GNSS Hardware-In-The-Loop Formation and Tracking Control

Frederick Bernard Harris Jr.

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
In
Electrical Engineering

Wayne A. Scales
Joseph Baker
Steven W. Ellingson

April 29, 2016
Blacksburg, VA

Keywords: GNSS, Hardware in the loop, formation flying, formation control, formation tracking, real-time, satellite, spacecraft, airplane, Ashtech, Spirent, Virginia Tech

GNSS Hardware-In-The-Loop Formation and Tracking Control

Frederick Bernard Harris Jr.

Abstract

Formation and tracking control are critical for of today's vehicle applications in and this will be true for future vehicle technologies as well. Although the general function of these controls is for data collection and military applications, formation and tracking control may be applied to automobiles, drones, submarines, and spacecraft. The primary application here is the investigation of formation keeping and tracking solutions for realistic, real-time, and multi-vehicle simulations. This research explores the creation of a predictive navigation and control algorithm for formation keeping and tracking, raw measurement data collection, and building a real-time GNSS closed HWIL testbed for simulations of different vehicles. The L1 frequency band of the Global Positioning System (GPS) constellation is used to observe and generate raw measurement data that encompasses range, pseudo-range, and Doppler frequency. The closed HWIL simulations are implemented using Spirent's Communication Global Navigation Satellite system (GNSS) 6560 and 8000 hardware simulators along with Ashtech, G-12 and DG-14, and Novetel OEM 628 receivers. The predictive navigation control is similar to other vision-based tracking techniques, but relies mainly on vector projections that are controlled by acceleration, velocity magnitude, and direction constraints to generate realistic motion.

The current state of the testbed is capable of handling one or more vehicle applications. The testbed can model simulations up to 24 hours. The vehicle performance during simulations can be customized for any required precision by setting a variety of

vehicle parameters. The testbed is built from basic principles and is easily upgradable for future expansions or upgrades.

GNSS Hardware-In-The-Loop Formation and Tracking Control

Frederick Bernard Harris Jr.

General Audience Abstract

This thesis work pertains to simulated one-way formation tracking and control using a Global Navigation Satellite System (GNSS) hardware and software. Virtual simulations are important for testing because they have the capability of modeling realistic vehicle behavior for both the target and follower vehicles. Performing these simulations can ensure that vehicle prototypes or designs can meet the performance requirements for travel. The primary application entailed here is the investigation of formation keeping and tracking solutions for realistic, real-time, and multi-vehicle simulations. This research explores the creation of a predictive navigation and control algorithm for formation keeping and tracking, raw measurement data collection, and building a real-time GNSS closed hardware-in-the-loop (HWIL) testbed for simulations of different vehicles using GPS and navigation principles. The closed HWIL simulations are implemented using Spirent's Communication GNSS 6560 and 8000 hardware simulators along with Ashtech, G-12 and DG-14, and Novetel OEM 628 receivers. Along with space applications, this research has applications outside the space environment in automated automobile travel using GPS that motor companies are investigating for the new generation of cars.

Acknowledgements

First and foremost, I acknowledge and honor God for everything. I thank Him for giving and allowing me to use the knowledge that I have to perform the research and produce functional results.

I would like to acknowledge my parents Frederick Bernard Harris Sr. and Leontyna Morniece Harris for always pushing my sister and myself to reach our maximum potential and loving us more than anything.

I would like to acknowledge Dr. Wayne A. Scales for his guidance and wisdom through my entire graduate career. Being under his tutelage has been a vital part of my graduate success and thesis research.

Finally, I would like to acknowledge Marc Jean for all of his help and assistance with this project. I am thankful to Marc Jean for sharing his coding expertise and knowledge of the hardware for the completion of the project. He is responsible for coding and integrating some of the research and theory provided in this paper.

Dedication

Dedicated to my parents who have always encouraged me to pursue my dreams and have guided me throughout my life.

Dedicated to my sister who has always made me want to be a better version of myself so I could continue to be a better role model for her.

Dedicated to Kiera Welch for her unwavering love and support during my graduate career and research.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

.N	Navigation File
CPU	Computer
DOP	Dilution of Precision
ECEF	Earth-Centered-Earth-Fixed
EKF	Extended Kalman Filter
EO-1	Earth Orbiter-1
HWIL	Hardware-in-the-loop
Hz	Hertz
GNSS	Global Navigation Satellite system
GPS	Global Positioning System
kg	Kilograms
km	Kilometers
LEO	Low Earth orbiting
LQG	Linear Quadratic Gaussian
m	Meters
m/s	Meters per second
m/s^2	Meters per second squared
m/s^3	Meters per second cubed
NEV	North, East, and Vertical
NMEA	National Marine Electronics Association
PRN	Pseudorandom Noise
PPS	Precise Positioning Services
PSD	Position Sensing Diode
RINEX	Receiver Independent Exchange Format
RF	Radio Frequency
RM	Raw Measurement
s	Second
SA	Selective Availability
SBAS	Space-based augmentations system
SPS	Standard Positioning Services
TEC	Total Electron Content
UTC	Coordinated Universal Time
WGS-84	World Geodetic Survey 1984

Chapter 1: Research Overview

Introduction

The applications of formation control and keeping are becoming an essential method for efficient data collection, military applications, and positioning services. Formation control refers to the simultaneous management and tracking of multiple autonomous vehicles that typically move relative to one another based off a shape, pattern, or separation distance [1]. Formation keeping is the ability to maintain the vehicle composition throughout the duration of travel. The primary application considered here is the development of a closed hardware-in-the-loop (HWIL) testbed to investigate formation-flying applications using multiple GNSS constellations. Such formations applied to spacecraft may be utilized for scientific mission objectives such as ionospheric Total Electron Content (TEC) acquisition, and also GNSS scintillation research [2]. This work describes the development of controlling a formation utilizing a HWIL testbed using L1 GPS signals for positioning. The goal of this research is to establish a functional closed HWIL testbed primarily for low Earth orbiting (LEO) satellites within GPS L1 band accuracy. This testbed can extend to applications of other vehicles. The purpose of this thesis is the development and validation of a closed hardware-in-the-loop (HWIL) testbed for multi-vehicle applications. This research began with receiving GPS receivers and a GNSS signal simulator without any legacy software for navigation and guidance. Being an original concept and one of the first closed HWIL testbeds built at Virginia Tech, the validation of the testbed is focused on the versatility for different vehicle applications and formation keeping accuracy. The experiments performed involve two vehicles traveling in a target/follower scenario. The follower vehicle is controlled to move relative to a target vehicle capable of independent motion. Developing this system to meet the L1 band positioning accuracy requires a navigation algorithm

to control the positioning of a vehicle to be within 30 meters of the formation of interest [2]. LEO satellites orbit the earth at approximately 8 km/s at altitudes up to 2000 km [3]. At these speeds, the accuracy and tolerance limits used to maintain formations are the most difficult obstacles to overcome. The inability of a vehicle to keep track of required positions during travel could cause errors in data collection or collision with another obstacle. This research has applications outside of the space environment in automated automobile travel using GPS that motor companies are investigating for the new generation of cars.

1.1 Formation control and tracking using L1 GPS signals

HWIL formation flying testbeds use receivers and radio frequency (RF) signals to generate current positions for vehicles that are used to control the motion. The GPS receivers using L1 GPS signals for positioning observes and locks into the PRN codes of GPS satellites. Once the GPS receiver is tracking the PRN codes and subsequently determines the ephemeris and ranging information of 4 or more GPS satellites, a navigation solution is produced. This navigation solution provides latitude, longitude, altitude and dilution of precision (DOP) of the navigation solution at accurate times [2]. The receiver can also be used to extract raw measurement information such as range, pseudo-range, and Doppler. The navigation algorithm outlined in this work stems from a line of sight-based formation flying method. This algorithm uses the navigation solution from the GPS receiver for current positions to control its movements. Positions, speeds, and accelerations of the follower are calculated based on the position and direction of the target (or chief) vehicle/path and the current location of the follower (or deputy) vehicle.

Using the HWIL setup allows for the elimination of receiver clock errors due to the time synchronization feature provided by the receivers. Time is synchronized when at least one GPS

PRN code is acquired. The main disadvantage of using the L1 frequency by itself is that the position error is on the meter-level in comparison to using both L1 and L2 frequencies' centimeter-level accuracy [2].

1.2 Requirement Overview

The goal of this research is to establish a universal closed HWIL testbed for multiple vehicles within GPS L1 band precision. The vehicles of interest may include spacecraft, satellites, aircraft, and automobiles. The testbed scenarios for each vehicle should display realistic vehicle behavior for both the target and follower vehicles. The table below shows the vehicles of interest.

LEO satellites orbit the earth at approximately 8 km/s at altitudes up to 2000 km. The orbits of these satellites are near circular (eccentricity $e \approx 0$). The average mass of recent LEO spacecraft is around 500 kg [4]. These spacecraft range from the International Space Station to Cubesats. Cubesats are growing in popularity among academia and professional research because they are cost effective, practical, and reliable and can be used in a formation [5]. These satellites provide another application for this research to be used as guidance and navigation for these satellites. A low power thruster on the satellites provides accelerations up to 0.001 m/s^2 [6]. For three thrusters, corresponding to X, Y, and Z dimensions, the maximum acceleration/deceleration of LEO satellites is approximately 0.0017 m/s^2. This acceleration value does not include the acceleration of gravity also applied to the receiver that creates a centripetal force on the body of the satellite. The minimum acceleration of the vehicle is assumed to be between 9.81 to 10 m/s^2

The requirements for modeling aircraft scenarios come from the fastest plane and jet specifications, which produce the highest dynamics. These aircraft include the General

Dynamics Aardvark, Lockheed's SR-71 Blackbird, Boeing's 747-8, the Tupolev TU 144, and others [7] - [8]. The average top speeds of the aircraft are around 500 m/s and average mass around 10^5 kg. The average accelerations and decelerations are $5m/s^2$ and $1.5m/s^2$.

Car requirements come from the fastest automobiles, which produce the highest dynamics. These cars include the SSC Ultimate Aero TT, the Bugatti Veyron 16.4 Super Sport Vitesse, the Porsche 911 Turbo S, and others. The average top speeds of the automobiles are around 100 m/s and average mass around 1500 kg. The average accelerations and decelerations are $25m/s^2$ and $5m/s^2$ respectively [9].

Table 1: Vehicle scenario requirements

Vehicle	Average Mass (Kg)	Average Top Speeds (m/s)	Average Acceleration (m/s^2)	Average Deceleration (m/s^2)
LEO Satellite (Excluding External Forces)	500	8000	0.0017	0.0017
Jets	91334.9	989.1	7.05	2.95
Commercial Planes	119799.1	350	2.8	0.2
Cars	1586.5	100	25	5

1.3 Relevant Past Research and Historical Perspective

Formation and keeping control extends to the 1990's during the investigation of autonomous flight. There have been several NASA missions and independent research to advance the performance and methods of formation control and keeping. Formation control has extended to different types of satellites, drones, planes, and automobiles. There have also been various filters to generate robust navigation solutions. This section provides a broad overview of related research in this area.

One of the earliest investigations of formation flying for spacecraft was performed by Joseph Guinn and Ronald Boain during NASA's New Millenium Earth Orbiter-1 (EO-1) mission in 1996 [10]. The goal of this mission was to provide the tracking and orbit determination of EO-1 to fly in formation with Landsat-7. By allowing the two satellites to fly in formation, NASA hoped to take co-registered images of specific locations on Earth. Design requirements for separation distances were between 230-675 km while using GPS for navigation. This work emulates the use of GPS during this mission, which is similar to the process described earlier. Increasing the antenna gain, reducing single blockage, and using a single omnidirectional antenna helped make this mission successful [9]. This group tested their real-time orbit determination methods under Standard Positioning Services (SPS), Precise Positioning Services (PPS) that included Kalman filtered solutions, and Selective Availability (SA). The results of this test showed that the SA navigation had the least accurate navigation solution of approximately 60 m while the Kalman filtered solution had the most accurate solution with better than 5 m of precision. This group validated that single frequency GPS has the possibility to predict movement and provide maneuvers to satellites along with an accurate navigation algorithm.

NASA, being one of the main pioneers in this field of research, has extended this research to many other missions such as the GRAIL missions and docking procedures on the International Space Station. In 2001, NASA developed a HWIL testbed for spacecraft formation flying applications to support their mission concepts [11]. This testbed's architecture consisted of several systems to support multiple programs. These systems consisted of a central simulation controller, an inter-satellite communication subsystem, an onboard processing subsystem, a ground control and telemetry subsystem, guidance and trajectory system, an environment subsystem, a vehicle control subsystem, a health and status subsystem, and a formation flying

executive [11]. The system's functions ranged from running the general hardware and software to controlling the systems of the spacecraft. These systems also modeled the ionosphere and communications between satellites and ground base stations. Their project uses a Linear Quadratic Gaussian (LQG) control algorithm to keep the spacecraft in formation in an integrated closed-loop analysis [11].

There have been many others who have provided different methods to achieve the formation control techniques. In 2004, Eric Johnson and his group implemented several vision-based methods for formation control with aircraft [12]. Information passed between the vehicles was 2-D vision, 3-D vision, or bearing information to maintain the formation. This group also used an Extended Kalman filter (EKF) to improve the performance of their motion control techniques. The group successfully performed their tests without knowing the lead vehicle's acceleration quantities during their simulations. This group's method was taken one-step further in this work by performing similar tests while only knowing the leader's position.

A group at Texas A&M University also investigated vision-based methods of formation flying using Position Sensing Diodes (PSD) [13]. This group used optical sensors to establish a relative navigation between the vehicles traveling in formation. A traditional Kalman filter was implemented to establish a robust navigation solution. The Texas A&M group validated their visual navigation sensor design with errors in position and velocity being near zero on average.

D. Lebedev and A. Tkachenko investigated magnetic attitude control for circular orbit LEO satellites without using a Kalman filtered solution. Lebedev attempted to maintain the motion of a LEO satellite based on three-axis magnetometers and magnetic actuators [14]. Damping devices were not used in this research, which led to difficulties in estimating the states of nonlinear dynamics [14]. The algorithm in this research uses the least squares method and

fuzzy ellipsoidal estimation. Lebedev's setup resulted in a nearly circular orbit with 10% error in inertial values. The results showed that vehicles could be controlled to follow specified paths. Although the EKF is not present, methods to control motion can still be used efficiently without it.

Recent work performed at Yonsei University in 2009 demonstrated HWIL simulations based on GPS-based navigation for satellite formation flying [15]. This group created their navigation solution from an EKF and nonlinear dynamics. A HWIL testbed was successfully built to perform real-time simulations of the navigation control of multiple satellites flying in formation. The testbed included a GPS signal simulator, GPS receivers, a GPS monitoring system, flight navigation station, and a remote control system. The GPS signal simulator's purpose is to receive commands from the remote control system and create RF signals for the receiver to generate a navigation solution. The GPS monitoring system extracts raw measurement data from the receiver and transfers the navigation solution to the flight control system. The flight control system generates new motion commands to be delivered on the next time-based motion update to the GPS simulator. The remote control section receives the new motion data and injects it into the software in real time. Once this process finishes, the cycle repeats. The experiment this group conducted was to have a leader-follower formation achieve a separation distance of 100 m for 141 s. The 3D RMS error for the position was 7.98 m on average and 0.0018m/s on average error in velocity.

The work in this thesis uses the concepts just described that were validated from previous works to perform a universal formation flying method with any vehicle in a HWIL testbed. Motion control in this work is based off fundamental principles similar to the line-of-sight 3-D

vision method. The filtering method relies on logic controls to help make the motion realistic. The goal is to provide accurate formation control between any two vehicles.

1.4 Thesis Overview

The organization of the subsequent sections of this thesis is as follows.

Chapter 2 discusses an overview of GPS concepts, GPS receiver functions, GPS signal simulators, coordinate systems, coordinate system conversions, and raw measurement calculations needed to build different systems of the testbed.

Chapter 3 introduces formation-flying concepts and analyzes the fundamentals used to generate the motion algorithm and the relative line-of-sight navigation theory between the lead and follower vehicles. This chapter also introduces logic controls used to filter the navigation algorithm.

Chapter 4 describes the HWIL tested and simulation setup. The chapter establishes a walkthrough of how each component of the testbed cooperates with one another and the processes of the testbed.

Chapter 5 outlines the results of experiments for different vehicles and scenarios. This section also explains errors and failed experiments that led to the final design.

Chapter 6 summarizes the work outlined in this document and provides suggestions for further improvements that can be implemented.

