
does_it_work = 'YES, FINALLY!';
```

```
% Call plot function

[plotted] = plot_residuals(data_type)

% End of on-orbit non-linear least squares filter.
```

Appendix B.

```
function dX = on_orbit_rhs(t,X,mode,JD,third_body_flag,J2_flag,...
    drag_flag,drag_coefficient,sat_mass,sat_area)

% Capt Brian L. Foster
% 27 January 2003

% This MATLAB code modeled after FORTRAN code written by
% Dr. William E. Wiesel for MECH 731 Modern Methods of
% Orbit Determination.

% This function calculates the equations of motion (EOM) and/or
% not and the equations of variation (EOV) for the problem of
% a spacecraft in orbit around the Earth.

% X is the 42-component 'total' state vector
%   X(1-3) are the x,y,z components of the position vector
%   X(4-6) are the x,y,z components of the velocity vector
%   X(7-42) are the (6 x 6) state transition matrix
%   stored row by row

% dX is the 42-component state vector derivatives
%   dX(1-3) are the x,y,z derivatives of position (velocity)

%   dX(4-6) are the x,y,z derivatives of velocity (acceleration)

%   dX(7-42) are the derivatives of the state transition matrix, phi dot


% Open output files for the various acceleration components

% The 'w+' instructs MATLAB that the file can be both read and written

% to and that any previous data in the file is overwritten.

fid1 = fopen('gravity_accleration_output.txt','w+');

fid2 = fopen('J2_acceleration_output.txt','w+');

fid3 = fopen('drag_acceleration_output.txt','w+');

fid4 = fopen('totatl_acceleration_output.txt','w+');


% Earth radius, RE, in kilometers

RE = 6378.1363;


% Earth gravitational parameter, mu, in km^3/sec^2

mu_earth = 398600.4415;


% The N-Body Problem with the origin at the center of the Earth.

% Reference Vallado pages 116-119 or Bate, Mueller, and White page 10.


% Position derivatives = velocity

dX(1) = X(4);
dX(2) = X(5);

dX(3) = X(6);


% Velocity derivatives = gravity acceleration due to the Earth

r_vector = [X(1); X(2); X(3)];

r = norm(r_vector);


f_earth(4) = - mu_earth*X(1)/r^3;

f_earth(5) = - mu_earth*X(2)/r^3;

f_earth(6) = - mu_earth*X(3)/r^3;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Calculate 3rd body perturbation accelerations, if desired.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


if(third_body_flag == 1)


   % Sun's gravitational parameter, km^3/s^2

   mu_sun = 1.32712428e11;


   % Call function 'Sun' for Sun's GEOCENTRIC position vector in km

   [r_sun] = Sun(JD);
```

```
% Vector from Sun to satellite

dx_sun = X(1) - r_sun(1);

dy_sun = X(2) - r_sun(2);

dz_sun = X(3) - r_sun(3);


% Distance from the Sun to the satellite cubed

r32_sun = (dx_sun^2 + dy_sun^2 + dz_sun^2)^(3/2);


% Distance from center of Earth (central body) to Sun cubed.

rp132_sun = (r_sun(1)^2 + r_sun(2)^2 + r_sun(3)^2)^(3/2);


% Acceleration terms due to Sun; 3rd body form of the equations

f_sun(4) = -mu_sun*(dx_sun/r32_sun - r_sun(1)/rp132_sun);

f_sun(5) = -mu_sun*(dy_sun/r32_sun - r_sun(2)/rp132_sun);

f_sun(6) = -mu_sun*(dz_sun/r32_sun - r_sun(3)/rp132_sun);


% Moon's gravitational parameter, km^3/s^2

mu_moon = 4902.799;


% Call function 'Moon' for Moon's GEOCENTRIC position vector in km

[r_moon] = Moon(JD);


% Vector from Moon to the satellite
```

```
  dx_moon = X(1) - r_moon(1);

  dy_moon = X(2) - r_moon(2);

  dz_moon = X(3) - r_moon(3);


  % Distance from the Moon to the satellite cubed

  r32_moon = (dx_moon^2 + dy_moon^2 + dz_moon^2)^(3/2);


  % Distance from center of Earth (central body) to the Moon cubed

  rp132_moon = (r_moon(1)^2 + r_moon(2)^2 + r_moon(3)^2)^(3/2);


  % Acceleration terms due to the Sun; 3rd body form of equations

  f_moon(4) = -mu_moon*(dx_moon/r32_moon - r_moon(1)/rp132_moon);

  f_moon(5) = -mu_moon*(dy_moon/r32_moon - r_moon(2)/rp132_moon);

  f_moon(6) = -mu_moon*(dz_moon/r32_moon - r_moon(3)/rp132_moon);


else

   f_sun(4)   = 0.0;

   f_sun(5)   = 0.0;

   f_sun(6)   = 0.0;

   f_moon(4)  = 0.0;

   f_moon(5)  = 0.0;

   f_moon(6)  = 0.0;
```

```
end     % 'end' statement to go with third body flag check


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate the perturbation of the Earth's oblateness due to J2.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if(J2_flag == 1)
   % J2 gravitational zonal coefficient from JGM-2 from Appendix D
   % of Vallado (1997).
   J2 = -0.1082626925638815e-2;