Chapter 2: GPS Theory and Technologies

2.1 Fundamental GPS Principles and Overview

Navigation and guidance for formation control in this work uses GPS for obtaining vehicle positions. GPS is the basis for calculating positions and trajectories for vehicles to use during simulations. GPS uses a series of satellites orbiting Earth to transmit electromagnetic waves for users to determine locations. GPS operates on two principles. The first principle is that radio frequency waves travel at certain speeds providing ranging information and the second principle is that user locations can be determined from the intersection of spheres with known center locations corresponding to the GPS satellite locations [2]. A radio signal in a vacuum travels at the speed of light c. If a radio signal is transmitted between two objects, the range $\boldsymbol{p}$ is the distance between the objects and can be found by simply using the following formula.

$$\boldsymbol{p} = \boldsymbol{c}(\boldsymbol{t_2} - \boldsymbol{t_1})$$

The equation shows $\boldsymbol{t_1}$ as the initial transmit time and $\boldsymbol{t_2}$ is shown as time the signal arrives at the receiver. In three dimensions this example can be related to a user's position and a transmitter station's current position. The user location would be some spherical radius $\boldsymbol{r}$ away from the transmitter station. Adding two more transmitters to the scenario would generate three independent distances to a user location. The intersection of these three spherical distances determines a user location. This technique is a simplified method that neglects offsets in the receiver and transmitter clocks from the GPS reference time. These offsets are accounted for by using a fourth transmitter to consider time as the fourth dimension.

GPS operations separate into the space segment, control segment, and the user segment. The space segment refers to the GPS satellites, the GPS satellites' precision clocks, GPS navigation message, and the ranging signals. There are currently 32 operational GPS satellites,

although only 24 are required (4 satellites in 6 different orbital planes). GPS satellites' orbits allow a user to view a minimum of 6 satellites from any location on Earth. Each satellite transmits three carrier signals called L1, L2, and L5. L1 refers to the signal frequency of 1.57542 GHz, L2 defines a GPS signal frequency of 1.2276 GHz, and L5 is a carrier frequency of 1.14675 GHz [2]. Each satellite has their carrier frequency modulated with pseudorandom noise (PRN) codes that allow each satellite to be uniquely identifiable. The control segment controls the GPS satellites motion and operations. The control segment also calculates the ephemeris data, which are parameters that can be used to locate GPS satellites. The user segment includes GPS receivers and the navigation solution. The navigation solution is created from ranging signals and ephemeris data contained in the navigation message.

The navigation solution provides a user's location that has high accuracy, can be rapidly updated, and can be used anywhere on earth due to the coverage of GPS satellites [2]. The navigation solution consists of three spatial coordinates and time that requires a minimum of four GPS satellite signals to be tracked by the receiver. The navigation solution can be used to find the range, pseudo-range, and Doppler frequency of each GPS satellite. The range $\boldsymbol{p}$ is the actual distance between the user and the satellite. The range is the distance between the GPS satellite transmitter and the user receiver as represented by the earlier example. The three-dimensional range can be found using the distance formula.

$$\boldsymbol{p} = \sqrt{(\boldsymbol{X_j} - \boldsymbol{X_i})^2 + (\boldsymbol{Y_j} - \boldsymbol{Y_i})^2 + (\boldsymbol{Z_j} - \boldsymbol{Z_i})^2}$$

The subscript *j* refers to a specific GPS satellite and the subscript *i* refers to a specific user. If the locations of the user and satellite are known, the distance can be found and used as the range.

The pseudo-range $\boldsymbol{P}$ is the measured distance between the GPS satellite and user receiver as determined by adding range with range equivalent clock errors of the GPS satellite and receiver. The following equation shows the calculation for pseudo-range.

$$\boldsymbol{P = p + c(\delta^j - \delta_R)}$$

$\boldsymbol{\delta^j}$ is the clock error of the GPS satellite and $\boldsymbol{\delta_R}$ is the clock error of the receiver. Clock errors primarily originate from drifts that may occur. These errors throw off the accuracy in determining the true distance. Satellite clock errors are calculated from the following equation.

$$\boldsymbol{\delta^j = af_0 + af_1(t - t_c) + af_2(t - t_c)^2 - T_{GD}}$$

The quantities in the satellite clock error are all provided in the navigation message where the $\boldsymbol{af}$ terms refer to satellite clock correction terms, $\boldsymbol{t}$ is the current GPS time, $\boldsymbol{t_c}$ is the time of epoch, and $\boldsymbol{T_{GD}}$ is the timing group delay. The pseudo-range is a way to analyze and compute how these errors affect the distance.

Doppler frequency provides a measurement of the relative motion between the transmitter and receiver [2]. Although the Doppler traditionally comes from carrier phase, the Doppler can also be calculated from the pseudo-range rate, $\boldsymbol{\dot{P}}$, divided by the wavelength. Dividing the relative velocity between the receiver and GPS satellite by the wavelength is another way of obtaining Doppler frequency. The equation below represents the equation for Doppler frequency where the subscript *R* is the user receiver, the subscript *T* is the satellite transmitter, $\boldsymbol{v}$ is the velocity, $\boldsymbol{x}$ is the current position, and $\boldsymbol{\lambda}$ is the wavelength of the transmit signal. Notational arrows having a single stem represent unit vectors and arrows having both stems represent vectors.

$$\boldsymbol{Doppler = \frac{\dot{P}}{\lambda} = \frac{-(\overrightarrow{v_T} - \overrightarrow{v_R}) \cdot \vec{x}}{\lambda} = \frac{-(\overrightarrow{v_T} - \overrightarrow{v_R}) \cdot \frac{(\overrightarrow{x_T} - \overrightarrow{x_R})}{|\overrightarrow{x_T} - \overrightarrow{x_R}|}}{\lambda}}$$

The velocities of the transmitter and receiver come from a derivative of position with time.

2.2 GPS Receiver Basics

GPS receivers in this work are used to obtain the locations of vehicles in a formation during simulations. Obtaining vehicle locations at current times allows kinematics to be applied on a vehicle to produce movement in the desired direction. GPS receivers obtain the observables (pseudo-range, Doppler frequency, and integrated carrier phase) from the ranging signals to generate navigation solutions. Since the signals from GPS satellites are modulated with a PRN code, steps are taken to acquire and track these signals and create a navigation solution. GPS receivers have a RF section, correlator section, and controller section [2]. The RF section receives the transmitted GPS signal, converts it to near baseband, and digitizes it. The correlator contains replicas of all the PRN codes and strips the converted signal of its PRN code and carrier to produce the encoded data contained in the signal [2]. GPS satellite locking occurs in the correlator and tracking begins once one GPS satellite is acquired. Lastly, the controller controls the other sections of the receiver and sends data to devices to allow the receiver to do certain user tasks such as creating the navigation solution.

Once the receivers are turned on, their primary goal is to collect satellite signals. Receivers need information on the satellites' locations to be known to obtain these signals efficiently. This process happens by storing ephemeris data into a GPS almanac that is a set of stored or calculated parameters that predict the location of GPS satellites. If no almanac, no initial approximate position, or approximate time is uploaded to the receiver before it begins looking for satellites, the receiver initializes a cold start [2]. Cold start refers to the receiver looking for all PRN codes and Doppler shifts until it has enough satellites to produce a navigation solution. This process takes a long time depending on the receiver's performance.

Warm starting a receiver occurs when accurate almanac data is uploaded to the receiver to allow it to produce a navigation solution much quicker through Doppler frequency determination. Initializing the receiver becomes crucial when running HWIL simulations and allowing the process to start on time.

2.3 Coordinate Systems and Transformations

There are various ways to describe the locations of vehicles. A vehicle can be at a location relative to Earth or to another vehicle. Although the locations would be the same, they would be represented with different values and units based on the reference or technique used to obtain the location. GPS receivers use various methods to define user and satellite locations. Understanding how GPS receivers specify locations is essential for applying dynamics or measurements. All GPS receivers essentially operate the same but can be designed to handle data differently. Some receivers produce navigation solutions in World Geodetic Survey 1984 (WGS-84) coordinates, some in different coordinate systems including Earth Centered Earth Fixed (ECEF), and others using a mixture of multiple coordinate systems. It is important to understand how to convert accurately between the coordinate systems. The coordinate systems discussed in this section are WGS-84, ECEF, and body frame coordinate systems.

WGS-84 and ECEF coordinate systems both use Earth's center as the origin of their respective coordinate systems. WGS-84 is an ellipsoidal coordinate frame and ECEF is an orthogonal Cartesian coordinate frame. The ECEF coordinate system uses traditional X, Y, and Z coordinates in a Cartesian coordinate system for describing user and satellite positions. WGS-84 uses latitude, longitude, and altitude for locating user positions. WGS-84 is used mainly for global navigation due to the non-spherical shape of the Earth. Figure 1 shows the relationship between WGS-84 and ECEF.

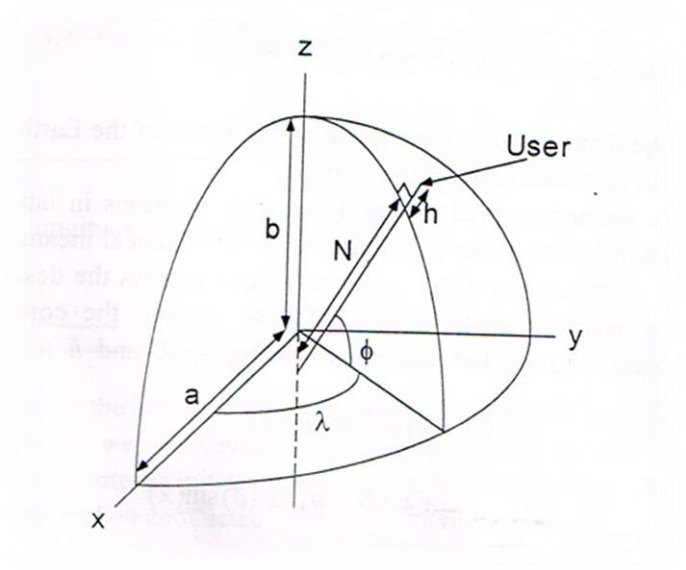

Figure 1: WGS-84 Coordinate Frame (λ, ϕ, and h) & ECEF Coordinate Frame (X, Y, and Z) [2]

Figure 1 shows ***a*** as the semi-major axis and ***b*** as the semi-minor axis of the Earth. ***N*** is the radius of curvature to the surface of Earth. ***h*** is the altitude. The altitude is the distance from ***N*** to the user. Equations to convert between the two coordinate systems are shown below. Latitude and longitude use units of radians. Note conversions from ECEF to WGS-84 must be done with a standard iterative procedure since the transformation equations are nonlinear.

$$\boldsymbol{x = (N + h)\, cos(\phi)\, cos(\lambda)}$$

$$\boldsymbol{y = (N + h)\, cos(\phi)\, sin(\lambda)}$$

$$\boldsymbol{z = \left(\frac{b^2}{a^2} N + h\right) sin(\phi) = (N + h)\left(1 - \frac{e^2 N}{N + h}\right) sin(\phi)}$$

$$\boldsymbol{N = \frac{a^2}{\sqrt{a^2 cos^2(\phi) + b^2 sin^2(\phi)}}}$$

$$\boldsymbol{h = \frac{\sqrt{x^2 + y^2}}{cos(\phi)} - N}$$

$$\boldsymbol{tan(\phi) = \frac{z}{p}\left(1 - \frac{e^2 N}{N + h}\right)^{-1}}$$

$$\boldsymbol{tan(\lambda) = \frac{y}{x}}$$

The value $\boldsymbol{e}$ is the eccentricity of earth that can be represented by the following equation.

$$\boldsymbol{e^2 = \sqrt{1 - \frac{b^2}{a^2}}}$$

North, East, and Vertical (NEV) unit vectors represent the body frame coordinate system. This coordinate frame is mainly used to map positions, velocities, or accelerations in one coordinate frame to ground track positions, velocities or accelerations. The ground track represents where or how the vehicle would be moving on earth's surface. *Figure 2* displays a diagram showing the relationship between ECEF, WGS-84, and the body frame coordinates.

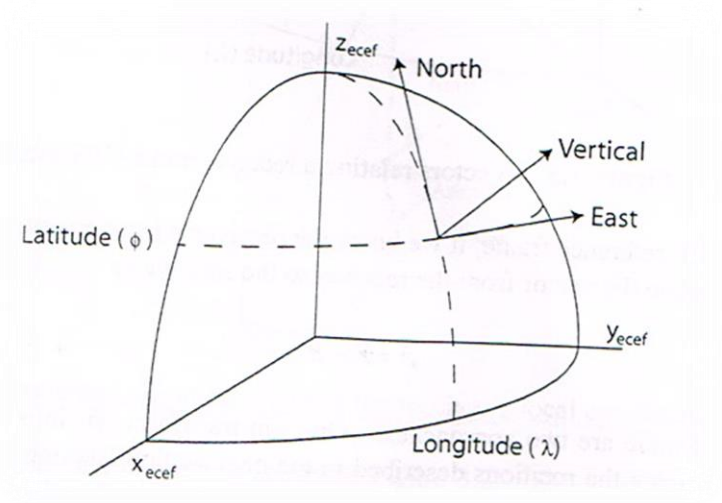

Figure 2: NEV Unit Vectors, WGS-84, & ECEF coordinate systems [2]

The NEV unit vectors can be mapped to WGS-84 using the following vector representations.

$$\boldsymbol{\vec{N} = [-cos(\lambda)\, sin(\phi)\,, -sin(\lambda)\, sin(\phi)\,, cos(\phi)]}$$

$$\vec{E} = [-\sin(\lambda), \cos(\lambda), 0]$$

$$\vec{V} = [\cos(\lambda)\cos(\phi), \sin(\lambda)\cos(\phi), \sin(\phi)]$$

The states in ECEF can be found from the NEV frame using a rotation matrix defined below.

$$[x_{P,V,A} \quad y_{P,V,A} \quad z_{P,V,A}] = \begin{bmatrix} -\cos(\lambda)\sin(\phi) & -\sin(\lambda)\sin(\phi) & \cos(\phi) \\ -\sin(\lambda) & \cos(\lambda) & 0 \\ \cos(\lambda)\cos(\phi) & \sin(\lambda)\cos(\phi) & \sin(\phi) \end{bmatrix} [N_{P,V,A} \quad E_{P,V,A} \quad V_{P,V,A}]$$

This matrix demonstrates that acceleration, velocity, or position can be easily converted to ECEF or body frame using the NEV unit vectors as a rotation matrix.

The last coordinate system referenced in this research is a Cartesian coordinate system with a vehicle body as the origin. Dynamics are applied to a vehicle of interest directly by using the vehicle as the origin. Vectors pointing from the vehicle to the direction of interest can allow one to apply velocities or accelerations in that direction. This system creates an awareness of how the environment or foreign bodies are moving relative to the object. This coordinate system aids with maintaining multiple vehicles moving relative to one another and can be established for multiple vehicles simultaneously.

The coordinate systems described above should be adequate to understand the data presented in the navigation solution produced by a GPS receiver. There is a need for a discussion pertaining to an issue between the fundamental GPS operation principles and the coordinate frames. During motion, errors in position are produced when signals are transmitted and received. Both the GPS satellites and the earth are moving simultaneously. If the calculation of the range and pseudo-range do not incorporate the propagation delay time of the RF signal the error in these distance quantities will be significant. The user and satellite clock offsets can cause range errors up to 1000's of kilometers [2]. The following images show the movement of the satellite between transmission t_T and reception t_R times in ECEF.

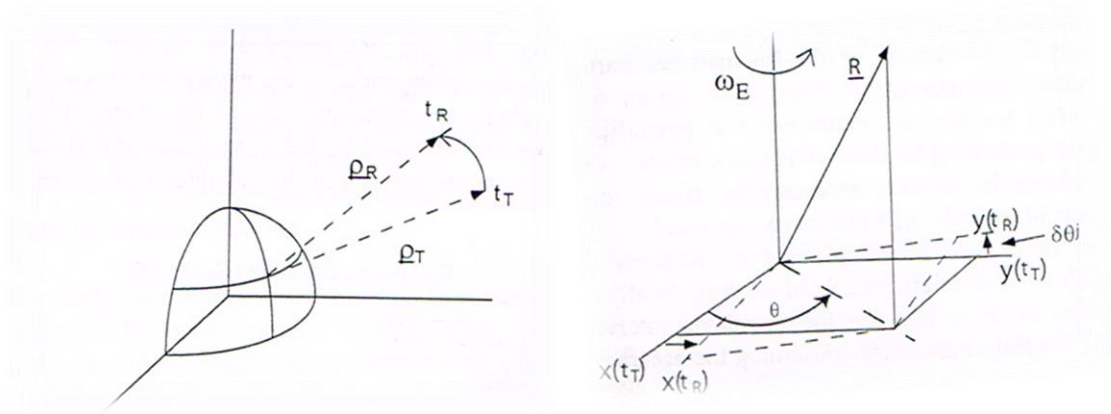

Figure 3 Demonstration of moving satellite in ECEF. Image of inertially fixed satellite on the left and movement of GPS satellite on the right [2].