   % Second harmonic J2 terms, km/s^2
   f_J2(4) = -3*J2*mu_earth*(RE^2)*X(1)/(2*r^5)*(1-((5*X(3)^2)/r^2));


   f_J2(5) = -3*J2*mu_earth*(RE^2)*X(2)/(2*r^5)*(1-((5*X(3)^2)/r^2));


   f_J2(6) = -3*J2*mu_earth*(RE^2)*X(3)/(2*r^5)*(3-((5*X(3)^2)/r^2));


else
   f_J2(4) = 0.0;
   f_J2(5) = 0.0;
   f_J2(6) = 0.0;
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate the perturbation effect of atmospheric drag.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if(drag_flag == 1)
   % Earth rotational rate in rad/s.
   earth_rotation_rate = 0.000072921158553;

   % Calculate the satellite's velocity vector relative to the
   % Earth's rotating atmosphere.

   % Relative velocity, km/s.
   v_rel(1) = X(4) + earth_rotation_rate * X(2);
   v_rel(2) = X(5) - earth_rotation_rate * X(1);
   v_rel(3) = X(6);

   % Magnitude of relative velocity, km/s.
   v_rel_mag = norm(v_rel);

   % Determine altitude above Earth's surface, km.
   altitude = r - RE;
```

```
    % Call function 'atmosphere' to get atmospheric density.

    [density,scale_height] = atmosphere(altitude);


    % Drag acceleration terms.

    f_drag(4) = -0.5 * (drag_coefficient * sat_area / sat_mass)...

        * density * v_rel_mag * v_rel(1) * 1000.0;


    f_drag(5) = -0.5 * (drag_coefficient * sat_area / sat_mass)...

        * density * v_rel_mag * v_rel(2) * 1000.0;


    f_drag(6) = -0.5 * (drag_coefficient * sat_area / sat_mass)...

        * density * v_rel_mag * v_rel(3) * 1000.0;


else


    f_drag(4) = 0.0;

    f_drag(5) = 0.0;

    f_drag(6) = 0.0;


end  % 'end' statement to go with drag_flag check.


% Total acceleration for the equations of motion.

dX(4) = f_earth(4) + f_J2(4) + f_drag(4) + f_sun(4) + f_moon(4);
dX(5) = f_earth(5) + f_J2(5) + f_drag(5) + f_sun(5) + f_moon(5);
dX(6) = f_earth(6) + f_J2(6) + f_drag(6) + f_sun(6) + f_moon(6);


if(mode ~= 1)
   dX = [dX(1); dX(2); dX(3); dX(4); dX(5); dX(6)];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% EQUATIONS OF VARIATION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% If mode = 1, then the equations of variation are processed.
if(mode == 1)
   % Calculate the A matrix (A = gradient of vector f).
   % Initialize to 0 first.
   A = zeros(6,6);

   % A is a 6 x 6 matrix.
   % The upper right 3 x 3 corner is an identity matrix.
   A(1,4) = 1.0;
   A(2,5) = 1.0;
   A(3,6) = 1.0;

   % Diagonal terms of the A matrix lower left corner 3 x 3
```

```
A(4,1) = -mu_earth/r^3 + 3*mu_earth*X(1)^2/r^5;

A(5,2) = -mu_earth/r^3 + 3*mu_earth*X(2)^2/r^5;

A(6,3) = -mu_earth/r^3 + 3*mu_earth*X(3)^2/r^5;


% Off-diagonal terms of the A matrix lower left corner 3 x 3

% Use symmetry to avoid as much calculation as possible.

A(4,2) = 3*mu_earth*X(1)*X(2)/r^5;

A(5,1) = A(4,2);

A(4,3) = 3*mu_earth*X(1)*X(3)/r^5;

A(6,1) = A(4,3);

A(5,3) = 3*mu_earth*X(2)*X(3)/r^5;

A(6,2) = A(5,3);



% Equations of variation due to third body effects

if(third_body_flag == 1)


    % Sun's gravitational parameter, km^3/s^2

    mu_sun = 1.32712428e11;


    % Call function 'Sun' for Sun's GEOCENTRIC position vector in km

    [r_sun] = Sun(JD);
```

```
% Vector from Sun to satellite

dx_sun = X(1) - r_sun(1);

dy_sun = X(2) - r_sun(2);

dz_sun = X(3) - r_sun(3);


% Distance from the Sun to the satellite cubed

r32_sun = (dx_sun^2 + dy_sun^2 + dz_sun^2)^(3/2);


% Distance from the Sun to the satellite to fifth power

r52_sun = r32_sun^(5/3);


% Diagonal terms for the Sun

A_sun(4,1) = -mu_sun*(1/r32_sun - 3 * dx_sun^2/r52_sun);