The figure on the left shows the actual ranges of the satellite at times of transmit and reception for Earth not moving. From these values, a propagation time can be calculated.

$$\delta t_p^j = \frac{p^j}{c} = t_R - t_T$$

The image on the right shows both the earth and satellite moving. These motions require another correction that involves the angle of rotation between the two bodies.

$$\delta\theta^j = \omega_E \delta t_p^j$$

The angle of rotation, θ, is a time correction term involving the angular rotation of Earth and the propagation time term. Since the earth rotates around the *Z* ECEF axis, the *Z* term is not affected by this rotation. The corrected *X* and *Y* coordinates due to Earth's rotation can be found from the following equations.

$$X_c{}^j = X^j + Y^j \delta\theta^j$$

$$Y_c{}^j = Y^j - X^j \delta\theta^j$$

2.4 GNSS Hardware and Software Signal Simulator Overview

This work uses a GNSS hardware signal simulator and its software to define vehicles and their motion trajectories virtually to produce RF signals for a GPS receiver to process. The GNSS hardware signal simulators generate RF signals to emulate signals from GNSS satellites in the receiver's line-of-sight. Such simulators support GNSSs including GLONASS (Russia), Galileo (European Union), BeiDou (China), and GPS (United States). Some GNSS systems use software that allows for the creation, monitoring, and control of the receiver virtually. The figures below show the Spirent Communications GNSS 8000 system and software used in this work. It provides a simulated RF signal for GPS L1, L2, and L5 frequency bands.

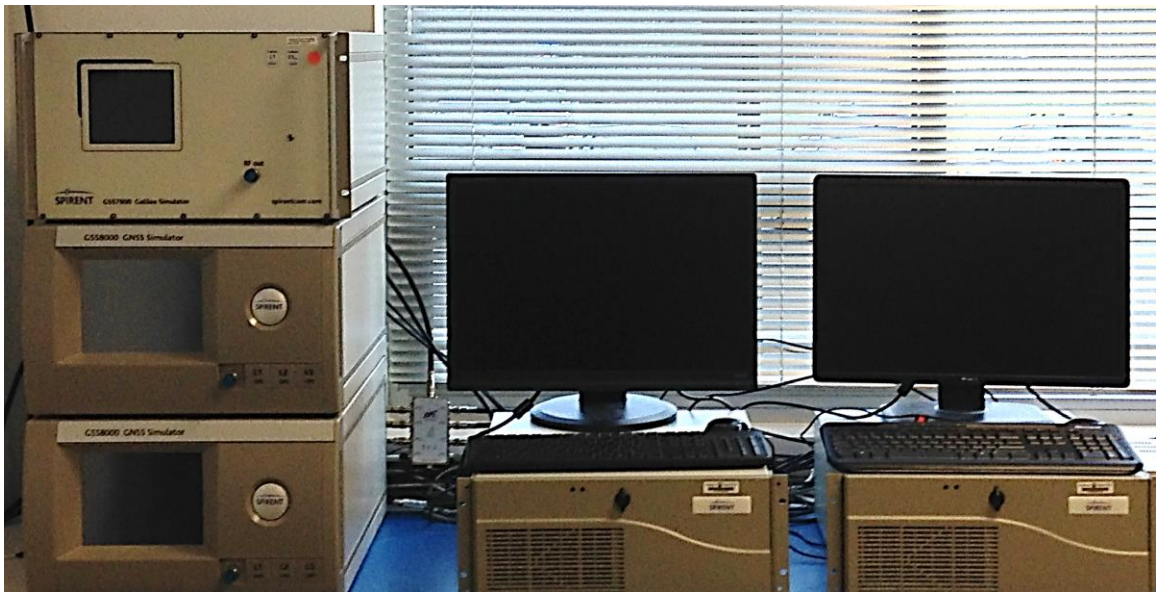

Figure 4: Spirent GNSS 8000 hardware RF signal simulator used in the study

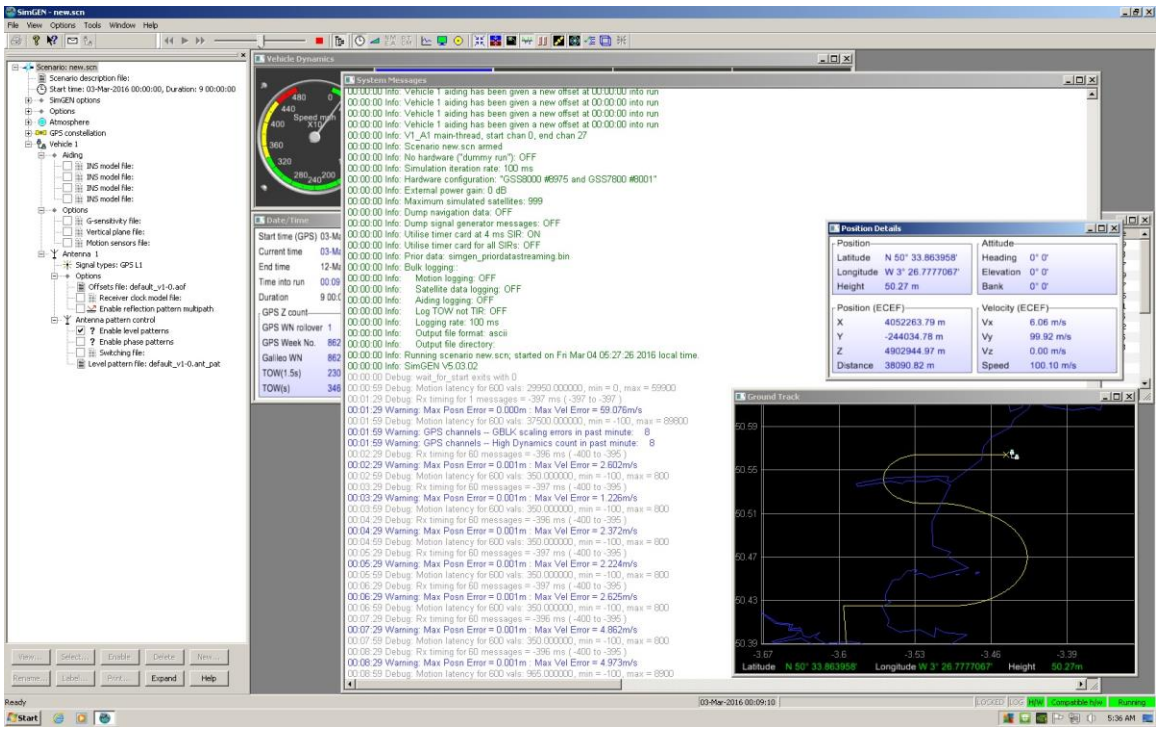

Figure 5:"SimGen" Spirent GNSS hardware RF signal simulator software interface

SimGen, also known as PosApp, is compatible software used with the Spirent GNSS 8000 hardware RF signal simulator. This software contains multiple features for creating and observing scenarios on Earth and in the space environment. This software supports the use of multiple GNSS constellations and vehicles. This software's features allow for the selection of a remote control vehicle for user inputs, adjustments of antenna parameters, monitoring of GPS observables of receivers, data logging, monitoring the trajectory performance, alternating the iteration rate of the software, and much more. All these features are essential to generate and control the vehicles' motion during travel.

Chapter 3: Theory of Navigation with Formation and Navigation Algorithm Principles

3.1 Relative Navigation of Leader-Follower Formation Basics

A leader-follower formation involves two or more vehicles where one lead vehicle is the reference trajectory for the following vehicles. In some texts, the leader-follower formations are referred to as chief-deputy or target-chaser formations. The formations are ideally the same but can be used differently based on the application. This work uses the chief-deputy phrase from the perspective that a vehicle may have a significant separation distance away from the chief and will have to move into formation. This scenario implies that the tracking strategy involves using a target-chasing method and then a leader-follower method. The chief-deputy phrase suggests that both perspectives can be applied at any time.

A prescribed geometry separates the vehicles traveling in formation. The geometry can be of any shape or size. *Figure 6* shows a basic chief-geometry in 2-D space. This work uses the Cartesian vehicle body system for discussing relative navigation in this section.

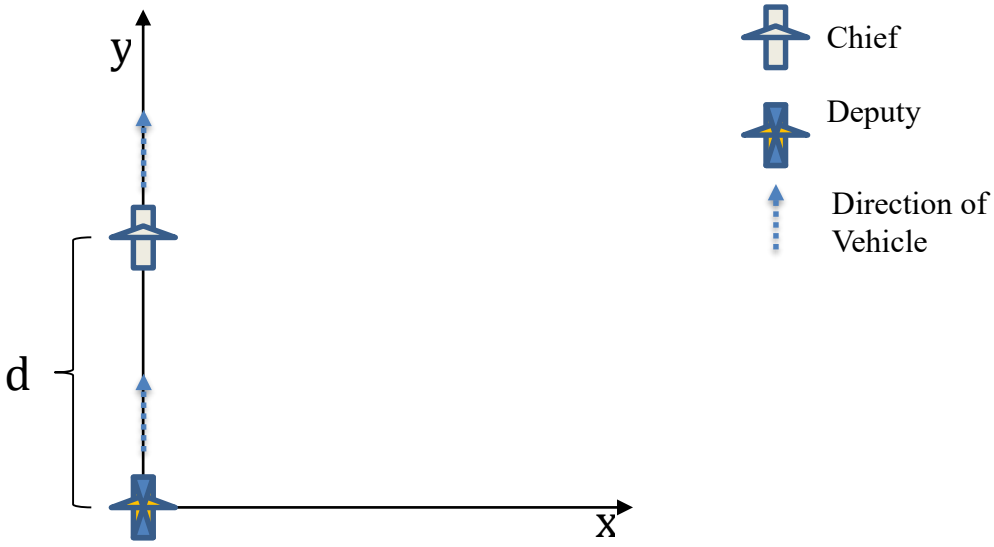

Figure 6: Basic 2-D example of vehicles in formation

In this example, the chief and deputy vehicles separated by some distance d travel in the same direction. The vehicles do not need to follow one behind the other necessarily. *Figure 7* displays the possible vehicle formations.

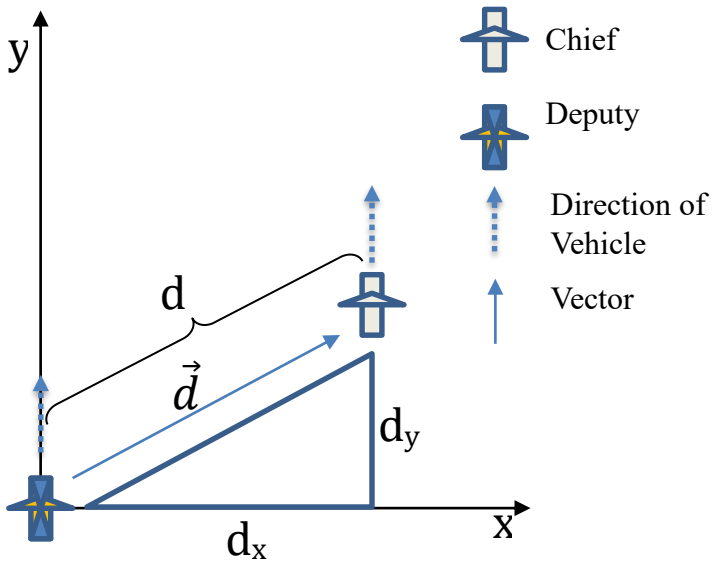

Figure 7: 2-D example of formation possibilities

Establishing a vector that has a magnitude and direction allows the deputy to apply dynamics for the guidance of the lead vehicle. Even with a separation distance in a specified direction, the deputy vehicle still continues to follow the trajectory of the chief. The figure shows that any velocity vector between the chief and deputy vehicle has a geometry that allows the vector magnitude $\boldsymbol{d}$ and directional components of the chief relative to the deputy to be obtained. The equations below represent obtainable quantities of relative position $\vec{\boldsymbol{d}}$ in 3-D space.

$$\vec{\boldsymbol{d}} = [\boldsymbol{d_x} \quad \boldsymbol{d_y} \quad \boldsymbol{d_z}] = [(\boldsymbol{x_{chief}} - \boldsymbol{x_{deputy}}) \quad (\boldsymbol{y_{chief}} - \boldsymbol{y_{deputy}}) \quad (\boldsymbol{z_{chief}} - \boldsymbol{z_{deputy}})]$$

$$|\vec{\boldsymbol{d}}| = \boldsymbol{d} = \sqrt{\boldsymbol{d_x}^2 + \boldsymbol{d_y}^2 + \boldsymbol{d_z}^2}$$

There can be any arrangement of deputy vehicles to a single chief vehicle to produce a shape or separation distances from other deputies.

3.2 Equations of Motion

A formation-flying scenario may have the vehicles at a separation distance away from their intended target path. Motion needs to be assigned to the deputy to meet desired positions at particular moments in time to either fall into or maintain the formation. Prescribed position, velocity, and acceleration allow the vehicle to move from one point to another. Each kinematic quantity has three dimensions to control the motion in any direction. The equations of position, velocity, and acceleration are shown below where r is the current position, $\boldsymbol{r_o}$ is the initial position, v is velocity, $\boldsymbol{v_o}$ is the initial velocity, $\boldsymbol{a}$ is acceleration, and $\boldsymbol{t}$ is a particular time or time interval.

$$\vec{\boldsymbol{r}} = [\boldsymbol{r_x} \quad \boldsymbol{r_y} \quad \boldsymbol{r_z}] = \vec{\boldsymbol{r}}_{\boldsymbol{o}} + \vec{\boldsymbol{v}}_{\boldsymbol{o}}\boldsymbol{t} + \frac{\boldsymbol{1}}{\boldsymbol{2}}\vec{\boldsymbol{a}}\boldsymbol{t^2}$$

$$\vec{\boldsymbol{v}} = [\boldsymbol{v_x} \quad \boldsymbol{v_y} \quad \boldsymbol{v_z}] = \frac{\boldsymbol{d}\vec{\boldsymbol{r}}}{\boldsymbol{dt}} = \vec{\boldsymbol{v}}_{\boldsymbol{o}} + \vec{\boldsymbol{a}}\boldsymbol{t}$$

$$\vec{\boldsymbol{a}} = [\boldsymbol{a_x} \quad \boldsymbol{a_y} \quad \boldsymbol{a_z}] = \frac{\boldsymbol{d}\vec{\boldsymbol{v}}}{\boldsymbol{dt}} = \frac{\boldsymbol{d^2}\vec{\boldsymbol{r}}}{\boldsymbol{dt^2}}$$

These equations assume the motion is linear, and acceleration is constant during the time interval. Position, velocity, and acceleration are derivatives of each other that correspond to the difference of a specific quantity at two instances of time divided by the total change in time. Linear motion defines the dynamics of an object moving between two positions in a straight line. This type of motion has limitations when it is used to approximate curvilinear or nonlinear motion. If the time interval, $\boldsymbol{dt}$, for linear motion is sufficiently small, linear motion can be used to approximate curvilinear or nonlinear motion.

These three quantities are essential for reaching a formation or keeping a vehicle in a formation. *Figure 8* demonstrates the use of the kinematic quantities for vehicles in a formation.

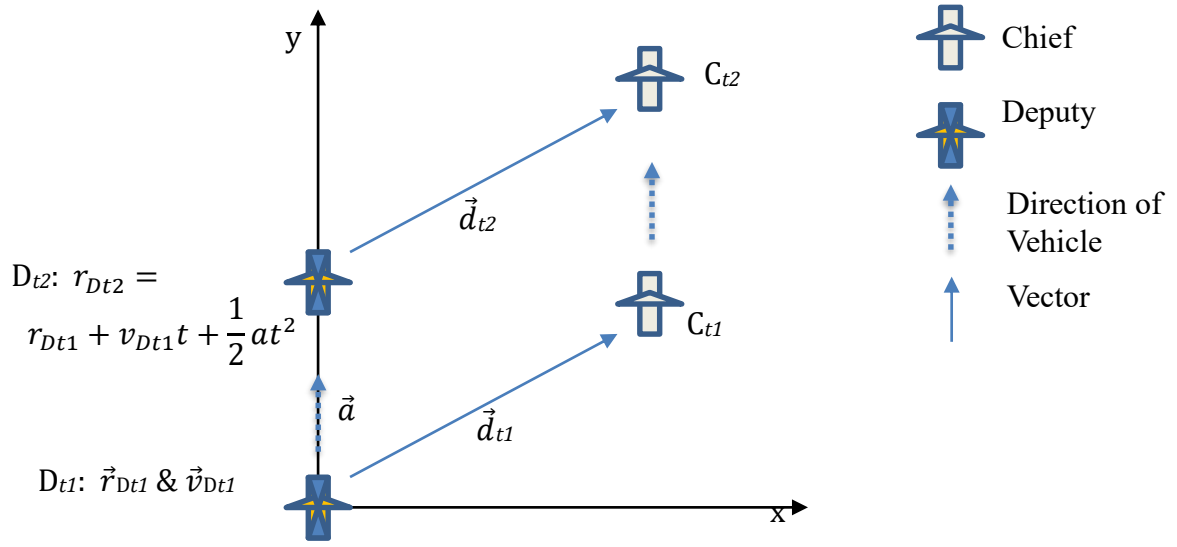

Figure 8: Kinematic demonstration of vehicles in formation

The figure shows a deputy and chief vehicle traveling in a formation. $\vec{V}_{t2}$ needs to be equal to $\vec{V}_{t1}$ to maintain the geometry between the vehicles. If the chief moves from its position at time t_1 to a new position at time t_2, the deputy needs to apply the proper motion to maintain the formation. If the location of the deputy at t_1 and chief positions at t_1 and t_2 are known at the least, a velocity and acceleration can be applied to allow the deputy to reach the necessary position for keeping the formation intact. Since the vector from the chief position is known for all time, the required deputy position is recognized for all time. If the vehicles were out of formation, the technique for applying motion would still be the same. Instead of motion being enforced in the same direction as the chief, the motion would be applied in the direction so the deputy would come into formation with the chief. Formation tracking uses this approach to allow the vehicles of interest to be arranged in a unique geometry.