A_sun(5,2) = -mu_sun*(1/r32_sun - 3 * dy_sun^2/r52_sun);

A_sun(6,3) = -mu_sun*(1/r32_sun - 3 * dz_sun^2/r52_sun);


% Sun's x and y terms

A_sun(4,2) = 3*mu_sun*dx_sun*dy_sun/r52_sun;

A_sun(5,1) = A_sun(4,2);


% Sun's x and z terms

A_sun(4,3) = 3*mu_sun*dx_sun*dz_sun/r52_sun;

A_sun(6,1) = A_sun(4,3);
```

```
% Sun's y and z terms

A_sun(5,3) = 3*mu_sun*dy_sun*dz_sun/r52_sun;

A_sun(6,2) = A_sun(5,3);


% Moon's gravitational parameter, km^3/s^2

mu_moon = 4902.799;


% Call function 'Moon' for Moon's GEOCENTRIC

% position vector in km

[r_moon] = Moon(JD);


% Vector from Moon to the satellite

dx_moon = X(1) - r_moon(1);

dy_moon = X(2) - r_moon(2);

dz_moon = X(3) - r_moon(3);


% Distance from the Moon to the satellite cubed

r32_moon = (dx_moon^2 + dy_moon^2 + dz_moon^2)^(3/2);


% Distance from the Moon to the satellite to fifth power

r52_moon = r32_moon^(5/3);
```

```
    % Diagonal terms for the Moon

    A_moon(4,1) = -mu_moon*(1/r32_moon - 3 * dx_moon^2/r52_moon);

    A_moon(5,2) = -mu_moon*(1/r32_moon - 3 * dy_moon^2/r52_moon);

    A_moon(6,3) = -mu_moon*(1/r32_moon - 3 * dz_moon^2/r52_moon);


    % Sun's x and y terms

    A_moon(4,2) = 3*mu_moon*dx_moon*dy_moon/r52_moon;

    A_moon(5,1) = A_moon(4,2);


    % Sun's x and z terms

    A_moon(4,3) = 3*mu_moon*dx_moon*dz_moon/r52_moon;

    A_moon(6,1) = A_moon(4,3);


    % Sun's y and z terms

    A_moon(5,3) = 3*mu_moon*dy_moon*dz_moon/r52_moon;

    A_moon(6,2) = A_moon(5,3);


    %third_body_EOV_status = 'Still going!'

else


    A_sun(4,1) = 0.0;

    A_sun(5,2) = 0.0;

    A_sun(6,3) = 0.0;
    A_sun(4,2) = 0.0;

    A_sun(5,1) = 0.0;

    A_sun(4,3) = 0.0;

    A_sun(6,1) = 0.0;

    A_sun(5,3) = 0.0;

    A_sun(6,2) = 0.0;


    A_moon(4,1) = 0.0;

    A_moon(5,2) = 0.0;

    A_moon(6,3) = 0.0;

    A_moon(4,2) = 0.0;

    A_moon(5,1) = 0.0;

    A_moon(4,3) = 0.0;

    A_moon(6,1) = 0.0;

    A_moon(5,3) = 0.0;

    A_moon(6,2) = 0.0;


 end


% Equations of variations due to J2

if(J2_flag == 1)


    % J2 gravitational zonal coefficient from JGM-2 from Appendix D
% of Vallado (1997).
J2 = -0.1082626925638815e-2;

A_J2(4,1) = -3/2*J2*mu_earth*RE^2*((1-5*X(3)^2)/r^2)*...
   (1/r^5 - 5*(X(1)^2/r^7) + 10 * (X(1)^2)*(X(3)^2)/r^9)

A_J2(4,2) = -3/2*J2*mu_earth*RE^2*X(1)*((-5*X(2)/r^7)*...
   (1-5*(X(3)^2)/r^2) + (10*X(2)*X(3)^2)/r^9);

A_J2(4,3) = -3/2*J2*mu_earth*RE^2*X(1)*((-5*X(3)/r^7)*...
   (1-5*(X(3)^2)/r^2) + (10*X(3)/r^7)*((X(3)^2)/r^2 -1));

A_J2(5,1) = -3/2*J2*mu_earth*RE^2*((-5*X(1)*X(2)/r^7)*...
   (1-5*(X(3)^2)/r^2) + (10*X(1)*X(2)*X(3)^2)/r^9);

A_J2(5,2) = - 3/2*J2*mu_earth*RE^2*((1-5*(X(3)^2)/r^2)*...
   (1/r^5 - 5*(X(2)^2)/r^7) + 10*(X(2)^2)*(X(3)^2)/r^9);

A_J2(5,3) = -3/2*J2*mu_earth*RE^2*X(2)*((-5*X(3)/r^7)*...
   (1-5*(X(3)^2)/r^2) + (10*X(3)/r^7)*((X(3)^2)/r^2 -1));

A_J2(6,1) = -3/2*J2*mu_earth*RE^2*X(3)*((-5*X(1)/r^7)*...
   (3-5*(X(3)^2)/r^2) + (10*X(1)*X(3)^2)/r^9);
```