For the deputy motion to be fluid during travel, multiple dynamic quantities need to be considered. Applying velocity by itself causes the motion to be a series of straight lines. Using velocity and acceleration produces a curvilinear trajectory. *Figure 9* shows the application of velocity and acceleration to positions.

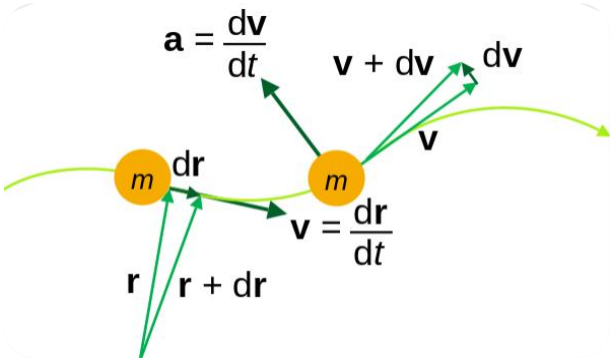

Figure 9: Motion of acceleration and velocity applied to position [16]

This figure shows how the kinematic quantities interact with one another during movement. A trajectory of vectors would be produced if velocity were solely used to update the motion because velocity is the change in position with respect to time. Each time interval provides a vector between two points and does not allow for smooth curvilinear motion. This behavior is not realistic. The trajectory can be approximated as a curvilinear path if the time interval between each motion is small. The curvilinear motion does occur when velocity and acceleration are applied which is demonstrated by the figure.

3.3 Navigation Algorithm

The navigation algorithm's purpose is to create and control motion for formation tracking and keeping. The algorithm receives position and velocity of a deputy from a receiver and then selects the best kinematic quantities to follow a target trajectory. A target trajectory is a path for

a deputy vehicle to emulate to achieve a formation. Once the kinematics is selected, the algorithm formats the information as a command and uploads it to the GPS simulator.

3.3.1 The Navigation Algorithm Process

The motion command to move the deputy vehicle is created after a series of steps. First, the target trajectory needs to be outlined. The target trajectory can be made to reflect the complete path or stream the path step by step. The algorithm can compute the deputy motion with the previous deputy kinematics, the current chief position, and the chief position at a future time. *Figure 10* demonstrates the algorithm's initial steps.

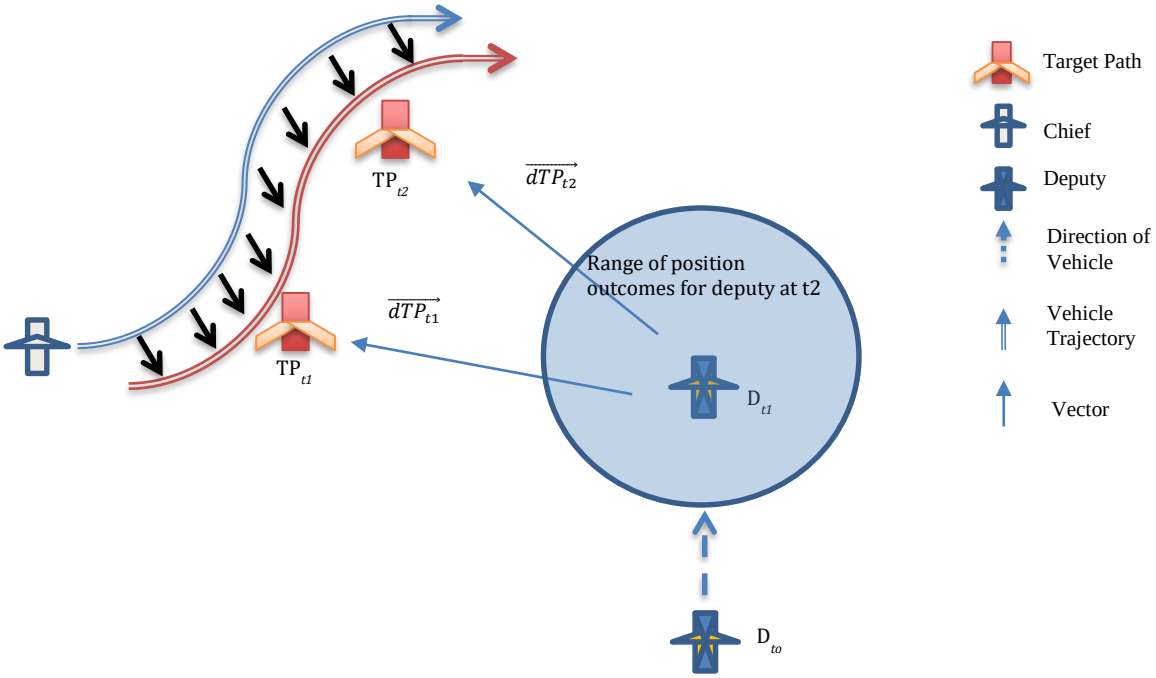

Figure 10: Algorithm initial steps for motion generation

The chief's trajectory produces a trajectory path for the deputy vehicle to follow because the geometry of the required formation is known. The initial deputy kinematics at time t_o generate certain kinematics for the current deputy position at time $\boldsymbol{t_1}$. For instance, if the position, velocity, and acceleration are known for a particular time step $\boldsymbol{t_o}$, the position and velocity can be

calculated for the next time step. If the position and velocity are known for a given time, the position can be approximated for the next time. When the deputy vehicle reaches its position at the next time t_1, it has at least position, but the missing quantities of either velocity or acceleration need to be estimated to produce motion for the deputy to reach the next required position at time t_2. The magnitude of $\overrightarrow{dTP_{t1}}$ is used to determine the performance of how well the deputy follows the formation. The performance of the formation keeping is essential to make sure the deputy has not overshot particular target path waypoints. The magnitude of $\overrightarrow{dTP_{t2}}$ is used to determine the distance between the deputy position at t_1 and the position the vehicle should achieve on the target path at t_2. If the magnitude of $\overrightarrow{dTP_{t2}}$ cannot be reduced in an applied time step then the deputy at t_2 falls somewhere along $\overrightarrow{dTP_{t2}}$ within the range of possible position outcomes determined by the vehicles' restraints.

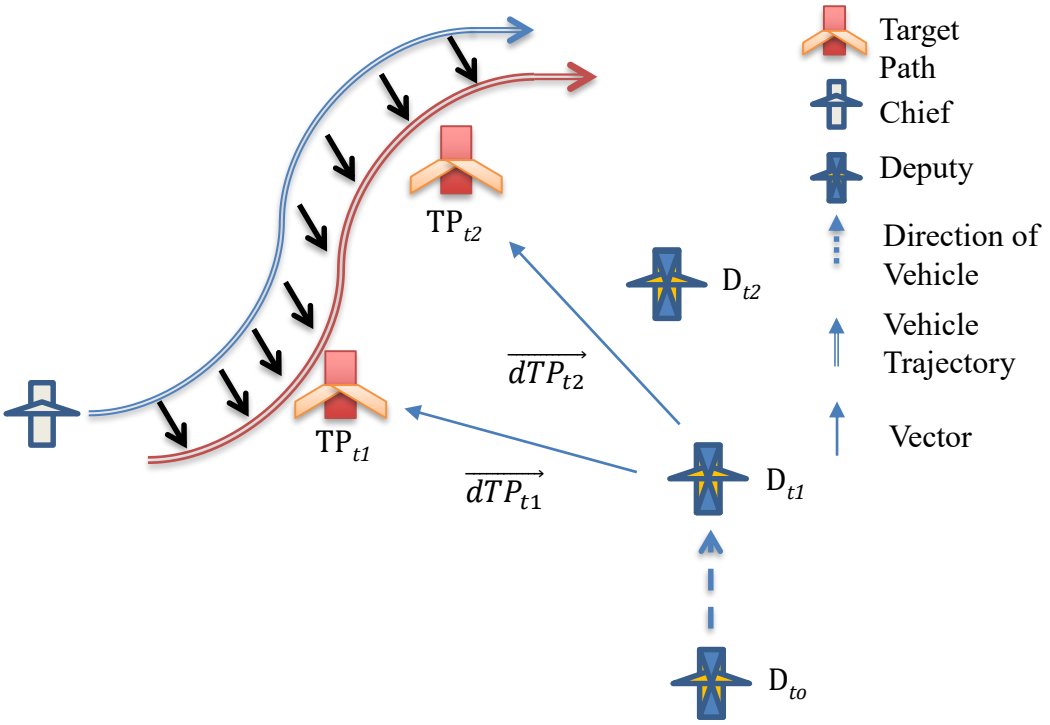

Figure 11: Final steps of navigation algorithm

Once the missing kinematic quantities are selected for the deputy's motion at t_1, a position represented at t_2 is established. *Figure 11* shows the final position at t_1 and the projected position at t_2 from determining the kinematics at t_1 to propel the deputy to fall into a formation. Logic controls are used to restrict the full capabilities the algorithm allows the deputy to achieve. The logic controls stem from vehicle limits and requirements of the motion scenarios.

3.3.2 Missing Kinematic Quantities of Current Time and Logic Controls

The missing kinematic quantities determine the position and possible velocity for the next time step. The missing quantity for the algorithm in this work is acceleration. The receiver provides position and velocity at each step. Determining acceleration produces a position and velocity for the algorithm to use in the next cycle. The algorithm generates a range of all possible combinations the three-dimensional acceleration values can be for any given time. This range is limited by the maximum and minimum acceleration values. The range of accelerations generates possible future positions and velocities for the next time step.

Logic controls are used to narrow down the acceleration value missing from the current time step by filtering the spectrum of current accelerations, the range of future positions, and future range of velocities. The position, velocity, and acceleration ranges can be filtered from selected rates of change between time steps for each respective quantity. Allowing a vehicle to speed up faster than it can achieve in the real world is unrealistic unless this is done for testing purposes. These derivative controls regulate the rates of change between time steps for each quantity.

Another suitable logic control is limiting the acceleration and velocity with some maximum value. For example, stopping the velocity with a maximum value restricts the vehicle from speeding up to a value that is out of the motion requirement. This control technique is made

stronger by zoning maximum limits based on $\overrightarrow{\boldsymbol{dTP_{t1}}}$. For example, the deputy is allowed to travel up to 100 m/s; the chief is allowed to travel up to 50 m/s; and the deputy vehicle is within 100 m of the chief. The deputy should begin to decelerate or slow its velocity depending on the separation. The maximum velocity can be set differently at select separation distances to accommodate for this feature.

Another method to zoning control is zoning by projections. If a chief moving some speed from a location is projected straight ahead over time and a deputy moving at another speed is projected in the direction of the expected chief location, the intersection of those vectors can determine if a deputy is moving too fast. *Figure 12* shows an example of this braking control.

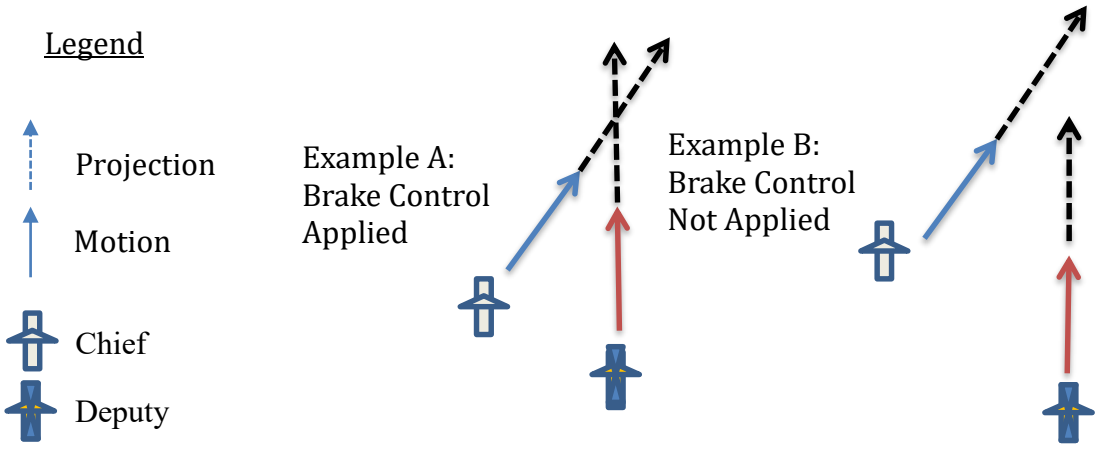

Figure 12: Example of zoning by projections

The control is applied when the projections of the vehicles intersect and is not applied when the projections do not intersect. Scaling unit vectors of the current magnitude in a certain direction creates projections.

The last control method applicable to the algorithm is the allowed angle of turn between the current position and the position at the next time step. Linking vectors head to tail of the deputy locations from to to t1 and from t1 to t2 produces the angle of turn. The dot product is

used to find the angle between the vectors. This angle is the angle of turn. Setting restrictions on this angle causes the accelerations only to be applied in certain directions.

After the algorithm's controls and calculations reduce the range of possible acceleration values, the acceleration that provides the closest distance to the target path is chosen. The controls put reasonable restrictions on a deputy vehicle to demonstrate real-world motion and maintain the vehicle from going out of control.

Chapter 4: Formation Tracking Testbed Configuration

4.1 Full System Setup

The full system of the closed Hardware-in-the-Loop-Formation-Testbed (HWILFTB) in this work is comprised of hardware and software discussed in previous chapters. This testbed is currently used for experimentation and simulating any two vehicle formation but can be expanded to more vehicles. *Figure 13* shows a diagram of the full system.

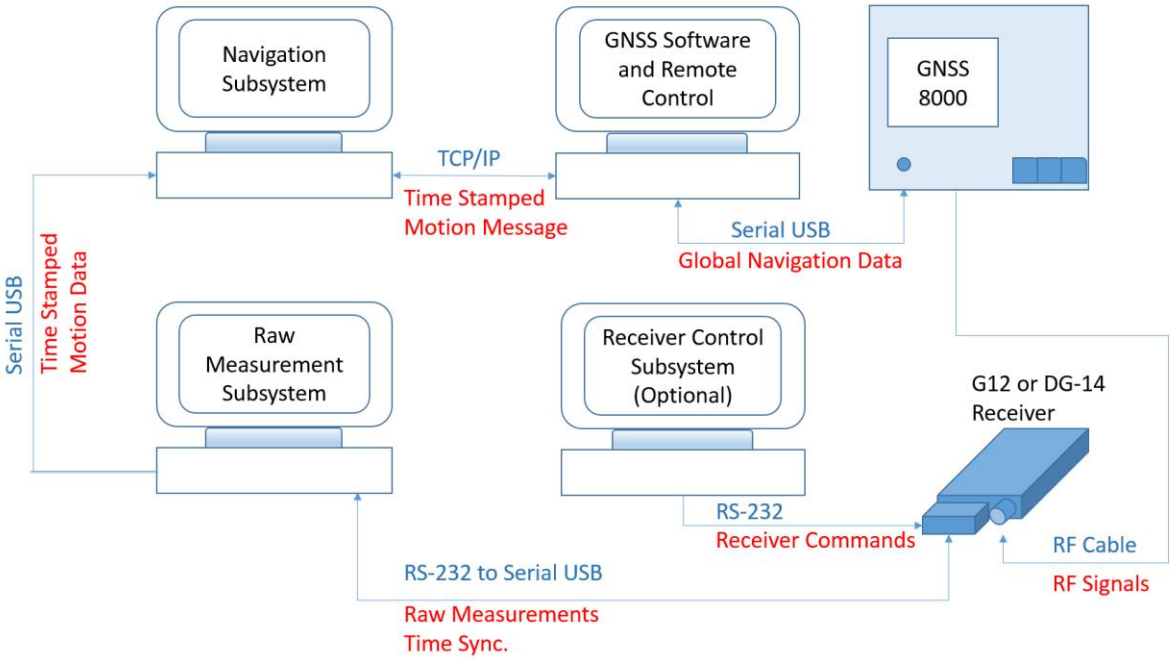

Figure 13: Full closed hardware-in-the-Loop formation testbed setup

The HWILFTB has a GNSS hardware RF signal simulator, the accompanying software user interface, a GPS receiver, a computer (CPU) containing the raw measurement section, and a CPU containing the navigation algorithm. Before the testbed starts, the GNSS software creates the chief vehicle trajectory and is exported to the navigation section. GPS ephemerides are uploaded to the software as well. The receiver control section or raw measurement section

changes the receiver settings and initializes a raw measurement data stream from the receiver. Lastly, the deputy vehicle parameters are set in the navigation section before the loop starts.