```
    A_J2(6,2) = - 3/2*J2*mu_earth*RE^2*X(3)*((-5*X(2)/r^7)*...
        (3-5*(X(3)^2)/r^2) + (10*X(2)*X(3)^2)/r^9);

    A_J2(6,3) = -3/2*J2*mu_earth*RE^2*(((3-5*X(3)^2)/r^2)*...
        (1/r^5 - 5*(X(3)^2)/r^7) + (10*(X(3)^2)/r^7)*((X(3)^2)/r^2)-1);
else
    A_J2(4,1) = 0.0;
    A_J2(4,2) = 0.0;
    A_J2(4,3) = 0.0;
    A_J2(5,1) = 0.0;
    A_J2(5,2) = 0.0;
    A_J2(5,3) = 0.0;
    A_J2(6,1) = 0.0;
    A_J2(6,2) = 0.0;
    A_J2(6,3) = 0.0;

end

% Equations of variation due to atmospheric drag.
if(drag_flag == 1)
```

```
% Earth rotation rate, rad/s.
earth_rotation_rate = 0.000072921158553;

% Calculate the satellite's velocity vector relative to
% the Earth's rotating atmosphere.

% Relative velocity, km/s.
v_rel(1) = X(4) + earth_rotation_rate * X(2);
v_rel(2) = X(5) - earth_rotation_rate * X(1);
v_rel(3) = X(6);

% Magnitude of relative velocity, km/s.
v_rel_mag - norm(v_rel);

% Determine altitude above Earth's surface, km.
altitude = r - RE;

% Call function 'atmosphere' to get atmospheric density and
% scale height.
[density, scale_height] = atmosphere(altitude);
big_H = scale_height;

% Drag constant, DC, for easy programming
```

```
DC = -0.5 * drag_coefficient * sat_area / sat_mass;



A_drag(4,1) = DC*density *v_rel(1)*(-X(1)*v_rel_mag/(big_H * r) -...
   earth_rotation_rate*v_rel(2)/v_rel_mag)*1000.0;


A_drag(4,2) = DC*density*(-X(2)*v_rel_mag*v_rel(1)/(big_H * r) +...
   earth_rotation_rate/v_rel_mag*v_rel(1)^2 + ...
   v_rel_mag * earth_rotation_rate)*1000.0;
A_drag(4,3) = DC*density*(-X(3)*v_rel_mag*v_rel(1)/(big_H * r))*1000.0;
A_drag(4,4) = DC*density*((v_rel(1)^2)/v_rel_mag + v_rel_mag)*1000.0;
A_drag(4,5) = DC*density*((v_rel(1)*v_rel(2))/v_rel_mag)*1000.0;
A_drag(4,6) = DC*density*(v_rel(1)*v_rel(3)/v_rel_mag)*1000.0;
A_drag(5,1) = DC*density*(-X(1)*v_rel_mag*v_rel(2)/(big_H * r) -...
   earth_rotation_rate*(v_rel(2)^2)/v_rel_mag - ...
   earth_rotation_rate*v_rel_mag)*1000.0;
A_drag(5,2) = DC*density*(-X(2)*v_rel_mag*v_rel(2)/(big_H * r) +...
   earth_rotation_rate*v_rel(1)*v_rel(2)/v_rel_mag)*1000.0;
A_drag(5,3) = DC*density*(-v_rel_mag*v_rel(2)*X(3)/(big_H * r))*...
   1000.0;
A_drag(5,4) = DC*density*(v_rel(1)*v_rel(2)/v_rel_mag)*1000.0;
A_drag(5,5) = DC*density*((v_rel(2)^2)/v_rel_mag+v_rel_mag)*1000.0;
A_drag(5,6) = DC*density*(v_rel(3)*v_rel(2)/v_rel_mag)*1000.0;
```

```
    A_drag(6,1) = DC*density*(-X(1)*v_rel_mag*v_rel(3)/(big_H * r) -...
        earth_rotation_rate*v_rel(2)*v_rel(3)/v_rel_mag)*1000.0;
    A_drag(6,2) = DC*density*(-X(2)*v_rel_mag*v_rel(3)/(big_H * r) +...
        earth_rotation_rate*v_rel(1)*v_rel(3)/v_rel_mag)*1000.0;
    A_drag(6,3) = DC*density*(-X(3)*v_rel_mag*v_rel(3)/(big_H * r))...
        *1000.0;
    A_drag(6,4) = DC*density*(v_rel(1)*v_rel(3)/v_rel_mag)*1000.0;
    A_drag(6,5) = DC*density*(v_rel(2)*v_rel(3)/v_rel_mag)*1000.0;
    A_drag(6,6) = DC*density*(v_rel(3)^2/v_rel_mag+v_rel_mag)*1000.0;
else
    A_drag(4,1) = 0.0;
    A_drag(4,2) = 0.0;
    A_drag(4,3) = 0.0;
    A_drag(4,4) = 0.0;
    A_drag(4,5) = 0.0;
    A_drag(4,6) = 0.0;