The overall loop processing is accomplished through a series of sequential steps involving each subsystem. The testbed starts at the navigation subsystem where the GNSS signal simulator software is initialized, and an initial position of the deputy vehicle is set. The GNSS software sends vehicle navigation data to the simulator for RF signal generation. RF signals generated are sent to the GPS receiver. Time synchronized raw measurement data at the current time produced from the receiver is sent to the raw measurement subsystem's computer. The raw measurement subsystem converts the raw data provided in WGS-84 coordinates and body frame into ECEF for the navigation subsystem to perform easy and quick calculations. Once the converted raw data is sent to the navigation subsystem, the raw measurement subsystem can be enabled to perform a secondary function that calculates range, pseudo-range, and Doppler of all the satellites visible to the receiver. The secondary function stores data on the raw measurement computer and is not exported to the other sections. The navigation subsystem receives the deputy ECEF data, predicts its trajectory to remain in formation, and create motion for the vehicle to achieve at the next time step. The predicted motion is packaged in a motion message and sent to the remote control section of the GNSS signal simulator software section. The remote control section determines if the motion message follows the laws of motion. If the signal simulator software accepts the motion on time with minimal to no position or velocity errors, the HWILFTB can repeat the cycle for the next time step with no RF signal distortion or loss. If there are inconsistencies detected between consecutive motion commands or the motion message is delivered late, the RF signal will be distorted. The HWIL testbed processes halt as a result and the loop is broken. A broken loop discontinues the simulation.

The next sections will describe the performance and functionality of the individual subsystems in the testbed. It is imperative that the subsystems' operations are accurate, efficient, and fast to prevent the motion message created at the end of each testbed iteration from expiring by the time the GNSS software at the next time step accepts it.

4.2 SimGen, GNSS sSgnal Simulator, and Remote Control Overview

The Spirent GNSS 8000 signal simulation software known as "SimGen" or "PosApp" is a powerful and versatile tool for creating nearly any type of vehicle scenario related to Earth. An image of the software user interface is displayed in *Figure 14*.

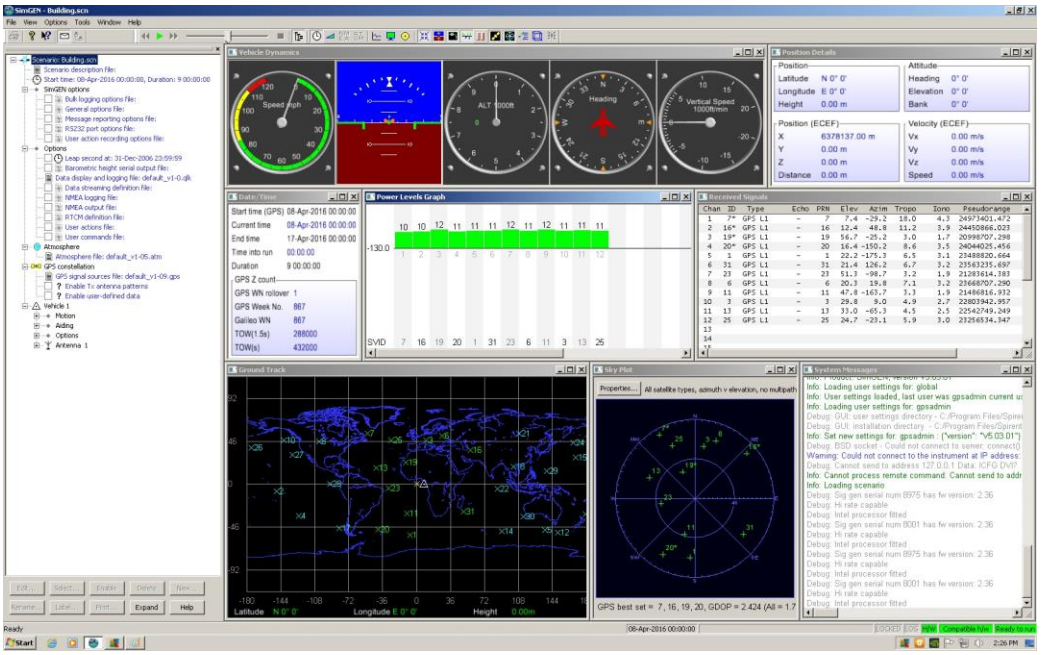

Figure 14: Image of GNSS signal simulator software user interface

The application contains many features to create, control and monitor vehicle motion and global navigation satellite constellations during simulation. SimGen allows the user to create scenarios for a static vehicle, simple motion model, aircraft, land vehicle, ship, spacecraft, remote control vehicle, or a combination of these vehicles. There are data logging features for time, vehicles, antennas, transmitters, and signals for post-processing analysis.

The GNSS hardware signal simulator RF box is synchronized to the SimGen software and the two function as a unit. When a scenario is started, navigation data is streamed to the simulator for RF signal creation. Spirent GNSS 8000 and 6560 simulators used in this work can model single or multi-vehicle simulations. The simulators are displayed in *Figure 15* and *Figure 16*.

Figure 15: GNSS 8000 Simulator

Figure 16: GNSS 6560 Simulator

The GNSS 8000 is a high dynamics GPS signal simulator and supports up to 20,000g signal dynamics, up to 120,000 m/s velocity, and one millisecond processing times [17]. The GNSS 8000 simulator has 3 RF carriers with one RF outputs per chassis [18]. This simulator supports GPS and SBAS (Space-based augmentations system). The Virginia Tech lab also has a GNSS 7800 signal simulator, which is a Galileo signal simulator that is master-slaved to the GNSS 8000 signal simulator. Multipath modeling, antenna gain between +20 to -49dB, and more are available also [18]. The GNSS 6560 is a high dynamic simulator and supports 12 channel GPS L1 C/A code. The simulator can create scenarios for velocities up to 15,000 m/s, accelerations up to 450 m/s^2, supports processing times up to 50 milliseconds, and allows antenna gains between +15 to -20dB [19].

The remote control subsystem is contained within the GNSS signal simulator software's remote control vehicle. Vehicle control is made through uploading motion commands with time stamps, motion commands without time stamps to be processed immediately, and through a device such as a lever or steering arm. The remote control motion commands are allowed to include information about position, velocity, acceleration jerk, heading, elevation, bank, angular velocity, angular acceleration and angular jerk in ECEF. Similar parameters can be uploaded for WGS-84 coordinates as well. Any motion in a specific command format can be applied to the software and will be accepted. The RF signals produced by the GNSS signal simulator will not be stable if any random motion is submitted, however. The updated trajectory needs to follow the laws of motion for the RF signal to remain stable.

4.2.1 Creating A Scenario

Scenarios establish the environment for vehicles to operate in during simulations. Once the application starts, a scenario is created for the current day. The start and end times of a

scenario must be established first. The window for displaying start time and duration is shown in *Figure 17*.

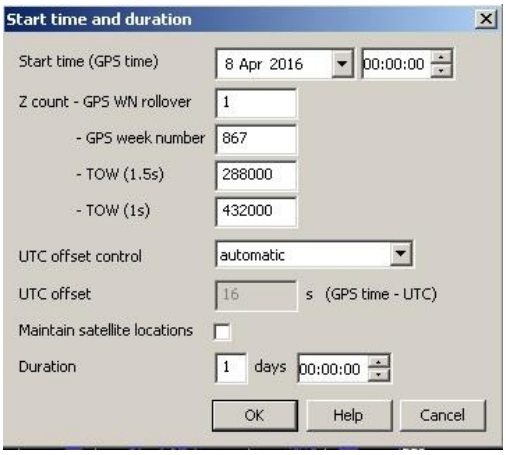

Figure 17: Options window for setting start and stop of scenario

The iteration rate is the next option that needs to be set under the general options and data streaming definition options. The iteration rate is set at the fastest rate the simulator is capable of achieving. The iteration rate is set for one millisecond on the GNSS 8000 simulator and 50 milliseconds on the GNSS 6560. The windows for iteration rate is displayed in general options, and data streaming are shown in *Figure 18* and *Figure 19*.

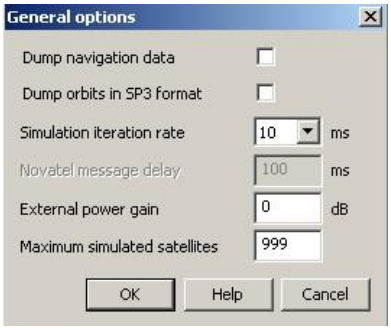

Figure 18: Options window for setting the iteration rate of a scenario

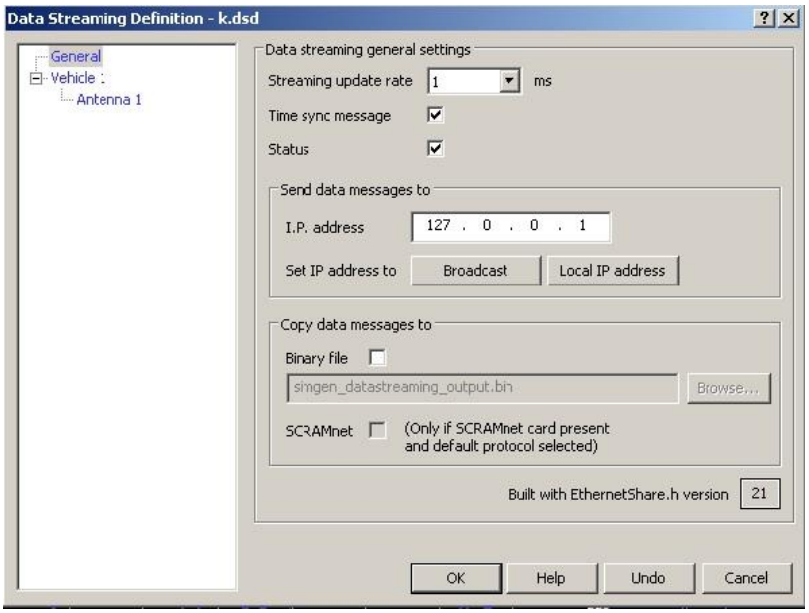

Figure 19: Options window for setting the iteration rate for data streaming in a scenario

Ephemeris data uploaded to SimGen generates accurate positions of GPS satellites for raw data and receiver location measurements. Navigation files (.N) are processed by the software, and the satellite orbits are updated through the GPS signal sources option. *Figure 20* shows the window for the GPS signal sources options and the window for uploading navigation files. NASA provides a database of observation and navigation files accessible by the public. The navigation file can be obtained by visiting ftp://cddis.gsfc.nasa.gov/pub/gps/data/daily/. Once the navigation file for the desired day and year is selected, it can be uploaded as a RINEX (Receiver Independent Exchange Format) file in the GPS signal sources options menu. RINEX has been developed for the exchange of GPS data collected during the large European GPS campaign in 1989 [20].

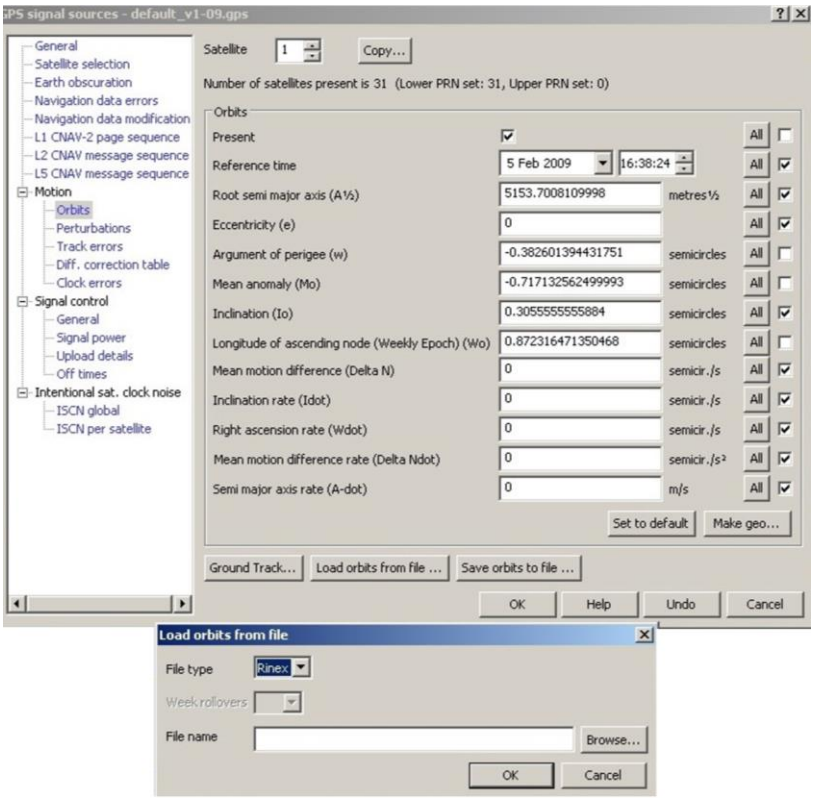

Figure 20: Options window for uploading a navigation file for GPS satellite orbits of a scenario

The choice of vehicle is the last of the initial setting that should be established. The options are different for each vehicle. The chief vehicle is represented as a static vehicle, simple motion model, land vehicle, aircraft, ship, or a spacecraft depending on the scenario. Vehicle paths and dynamic limits can be established through the motion command and personality options. Each vehicle has different methods of propagation and dynamic limits. All deputy vehicles are represented as remote control vehicles. Remote vehicle commands are sent to SimGen through a TCP/IP configuration. *Figure 21* displays n example of the motion command and personality options.

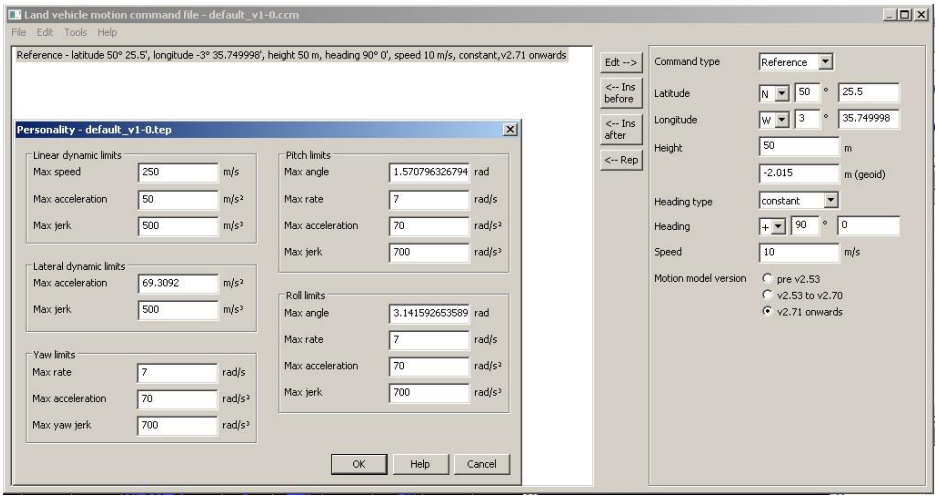

Figure 21: Options Window for Motion Commands and Personality Profile Example of Vehicles in a Scenario

Power adjustment and data logging options are optional but are set before or after a simulation starts. The power adjustment alters the antenna gain of vehicles within the simulation. Maximizing the gain boosts the signal stability and allows the RF signal to be stable under high dynamics when the receiver obtains it. The data logging option allows for post-simulation analysis of vehicle, signal, time, and transmitter data. The data-logging feature is used to create chief vehicle trajectories for deputy vehicles to follow.

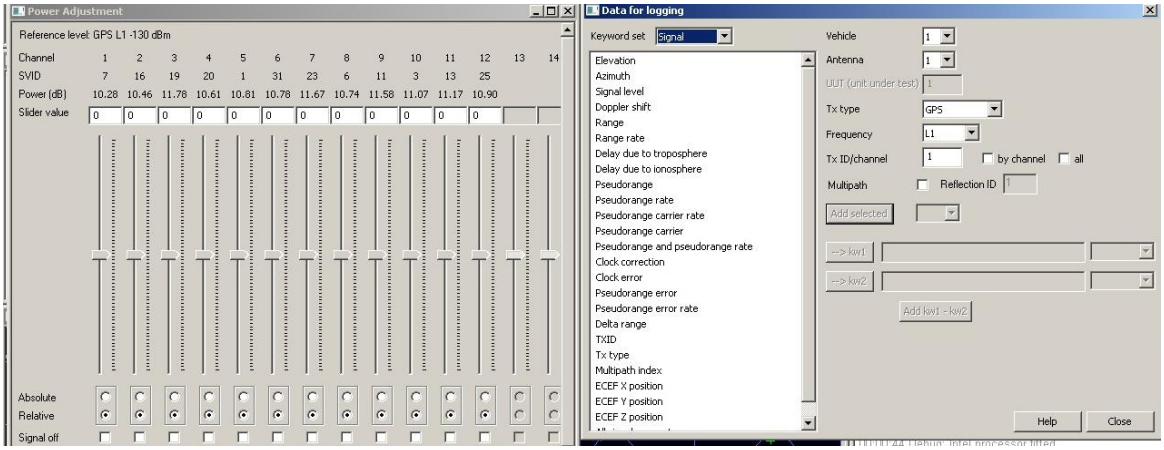

Figure 22: Options window for power adjustment and data logging during a scenario

4.3 Ashtech G-12 and Ashtech DG-14 Receivers

As discussed in earlier sections, receivers produce navigation messages comprised of time-stamped position, velocity, and raw measurement data. This information provides a sense of direction and location for the vehicles it monitors. The testbed uses Ashtech G-12 and Ashtech DG-14 receivers for providing these navigation messages to the deputy vehicles. An image of the receivers is displayed in *Figure 23*.

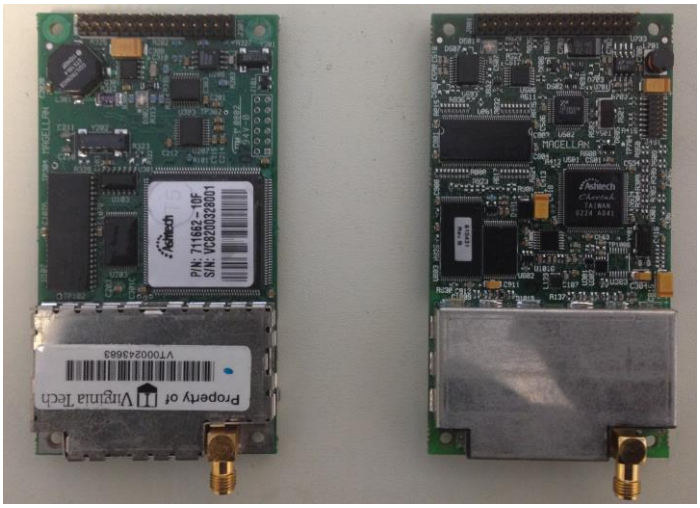

Figure 23: Ashtech G-12 on left | Ashtech DG-14 on right

The time-stamped message is synchronized to the GNSS software once one satellite is acquired from the RF signal produced by the signal simulator. The navigation message is displayed after four satellites are acquired. The location accuracy and proper time synchronization play a significant role in the HWILFTB. Inaccuracies in location and non-synchronized time messages will cause a broken loop.

Up to three Ashtech receivers are available for testbed simulation in this work. The G-12 receiver has 12 channels and can display three-dimensional position and velocities within 16 meters or less [20]. This receiver has 60-second cold starts to obtain one satellite. The position's accuracy can reach one centimeter or less, and the velocity accuracy can reach one centimeter

per second if the differential mode is enabled [20]. The differential mode allows the receiver to use corrections provided by an autonomous solution. Although this feature is unavailable for this work, local radio or beacon receivers can be used as the autonomous device to enable these higher accuracies. The G-12 supports up to 20g accelerations and jerks up to $5m/s^3$. The device has many features for receiving data and improving performance. The DG-14 receiver has similar performance to the G-12 receiver, but its altitude is limited to 60,000 ft. The G-12 and DG-14 have the ability to produce multiple types of National Marine Electronics Association (NMEA) data messages. The POS, GSN, and PER commands are used in this testbed.

The POS command provides the navigation message in WGS-84 and body frame units. The parameters of the POS command include the current UTC (Coordinated Universal Time), latitude, longitude, altitude, true track angle, speed over ground, vertical velocity and DOP information.

The GSN command provides the satellite numbers and signal to noise ratios of the satellites acquired by the receiver. These two messages are the base of generating future positions, range, pseudo-range, and Doppler of the deputy vehicle. The receiver can produce range, pseudo-range, Doppler, and carrier phase information. However, the raw data is encrypted with a proprietary almanac ANSI/ASCII format. Calculation of the raw measurements can be eliminated if the data can be decoded.

The PER message allows the receiver to change the position update rate option up to 10 Hz (0.1 seconds). Other options of choosing the elevation and DOP masks are available but are not vital for testbed functionality.

4.4 Raw Measurement Subsystem

The raw measurement subsystem's primary purpose is to decode the NMEA messages exported from the receiver, perform coordinate frame conversions to ECEF, and submit the converted data to the navigation section. An optional secondary command calculates the range, pseudo-range, and Doppler values since the parameters are not directly obtainable from the receiver. The process of the raw measurement subsystem is shown in Figure 24. The raw measurement subsystem is in the form of an algorithm created with MATLAB shown in Appendix 1. MATLAB is a powerful mathematical software that covers a broad range of user applications and allows for large and complex data processes to be quickly processed. The computer's processor handling the raw measurement algorithm is an Intel® Core™ i7-4700MQ CPU @ 2.4 GHz on a 64-bit operating system with 8 GB of installed RAM.

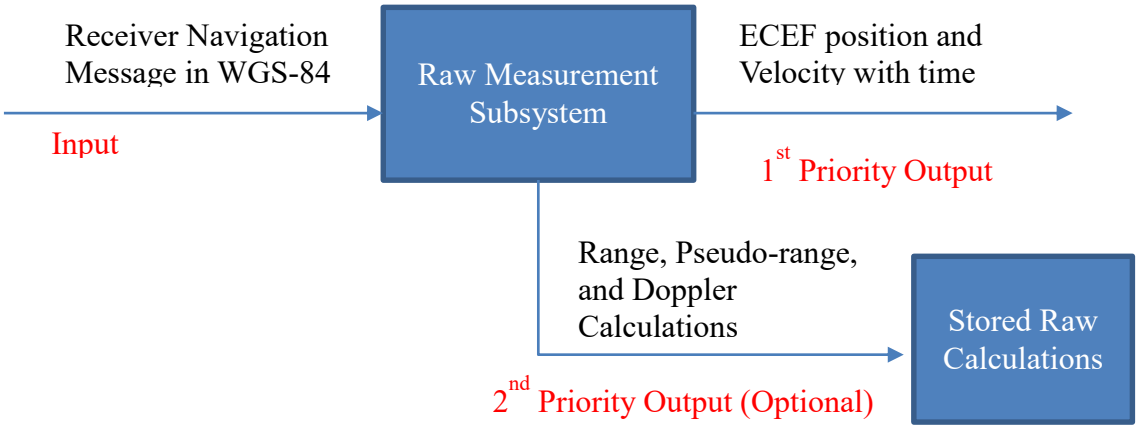

Figure 24: Raw measurement subsystem processes

There are several processes within the raw measurement section. Processes in this subsection follow the theory outlined throughout Chapter 2. First, constants and parameters are established for the time, ephemeris data, parameters of the Earth, and satellite locations. The first step also creates a database for storing calculated raw data. Second, a data stream import and

export are established for receiving data from the receiver and exporting data to the navigation section. The imported data is either a POS or GSN message and is characterized by the NMEA message type. In the third step, the POS message with WGS-84 and body frame units is decoded. Next, the decoded values are converted to the ECEF position and velocity of the vehicle. The position, velocity, and time are exported to the next subsystem at the end of this step. If the optional command for decoding the GSN message is enabled, the GSN command is decoded after the ECEF position and velocity are sent. The data of the GSN command is not exported during the loop. This reasoning suggests that the second command should occur after the primary command is finished to reduce the processing time required during each iteration within the testbed loop. Once the GSN message is decoded, the raw values are calculated for each satellite listed in the message. These values are then stored in the database. The raw measurement section performs both the primary and secondary functions in 0.02 seconds for the first iteration and approximately 0.015 seconds for successive iterations using the processing configuration. It takes 0.022 seconds for the first iteration and 0.012 seconds for successive iterations if the system is reconfigured only to perform the primary function.

4.5 Navigation Subsystem

The navigation subsystem is responsible for using the time, position, and velocity provided in ECEF units by the receiver to generate future motion that is provided to the remote control for SimGen to follow. The testbed loop is started from this subsystem, and the initial position of the deputy vehicle is also specified here. This subsystem is also in the form of an algorithm using MATLAB on a separate computer from the raw measurement subsystem. Data is received through a serial port connection from the computer of the raw data section and is exported to the SimGen software by Ethernet through a TCP/IP connection. The navigation

section's computer processor is an Intel® Core™ i5-6300U CPU @ 2.5 GHz on a 64-bit operating system with 4 GB of installed RAM. Although the processors between the subsystems are different, there should not be too much difference in the processing time. Code for this section is viewable in Appendix 2.

There are several processes in the navigation subsystem. The target path, deputy motion constraints, and range of possible accelerations consistent with the deputy's constraints. Second, a data stream import from the navigation is established through a serial connection and a data stream export to SimGen's remote control section is established through a TCP/IP connection. A wait command is installed to halt the iteration loop of the navigation section and allow the receiver to acquire 4 or more satellites. Once the receiver locks into the required number of satellites for a navigation message, data is received from the raw measurement subsystem. The data from the raw measurement subsystem is used to generate a solution at the next time step. This solution follows the fundamentals discussed in Chapter 3. Lastly, the future solution for the next time step is packaged and sent to the remote control section. This step finalizes the processes of the testbed and the loop restarts for the following time step. The navigation subsystem performs both the primary and secondary functions in 0.08 seconds for the first iteration and approximately 0.031 seconds for successive iterations using the processing configuration.

4.6 Resolving Subsystem Integration Issues

There is a problem using the receiver with its available accuracy within the loop. The receiver has positional errors that can reach up to 16 meters. These errors constantly occur and are not tolerated by SimGen's remote control features if the receiver's information is fed to the algorithm.

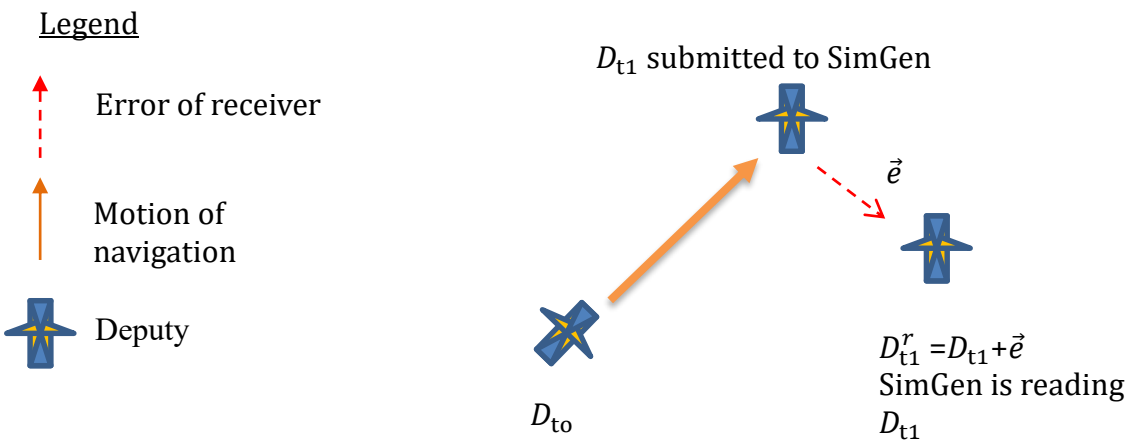

Figure 25: First iteration of testbed loop including receiver data

If motion is generated to move a vehicle from its position at time t_o to the actual position at time t_1, the receiver will measure the position at t_1 with some added error $\vec{e}$ shown in *Figure 25*. However, SimGen will read the value of the position specified to occur at t_1 by the navigation algorithm.

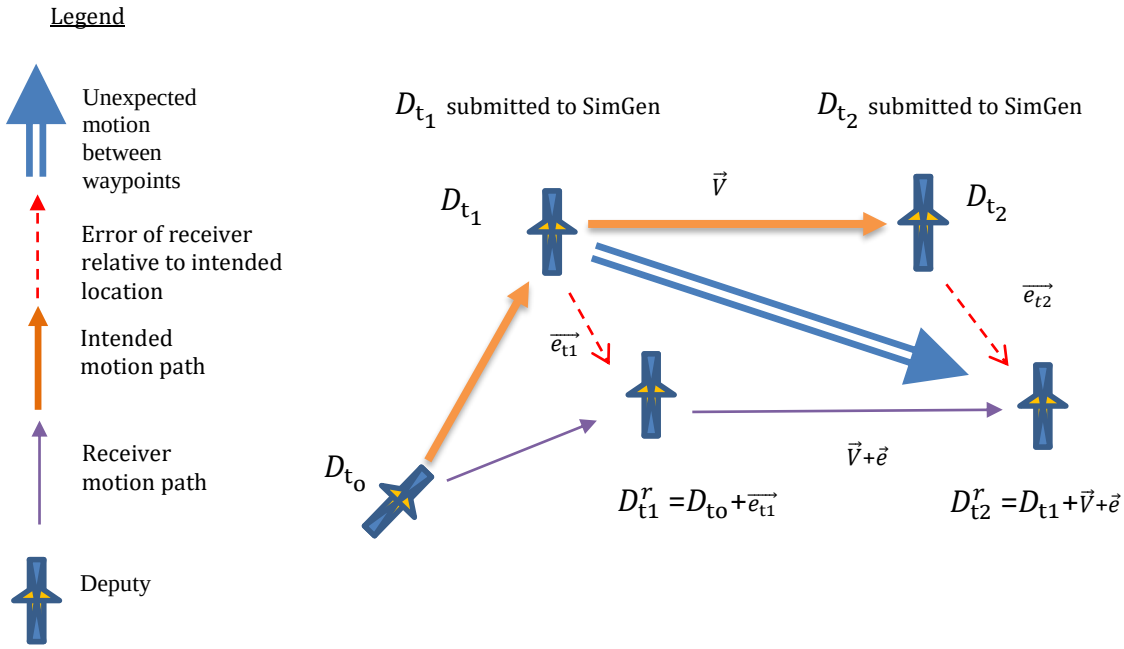

Figure 26: Second iteration of testbed loop including receiver data

The loop breaks at the next step demonstrated in *Figure 26*. If the vehicle location measured by the receiver at t_1, $\boldsymbol{D}_{t1}^{r}$, is used as the basis of motion for producing a position at t_2, SimGen reveals the error. This error occurs because SimGen does not know how a different position submitted by the algorithm was calculated at time t_2 when compared to the position SimGen calculated. The errors can occur in both position and velocity and drastically affect the stability of the RF signal produced by the simulator. If the magnitude of the error provided by the receiver is not kept to below a tolerable threshold, the receiver will not be able to acquire satellites from the RF signal. Navigation messages are not produced when fewer than four satellites are acquired. The G-12 and DG-14 receivers provide high errors in the navigation message and cause complete loss of the RF signal when used in the testbed. This issue deems the current configuration of the navigation subsystem and raw measurement subsystem useless since no data will be provided from the receiver.

An alternative solution for the loop to function properly is to use the synchronized time given by the receiver in the POS NMEA message and only use the dynamics produced by the navigation algorithm. Excluding the dynamics provided by the receiver eliminates all position and velocity errors. In this configuration, the dynamics sent to SimGen will follow the laws of motion and the loop can function without issues. All processes can still occur with only producing motion when based on the navigation algorithm dynamics. The original configuration using the Ashtech receiver will work correctly if it is used in differential mode.

Chapter 5: System Component Testing and Tested Simulation Experimentation

5.1 Testing Methodology

This chapter outlines the testbed component limitations and scenario testing. The restrictions of the system must be established to maximize vehicle performance in testbed simulations. The measurement issues involving the receiver and raw measurement section will be quantified. This chapter also quantifies the remote control tolerances. Testing of navigation subsystem's parameter effects on vehicle trajectories is the last testbed component examined. After all the system limitations and parameters are determined, different vehicle scenarios of the full testbed are demonstrated. The testbed scenarios will display the versatility, accuracy, and capabilities of this setup.

5.2 System Component Testing

Individual components of the testbed will be tested in this section to establish system limits and tolerances. The SimGen software and GNSS signal simulator are assumed to act ideally and are not subjected to testing.