    A_drag(5,1) = 0.0;
    A_drag(5,2) = 0.0;
    A_drag(5,3) = 0.0;
    A_drag(5,4) = 0.0;
    A_drag(5,5) = 0.0;
    A_drag(5,6) = 0.0;


    A_drag(6,1) = 0.0;
    A_drag(6,2) = 0.0;
    A_drag(6,3) = 0.0;
    A_drag(6,4) = 0.0;
    A_drag(6,5) = 0.0;
    A_drag(6,6) = 0.0;
end
% Sum the components.
% Diagonal terms.
A(4,1) = A(4,1) + A_J2(4,1) + A_drag(4,1) + A_sun(4,1) + A_moon(4,1);
A(5,2) = A(5,2) + A_J2(5,2) + A_drag(5,2) + A_sun(5,2) + A_moon(5,2);
A(6,3) = A(6,3) + A_J2(6,3) + A_drag(6,3) + A_sun(6,3) + A_moon(6,3);


% Off-diagonal terms.
A(4,2) = A(4,2) + A_J2(4,2) + A_drag(4,2) + A_sun(4,2) + A_moon(4,2);
A(5,1) = A(5,1) + A_J2(5,1) + A_drag(5,1) + A_sun(5,1) + A_moon(5,1);


A(4,3) = A(4,3) + A_J2(4,3) + A_drag(4,3) + A_sun(4,3) + A_moon(4,3);
A(6,1) = A(6,1) + A_J2(6,1) + A_drag(6,1) + A_sun(6,1) + A_moon(6,1);
```

```
A(5,3) = A(5,3) + A_J2(5,3) + A_drag(5,3) + A_sun(5,3) + A_moon(5,3);

A(6,2) = A(6,2) + A_J2(6,2) + A_drag(6,2) + A_sun(6,2) + A_moon(6,2);

% Equations of variation that are velocity related.

A(4,4) = A_drag(4,4);

A(4,5) = A_drag(4,5);

A(4,6) = A_drag(4,6);

A(5,4) = A_drag(4,4);

A(5,5) = A_drag(4,5);

A(5,6) = A_drag(4,6);

A(6,4) = A_drag(4,4);

A(6,5) = A_drag(4,5);

A(6,6) = A_drag(4,6);


% Extract phi matrix in normal form from the total state

% column vector X.

phi = [X(7) X(8) X(9) X(10) X(11) X(12);

   X(13) X(14) X(15) X(16) X(17) X(18);

   X(19) X(20) X(21) X(22) X(23) X(24);

   X(25) X(26) X(27) X(28) X(29) X(30);

   X(31) X(32) X(33) X(34) X(35) X(36);

   X(37) X(38) X(39) X(40) X(41) X(42)];
```

```
% Calculate the derivative of the state transition matrix, phi dot.
   phi_dot = A * phi;
   % Write the total state derivative as a column vector to return.
   dX = [dX(1); dX(2); dX(3); dX(4); dX(5); dX(6);...
         phi_dot(1,1); phi_dot(1,2); phi_dot(1,3); phi_dot(1,4);...
         phi_dot(1,5); phi_dot(1,6); phi_dot(2,1); phi_dot(2,2);...
         phi_dot(2,3); phi_dot(2,4); phi_dot(2,5); phi_dot(2,6);...
         phi_dot(3,1); phi_dot(3,2); phi_dot(3,3); phi_dot(3,4);...
         phi_dot(3,5); phi_dot(3,6); phi_dot(4,1); phi_dot(4,2);...
         phi_dot(4,3); phi_dot(4,4); phi_dot(4,5); phi_dot(4,6);...
         phi_dot(5,1); phi_dot(5,2); phi_dot(5,3); phi_dot(5,4);...
         phi_dot(5,5); phi_dot(5,6); phi_dot(6,1); phi_dot(6,2);...
         phi_dot(6,3); phi_dot(6,4); phi_dot(6,5); phi_dot(6,6)];

   dX_size = size(dX);
end
% Close output data files.
fclose(fid1);
fclose(fid2);
fclose(fid3);
fclose(fid4);
% End of on-orbit rhs function
```

Appendix C.

```
function [zpred,H,Q_inv] = obser(r_tgt,v_tgt,r_micro,v_micro,data_type)

% Capt Brian L. Foster

% 20 December 2002


% This MATLAB code modeled after FORTRAN code written by

% Dr. William E. Wiesel for MECH 731 Modern Methods of

% Orbit Determination.


% This subroutine performs the observation relation processing.

% It calculates the predicted observation, z_pred; H matrix; and

% returns the inverse of the data (instrument or measurements)

% covariance matrix, Q_inv.


format long g


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Data type: range and range-rate

% Relative position vector (3 x 1) (range) in IJK coordinates

% from the microsatellite (with the tracking sensor) to the

% target satellite.

range_vector = r_tgt - r_micro;
```

```
% Magnitude of range vector in IJK coordinates, kilometers

range = norm(range_vector);


% Relative velocity in IJK coordinates, in km/s

relative_velocity = v_tgt - v_micro;


% Magnitude of range rate in IJK, in km/s

range_rate = dot(range_vector,relative_velocity)/range;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Range only processing

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if(data_type == 1)


% Form z, predicted data vector. (2 x 1)

% Each component of zpred is a scalar.

zpred = [range];


% Form Q, the instrumental covariance matrix

Q = zeros(1,1);

Q(1,1) = 0.002^2;  % Instrumentation sigma squared ( 2 meters = 0.002 km)

Q_inv = inv(Q);
```

```
% Form H, the observation matrix, here.
% H matrix found on pages 75-76 of Wiesel and signs changed on
% row 2 in accordance with text on page 80 to account for the
% azimuth difference.
% H is a 2 x 6 matrix based on SEZ coordinates.
% Initialize H to zeros first then build up needed components.
H = zeros(1,6);

% Equations for range partial derivatives that change wrt position
H(1,1) = range_vector(1)/range;
H(1,2) = range_vector(2)/range;
H(1,3) = range_vector(3)/range;

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Range and range-rate processing
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if(data_type == 2)

% Form z, predicted data vector. (2 x 1)
% Each component of zpred is a scalar.
```

```
zpred = [range; range_rate];


% Form Q, the instrumental covariance matrix

Q = zeros(2,2);


Q(1,1) = 0.000004;

Q(2,2) = 0.000004;


Q_inv = inv(Q);


% Form H, the observation matrix, here.

% H matrix found on pages 75-76 of Wiesel and signs changed on

% row 2 in accordance with text on page 80 to account for the

% azimuth difference.

% H is a 2 x 6 matrix based on SEZ coordinates.

% Initialize H to zeros first then build up needed components.

H = zeros(2,6);


% Equations for range partial derivatives that change wrt position

H(1,1) = range_vector(1)/range;

H(1,2) = range_vector(2)/range;

H(1,3) = range_vector(3)/range;
```

```
% Equations for range-rate partial derivatives that change wrt
% position and velocity.
H(2,1) = relative_velocity(1)/range - range_rate*range_vector(1)/range^2;
H(2,2) = relative_velocity(2)/range - range_rate*range_vector(2)/range^2;
H(2,3) = relative_velocity(3)/range - range_rate*range_vector(3)/range^2;
H(2,4) = range_vector(1)/range;
H(2,5) = range_vector(2)/range;
H(2,6) = range_vector(3)/range;


end
```

Appendix D.

```
function [v2,warning] = gibbs(r1,r2,r3)

% Test case vectors
%r1 = [1684.709420; -6982.280710; -601.808660]
%r2 = [1599.998580; -7028.257130; 14.877410]
%r3 = [1502.433630; -7021.711680; 631.719850]

% Capt Brian L. Foster
% 23 December 2002

% This is Algorithm 48 from Vallado (1997) page 414.
% It returns the velocity vector associated with position
% vector r2.

% The input vectors r1, r2, and r3 are in the IJK coordinate system
% and with units of kilometers.

format long g

% Earth's gravitational parameter, km^3/s^2
mu = 398600.4415;
```

```
% Normal vectors

Z12_vec = cross(r1,r2);

Z23_vec = cross(r2,r3);

Z31_vec = cross(r3,r1);


% Vectors are coplanar if Z23_vec is perpendicular to r1.


% Magnitudes of the position vectors

r1_mag = norm(r1);

r2_mag = norm(r2);

r3_mag = norm(r3);


% Check to see how coplanar the vectors are.

alpha_cop = 90.0 - acos(dot(Z23_vec,r1)/...

   (norm(Z23_vec)*r1_mag))*180.0/pi


% Determine angular separations to ensure sufficient separation

% Angular separation between r1 and r2, in degrees

alpha12 = acos(dot(r1,r2)/(r1_mag*r2_mag))*180.0/pi


% Angular separation between r2 and r3, in degrees
```

```
alpha23 = acos(dot(r2,r3)/(r2_mag*r3_mag))*180.0/pi


if(alpha12 < 1.0 | alpha23 < 1.0)
   warning = 'r1, r2, and r3 are too close.  Use Herrick-Gibbs.'
   v2 = 'v2 not calculated.'
   return
end


% Intermediate vectors
N_vec = r1_mag * Z23_vec + r2_mag * Z31_vec + r3_mag * Z12_vec;
D_vec = Z12_vec + Z23_vec + Z31_vec;
S_vec = (r2_mag - r3_mag)*r1 + ...
   (r3_mag - r1_mag)*r2 + (r1_mag - r2_mag)*r3;
B_vec = cross(D_vec,r2);
Lg = sqrt(mu/(norm(N_vec) * norm(D_vec)));


% Velocity vector associated with r2, units in km/s
v2 = Lg/r2_mag * B_vec + Lg * S_vec
warning = 0;
return
```

Appendix E.