5.2.1 Remote Control Tolerance

SimGen's remote control receives motion commands from the navigation subsystem to move the vehicle during simulations. If the motion provided during subsequent iterations does not match uploaded motion, there will be RF signal distortion. Section 4.6 mentions this process. The error affects the RF signal immediately. The following tests place errors randomly in a motion trajectory to measure the RF signal response of the receiver after it begins producing a navigation message. The purpose of this test is to determine the allowable error the remote control section will accept without distorting the RF signal.

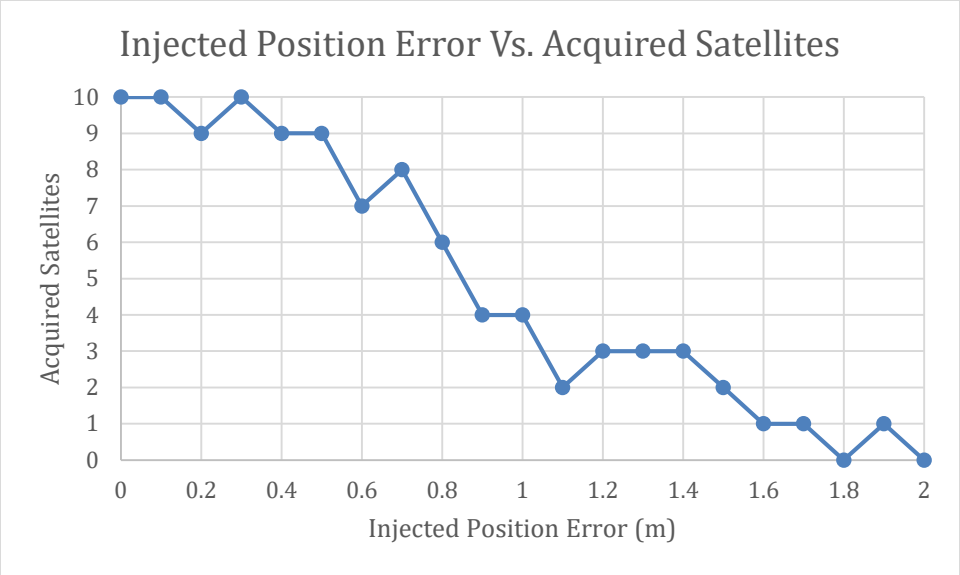

Figure 27: Injected Position Error for Determining RF stability

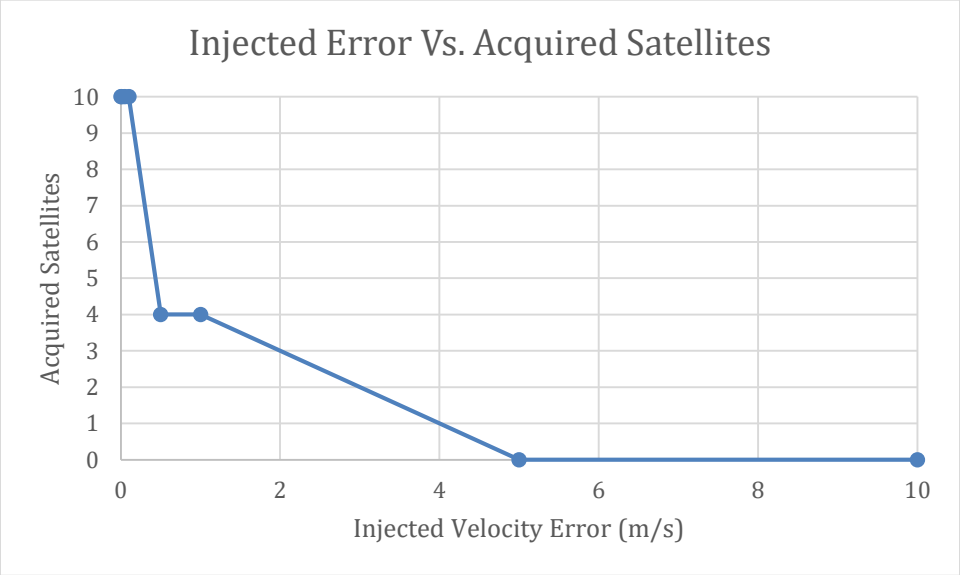

Figure 28: Injecting velocity error for determining RF signal stability

Figure 27 and *Figure 28* indicate two tests for position and velocity error effects on RF signal stability for motion uploaded to SimGen's remote control. The tests show that the receiver is attempting to acquire up to 12 satellites, but errors are injected at random times. Certain errors cause no change or drastic loss in the receiver's acquired satellites. Receiver loss occurs around

0.2 m of injected position error or between 0.1 to 0.5 m/s of injected error. There is a direct relationship between the increase in error applied and the loss in satellites.

5.2.2 Receiver Performance

The Ashtech G-12 and DG-14 receivers are space-qualified receivers used in this testbed. Using the receiver to measure a vehicle's position comes with some error. If this measurement is used as the starting position at each time step, this may cause an error within the remote control section as explained in section 4.6. The following test used the receiver to measure the position and velocity of different vehicle scenarios. The purpose of this test is to determine how much error the receiver causes at each time step during different scenarios. The receiver is tested for a stationary vehicle, a car traveling at 100 m/s, an airplane traveling at 500 m/s, and a spacecraft traveling at 8 km/s. Ten-minute measurements of a vehicle were gathered for each second and average for each vehicle type. WGS-84 and ECEF errors were averaged and compared.

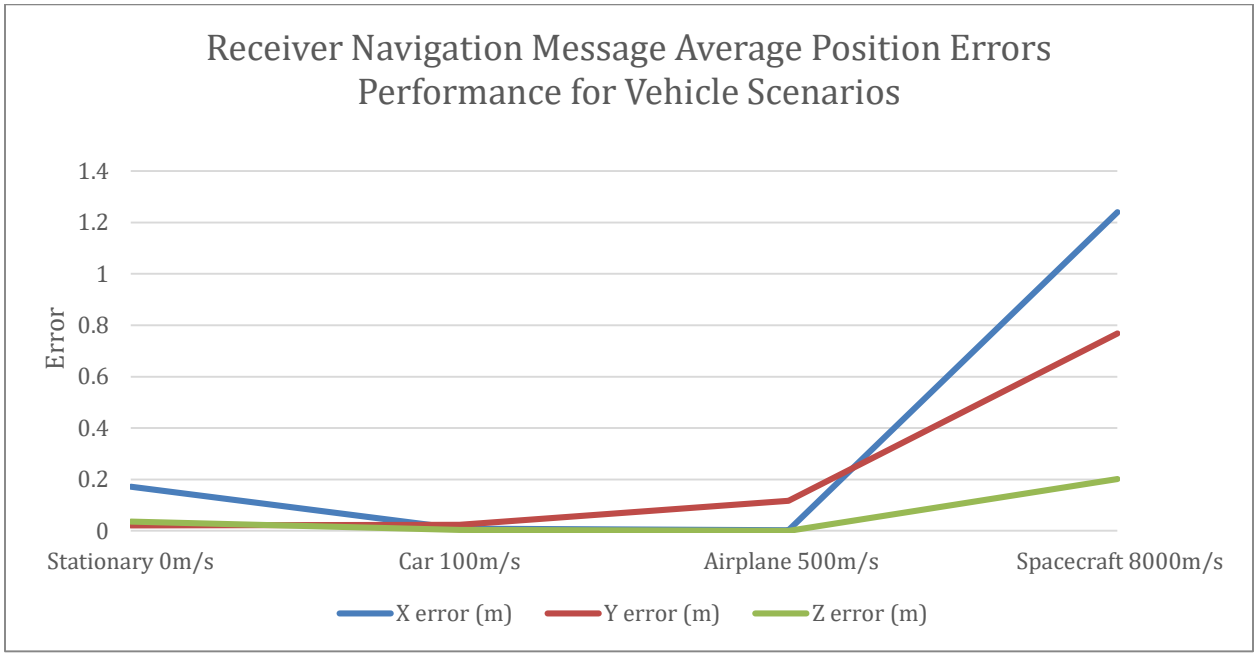

Figure 29: Receiver Navigation Message Error for Vehicle Scenarios

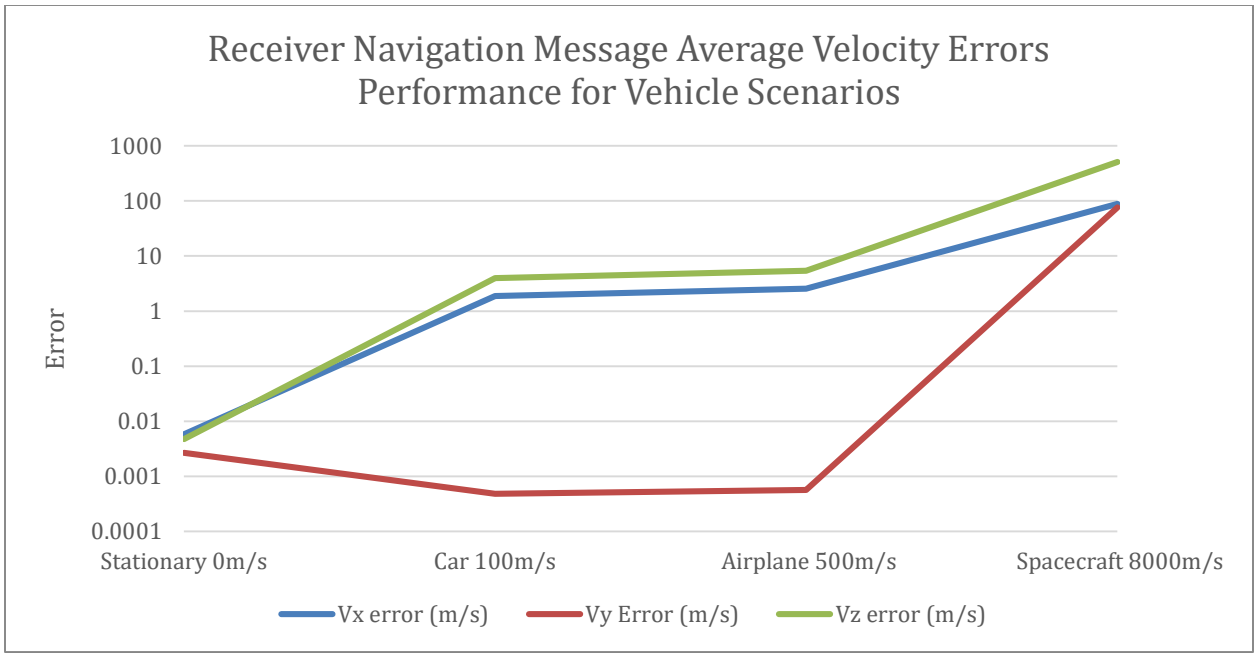

Figure 30: Receiver Navigation Message Error for Stationary, Car, and Airplane Scenario

Figure 29 and *Figure 30* display the receiver error for a stationary, car, airplane and spacecraft vehicle. For a stationary vehicle, the main error comes from the altitude measurement. The error in altitude averaged to be 0.17 m per sample. The overall slant distance between the actual position and the receiver measurement is approximately 0.18 m per sample. The errors of the receiver increase as the vehicle speed increases. This error arises mainly from the estimate of velocity. The velocity magnitude ranges from 0.007 m/s per sample with a stationary vehicle to 520 m/s per sample with a spacecraft vehicle on average. The remote control section detects the error immediately and will reject RF signal communication if the error causes too much inconsistency with the actual position. The average distance per sample between the measurement and actual location for a stationary vehicle, car, airplane, and spacecraft is 0.18 m, 0.03 m, 0.12 m, and 1.47 m respectively. The average velocity difference per sample between the measurement and actual location for a stationary vehicle, car, airplane, and spacecraft is 0.007 m/s, 4.41 m/s, 5.99 m/s, and 520 m/s respectively. These errors are not compatible with SimGen

remote control that will cause RF distortion for errors in position above 0.2 m and velocity between 0.1 to 0.5 m/s. This tolerance shows a vehicle cannot use the G-12 receiver to measure the location of a vehicle and use that measurement to produce motion for itself at the next time step because the receiver will lose lock during the next time step. Once the receiver loses lock, the loop will be broken because components down the line of the testbed rely on consistent updates from the receiver to provide continual motion to the vehicle. Fortunately, this does not restrict one from allowing the testbed to rely on the receiver's synchronized time or using the receiver measurement to measure motion from one vehicle and use it to move another vehicle.

5.2.3 Raw Measurement Performance

The raw measurement subsystem is responsible for converting the receiver's data from WGS-84 and body frame units to ECEF units and performing raw measurement calculations for the range, pseudo-range, and Doppler of each satellite visible to the receiver. There are accuracy issues in the coordinate conversions and calculation of the raw data. The following test recorded data of a vehicle from SimGen's software in WGS-84 units, body frame units, and ECEF units. The WGS-84 and body frame units were uploaded to the raw measurement section to determine the accuracy of the section's ECEF conversions and raw data results. Results of the test are shown in *Table 2*.

Table 2: Raw measurement section error after ECEF conversion and raw data calculation

X (m)	Y (m)	Z (m)	Vx (m/s)	Vy (m/s)	Vz (m/s)	R (m)	P (m)	D (Hz)
4.42E-05	3.44E-04	-2.63E-05	-2.27E-01	-7.84E-05	-4.83E-01	-1.57E+00	6.12E+01	1.63E+00

ECEF position conversion accuracy of the subsystem is within 10^{-4} meters. The velocity accuracy is poor in the conversions for the X and Z directions. This error would not be tolerable by the remote control. The range has reasonable accuracy within two meters and the Doppler

calculations show reasonable accuracy within two Hertz. However, the pseudo-range calculation is within two orders of magnitude relative to the actual value. This error can be corrected if the proper time errors were accounted for or if the ephemeris data used for the calculations is more accurate.

5.2.4 Navigation Subsystem Vehicle Parameter Effect Testing

The navigation subsystem is responsible for taking the present dynamics of a vehicle and determining the motion for the vehicle to undertake at the next time step. The laws of motion and vehicle parameters determine the motion. The initial vehicle parameters are the time step of each motion, the maximum velocity of the vehicle, the velocity cutbacks and distances for the zoning method, the maximum and minimum accelerations, the maximum and minimum jerks, the turn angle restraints, the incremental control of acceleration, the initial acceleration, the initial velocity, and the initial position.

The following test will demonstrate the effects each parameter has on formation keeping and control for a vehicle. A deputy vehicle is made to follow an airplane trajectory that travels at a constant 250 m/s and a constant altitude of 1 km with specific initial parameters. The path is shown in Figure 31. Each parameter excluding the initial position, velocity, and acceleration, is altered to view the effect the parameter has on vehicle's overall formation keeping and control performance.

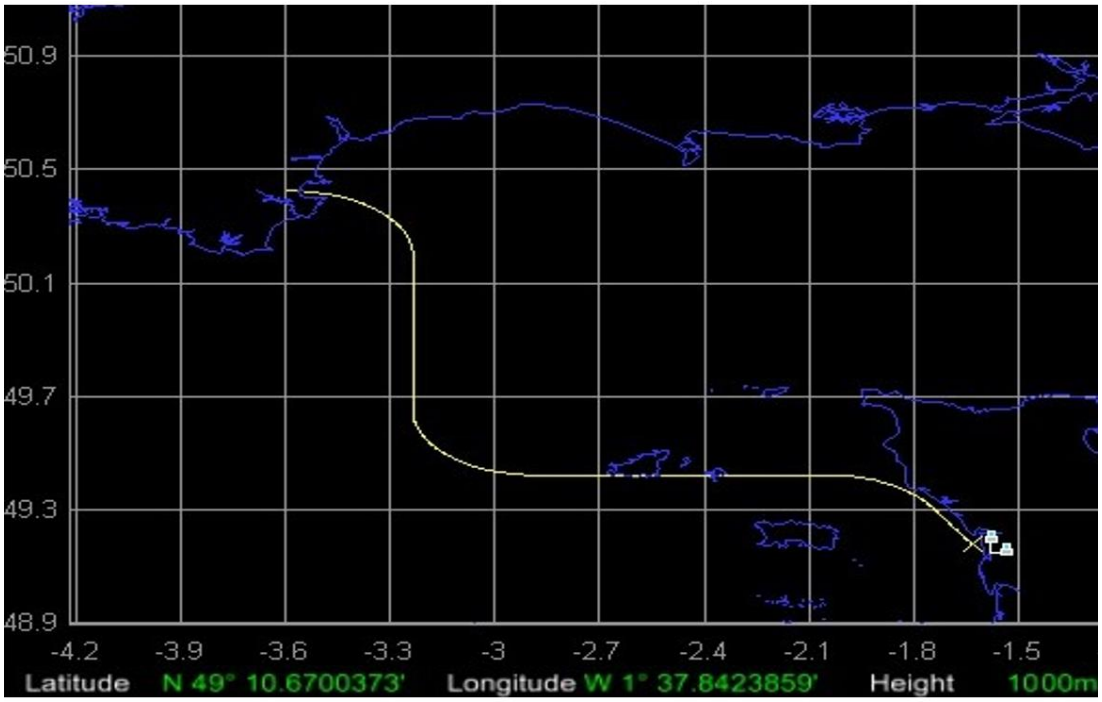

Figure 31: Chief path for vehicle to follow on 2D world map

The initial conditions and performances of those conditions for the vehicle and formation control and keeping are shown in Table 3 and Figure 33.