```
function [v2] = h_gibbs(r1,r2,r3,JD1,JD2,JD3)

% Capt Brian L. Foster

% 23 December 2002


% Test case vectors

r1 = [1607.879850;-7026.697450; -15.031650]

r2 = [1599.998580; -7028.257130; 14.877410]

r3 = [1592.705670; -7030.083050; 44.770310]


% Julian Dates of test case vectors

JD1 = 2452734.4999537

JD2 = 2452734.5

JD3 = 2452734.5000463


% This is Algorithm 49 from Vallado (1997) page 420.


format long g


% Earth's gravitational parameter, km^3/s^2

mu = 398600.4415;
```

```
% The position vectors r1, r2, and r3 are in the IJK

% coordinate system with units of kilometers.



% Remember that JD dates are in "DAYS" and must be

% converted to seconds.


del_t31 = (JD3 - JD1)*86400.0;

del_t32 = (JD3 - JD2)*86400.0;

del_t21 = (JD2 - JD1)*86400.0;


% Data for test case debugging.

% del_t31 = 153.04;

% del_t32 = 76.56;

% del_t21 = 76.48;


Z23_vec = cross(r2,r3);

Z23 = norm(Z23_vec);


r1_mag = norm(r1);

r2_mag = norm(r2);

r3_mag = norm(r3);
```

```
alpha_cop = 90.0 - acos(dot(Z23_vec,r1)/(Z23*r1_mag))*180.0/pi


% Determine angular separations to ensure sufficient separation

% Angular separation between r1 and r2

alpha12 = acos(dot(r1,r2)/(r1_mag*r2_mag))*180.0/pi


% Angular separation between r2 and r3

alpha23 = acos(dot(r2,r3)/(r2_mag*r3_mag))*180.0/pi


if(alpha12 > 5.0 | alpha23 > 5.0)

   v2 = 'v2 not calculated.';

   warning = 'r1, r2, and r3 are too far apart.  Use Gibbs method.';

   return

end


% Velocity vector associated with second position vector in km/s.

v2 = -del_t32*(1/(del_t21*del_t31) + mu/(12*r1_mag^3))*r1 +...

   (del_t32 - del_t21)*(1/(del_t21*del_t32) + mu/(12*r2_mag^3))*r2 +...

   del_t21*(1/(del_t32*del_t31) + mu/(12*r3_mag^3))*r3

warning = 0;

return
```

Bibliography

1. AN/FPS -85 Spacetrack Radar Physical Dimensions. http://www.globalsecurity.org/space/systems/an-fps-85.htm.

2. Bate, Roger R., Donald D. Mueller, Jerry E. White. *Fundamentals of Astrodynamics.* New York: Dover Publications, Inc., 1971.

3. Cantafio, Leopold J. *Space-based Radar Handbook*. Norwood, MA: Artech House, Inc., 1989.

4. "China Completes Ground Tests of Anti-satellite Weapon." *Sing Tao*, Hong Kong, January 5, 2001. Translated by FBIS. Document ID: CPP20010105000026.

5. Cole, Timothy D., Mark Boies, Ashruf El-Dinary. "Laser Radar Instrument for the Near-Earth Asteroid Rendezvous (NEAR) Mission," *Proceedings of SPIE*, Vol. 2748 (1996), pp. 122-139.

6. Cosyn, Philippe. "China Plans Rapid-Response, Mobile Rocket, Nanosatellite Next Year." SpaceDaily.com website. http://www.spacedaily.com/news/china-01zc.html.

7. Ground-Based Electro-Optical Deep Space Surveillance (GEODSS) data. http://www.globalsecurity.org/space/systems/geodss.htm.

8. Hasson, V., F. Corbett, M. Kovacs, M. Groden, D. Hogenboom, G. Dryden, R. Pohle, C. Phipps, D. Werling, S. Czyzak, J. Gonglewski, and J. Campbell. "Use of Laser Radar for Small Space Object Experiments," *Proceedings of SPIE*, Vol. 4091 (2000), pp. 363-374.

9. International Laser Ranging Service website. http://ranier.oact.hq.nasa.gov/Sensors_page/Laser/SLR.html

10. Keil, Robin, editor. *Missile Systems of the World*. Bremerton, WA: AMI International, 1999.

11. Kovacs, M., G. Dryden, R. Pohle, K. Ayers, R. Carreras, L. Crawford, and R. Taft. "HI-CLASS on AEOS: A Large Aperture Laser Radar for Space Surveillance/Situational Awareness Investigations," *Proceedings of SPIE*, Vol. 4490 (2001), pp. 298-306.

12. Montenbruck, Oliver and Eberhard Gill. *Satellite Orbits-Models, Methods, and Applications.* Heidelberg, Germany: Springer-Verlag, 2000.

13. Nicolas, Joëlle, Francis Pierron, Michel Kasser, Pierre Exertier, Pascal Bonnefond, François Barlier, and Jennifer Hasse. "French Transportable Laser Ranging Station:

Scientific Objectives, Technical Features, and Performance." *Applied Optics*, Vol. 39, No. 3, 20 January 2000, pp. 402-410.

14. Psiaki, Mark L., "Satellite Orbit Determination Using a Single-Channel Global Positioning System Receiver," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 1, 2002, pp. 137-144.

15. Schwartz, Jon A. "Pulse Spreading and Range Correction Analysis for Satellite Laser Ranging," *Applied Optics*, Vol. 29, No. 25, 1 September 1990, pp. 3597 – 3602.

16. Sietzen, Frank. Jr. "Microspace Technology Comes to China." SPACE.com website. October 19, 2000. http://www.space.com/news/spaceagencies/microsat_china_001019.html.

17. Streetly, Martin, editor. *Jane's Radar and Electronic Warfare Systems 2001-2002, 13th Ed.* Alexandria, VA: Jane's Information Group, Inc., 2001.

18. Sydney, Paul, John Africano, Amy Fredericks, Kris Hamada, Vicki SooHoo, Daron Nishimoto, Paul Kervin, Steve Bisque, and Matthew Bisque. "Raven Automated Small Telescope Systems," *Proceedings of SPIE*, Vol. 4091 (2000), pp. 237-247.

19. Tansey, R.J., B. Campbell, and A. Koumvakalis. "Description and Experimental Results of a 58-lb Portable LEO Satellite Tracker," *Proceedings of SPIE*, Vol. 3434 (1998), pp. 78-87.

20. Vallado, David A. *Fundamentals of Astrodynamics and Applications.* New York: McGraw-Hill, 1997.

21. Wei, Long. "China's First Microsat Operational." Report posted on SpaceDaily website July 11, 2000. http://www.spacedaily.com/news/microsat-00k.html.

22. Wertz, James R. and Wiley J. Larson. *Space Mission Analysis and Design, 3rd Ed.* Torrance, CA: Microcosm Press, 1999.

23. Wiesel, William E. Class handout distributed in MECH 731, Modern Methods of Orbit Determination. Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, June 1998.

24. Wilson, Tom. "Threats to United States Space Capabilities." Background paper prepared for the Commission to Assess United States National Security Space Management and Organization, January 11, 2001.

Vita

Captain Brian L. Foster graduated from Panola High School in Panola, Oklahoma in 1986. He was commissioned on 9 May 1992 through AFROTC Detachment 675 at the University of Oklahoma where he received a Bachelor of Science in Aerospace Engineering. Upon entering active duty in February 1993, he attended Undergraduate Missile Training at Vandenberg AFB, California where he graduated as a Top Performer. After UMT, he was assigned to the 741st Missile Squadron at Minot AFB, North Dakota as a Deputy Missile Combat Crew Commander. He also served as a Missile Combat Crew Commander instructor and Flight Commander. In July 1997 he was assigned to the 3rd Space Operations Squadron Schriever AFB, Colorado as an orbital analyst for the Defense Satellite Communication System III and Ultra-high Frequency Follow-On programs. While in 3 SOPS, he was a key member of three teams which conducted launch operations for DSCS III B-8 and UHF F/O Flights 9 and 10. In June 2000, he became a space operations instructor in Detachment 1, 533rd Training Squadron at Schriever AFB, Colorado. During his tour of duty in Colorado, he completed a Master of Engineering in Space Operations from the University of Colorado at Colorado Springs. In August 2001, he was one of five officers to enroll in the Graduate School of Engineering and Management, Air Force Institute of Technology, as the first ever class under Air Force Space Command's Vigilant Scholar Program. Upon graduation, he will be assigned to Headquarters Air Force Space Command, Directorate of Requirements, Navigation and Communication Division, where he will work technical analysis issues related to navigation warfare and modernization of the Global Positioning System.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)* 14-03-2003	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From – To)* Sep 2002 – Mar 2003

4. TITLE AND SUBTITLE ORBIT DETERMINATION FOR A MICROSATELLITE RENDEZVOUS WITH NON-COOPERATIVE TARGET	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Foster, Brian L., Captain, USAF	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology, Graduate School of Engineering and Management (AFIT/EN), 2950 Hobson Way, Building 640, WPAFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT/GAI/ENY/03-2
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This study investigated the minimum requirements to establish a satellite tracking system architecture for a hostile "parasitic microsatellite" to rendezvous with a larger, non-cooperative target satellite. Four types of tracking systems and their capabilities were reviewed with emphasis on "low-technology" level and/or mobile systems which could be used by technologically unsophisticated state or non-state adversaries. With the tracking system architecture selected, simulated tracking data was processed with a non-linear least squares orbit determination filter to determine and/or update the target satellite's state vector.

15. SUBJECT TERMS

Microsatellite, Orbit Determination, Non-linear Least Squares, Rendezvous

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Dr. Steven G. Tragesser
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(Include area code)*
U	U	U	UU	127	(937) 255-6565, ext 4286; e-mail: Steven.Tragesser@afit.edu

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18