Table 3: Initial conditions for vehicle parameter test

Time Step	Maximum Velocity	Zone 1 Cutback	Zone 2 Cutback	Maximum Acceleration	Minimum Acceleration	Max./Min. Jerk	Turn Angle	ΔA
1 s	260 m/s	<50m, 256 m/s	<20m, 252 m/s	25 m/s^2	-25 m/s^2	+25/-25 m/s^3	360°	1.25 m/s^2

The initial conditions were selected to allow the deputy to be relatively in formation with a desired trajectory. The vehicle begins with 480 m/s velocity and no acceleration to achieve a separation of 1.4 km before it begins entering its true formation-keeping phase. The deputy vehicle is purposely made to move at an angle initially to generate a separation distance. A diagram showing the starting positions of the deputy and chief vehicles are shown in Figure 32.

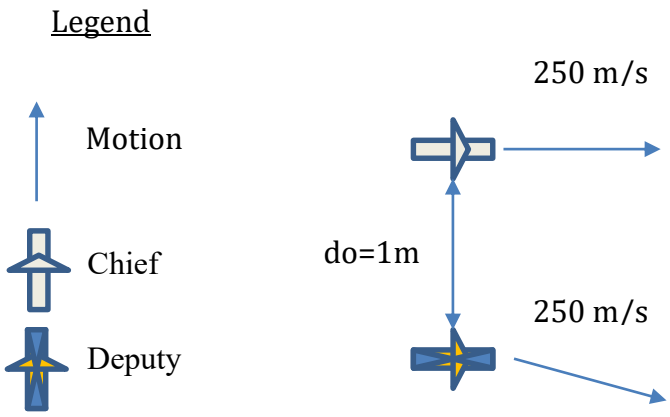

Figure 32: Initial setup of benchmark case for navigation subsystem testing

The formation-keeping phase occurs when a vehicle attempting to reach a formation decreases the separation between itself and a time-based waypoint. Depending on the scenario and initial parameters, the formation acquisition phase can take a long time to reach the formation-keeping phase. The formation-keeping phase occurs when a vehicle is within a specific separation distance and consistently keeps that distance to a minimum for the remaining travel. The formation-keeping phase is based on the GPS accuracy requirement and will `begin when a vehicle is within 30 meters of the formation.

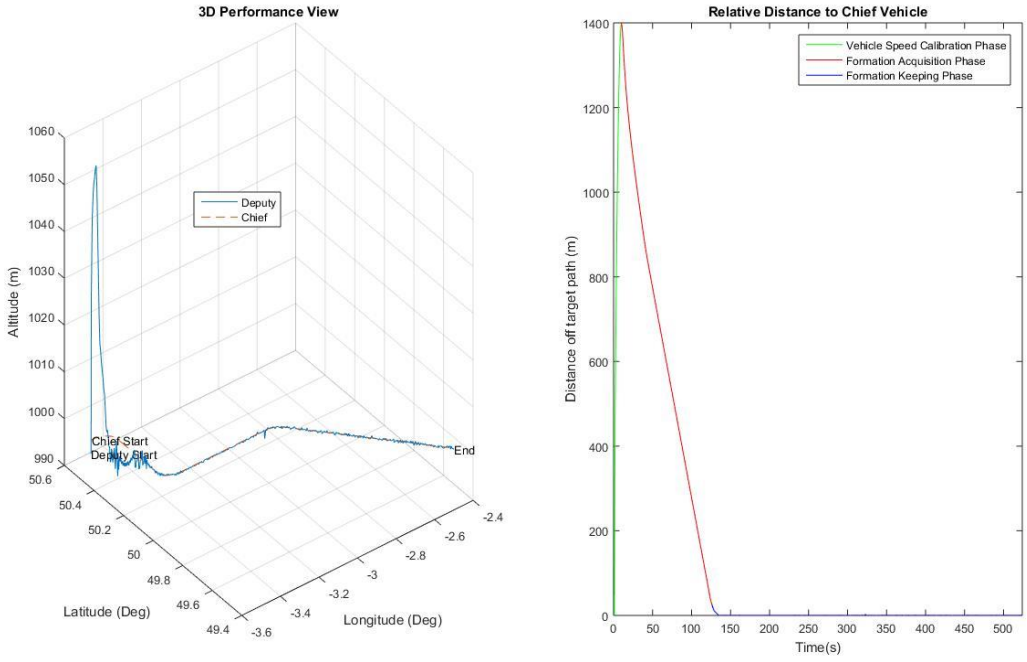

Figure 33: Formation control and keeping performance from initial parameters of benchmark case

Figure 33 shows the results of a vehicle attempting to reach a desired formation and keeping the formation once reached. The 3D view on the left shows the performance of the deputy trajectory relative to the chief trajectory path. Towards the beginning of the flight, the dark spots indicate that the deputy vehicle is slightly off path. This shows the formation acquisition phase of the vehicle as it tries to achieve formation. After the acquisition phase, the rest of the path has few to no dark spots which means the deputy vehicle is within the formation with little error. The initial conditions allow the deputy vehicle to incur a separation distance of 1400 m. It takes 126 seconds for the vehicle to finish its acquisition phase and keeps the formation within an average and standard deviation of 0.6 m and 2.2m, respectively.

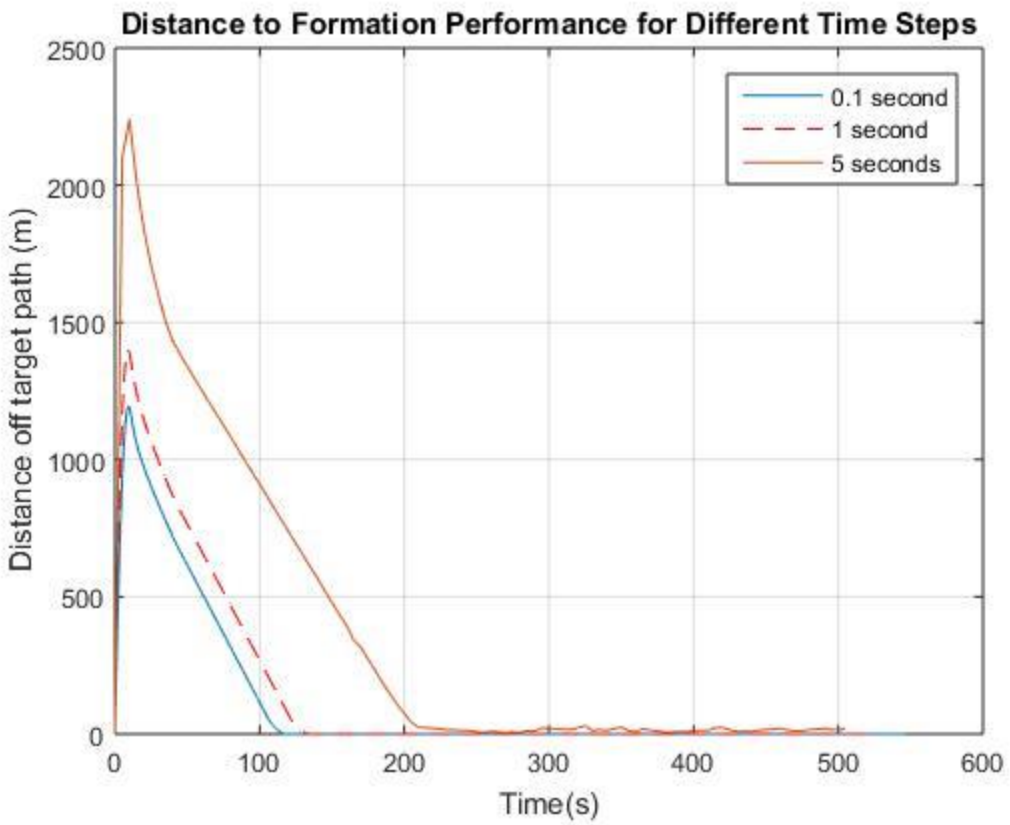

Figure 34: Formation control and keeping performance for different time steps. Benchmark case is dotted line with initial separation of 1 m and maximum separation of 1.4 km

Figure 34 shows the effects of changing the frequency of applying motion to the vehicle. The test investigated the performance differences between the original time step of one second to 0.1 seconds and 5 seconds. Poor control of the vehicle occurs when a large time step is used. The deputy vehicle is not able to capture and keep up with the intricate motion of the chief trajectory. This poor control mainly occurs when the chief trajectory undergoes a turn. For five second motion uploads, the target acquisition phase begins with a separation distance over 2241 m and takes over 200 seconds to obtain the formation which is ultimately lost. 0.1-second motion injections have the opposite effect. The quicker injection times increase coverage and agility of the deputy vehicle. The separation distance is 1194 m and it takes 100.4 seconds for the vehicle to finish the acquisition phase. The average and standard deviation of the 0.1-second time steps

are 0.21 m and 1.91 m respectively. The average and standard deviation of the 5-second time steps are 15.2 m and 6.6 m respectively. Using smaller time steps drastically increases the performance of the deputy vehicle.

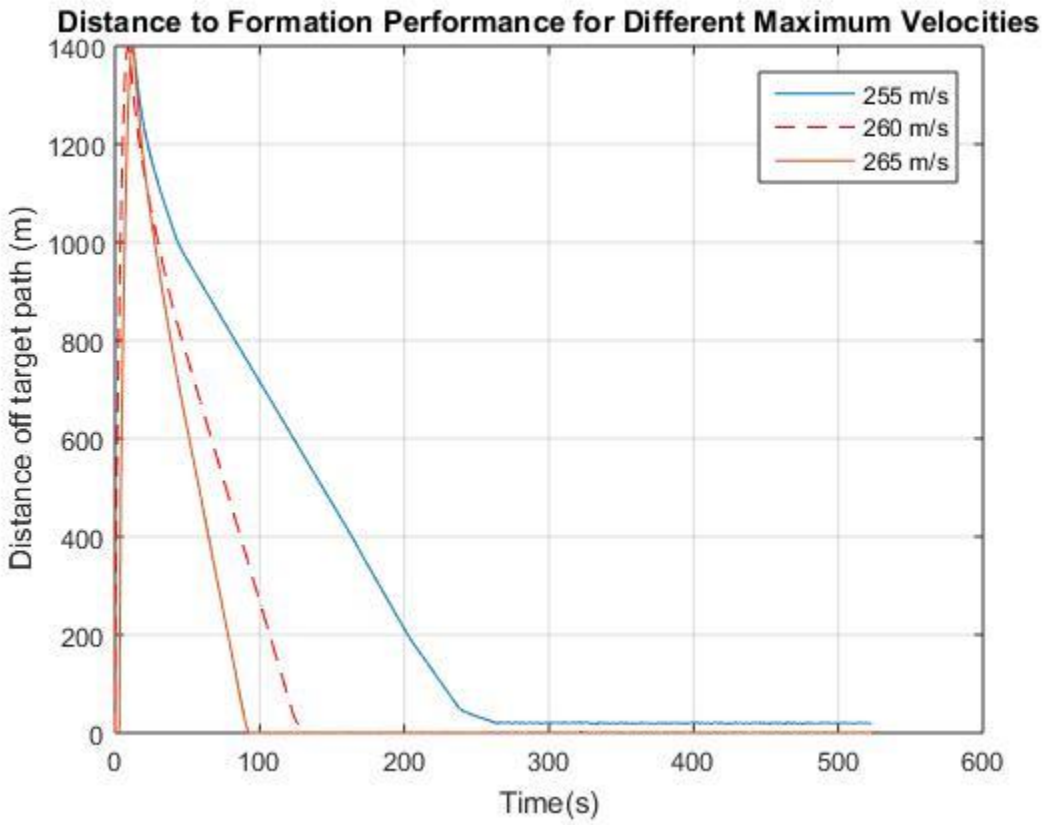

Figure 35: Formation control and keeping performance from different maximum velocities. Benchmark case is dotted line with initial separation of 1 m and maximum separation of 1.4 km

Figure 35 shows the effects of changing the maximum velocity the deputy vehicle is allowed to reach during a simulation. The velocity allows a vehicle to reduce the time of the acquisition phase but costs control generally. Maximum velocities of 255 m/s and 265 m/s are compared to the performance of the vehicle with a maximum velocity of 265 m/s. Smaller maximum velocities take longer for the vehicle to get into formation and provide less accuracy when keeping the formation. The target acquisition phase of the 255 m/s maximum velocity

begins with a separation distance of 1.4 km and takes 242 seconds to obtain the formation. The average and standard deviation for the formation-keeping phase are 20.1 m and 1.25 m, respectively. The 265 m/s parameter shows a faster acquisition phase and more control of the formation. The separation distance is essentially the same as the 255 m/s parameter, but it takes 78 seconds for the vehicle to finish the acquisition phase. This higher velocity parameter is three times faster than the smaller velocity parameter. The average and standard deviation of the 265 m/s maximum velocity are 0.37 m and 1.26 m respectively. Using high maximum velocity increases the performance of the deputy vehicle. However, using a maximum velocity too large in a different scenario may produce poor performances in control.

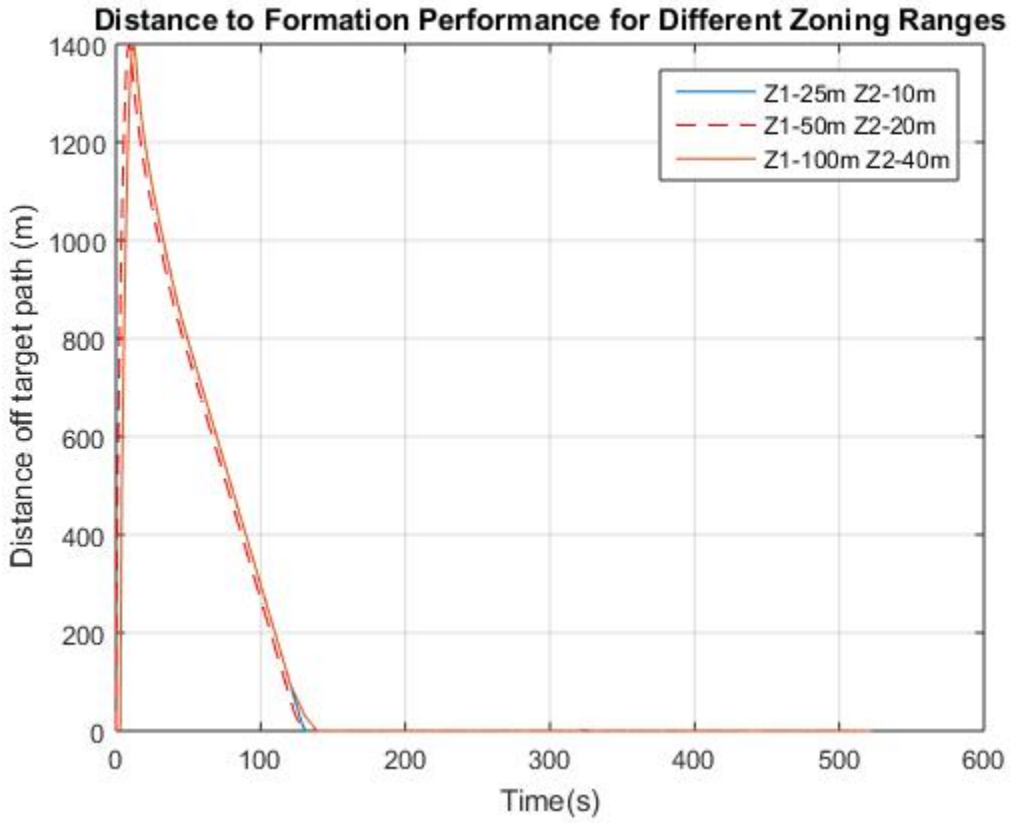

Figure 36: Formation control and keeping performance for different zoning parameters. Benchmark case is dotted line with initial separation of 1 m and maximum separation of 1.4 km

Zoning parameters allow the vehicle to reduce its maximum velocity within certain separation distances from the formation. Typically, under high dynamic scenarios, the zoning distances and velocity cutbacks prevent the vehicle from overshooting a waypoint. If the zoning cutbacks are performed farther away from the target position, the deputy vehicle will take longer to catch up to the required position at its reduced speed. If the zoning cutbacks are performed too close to the target position, the deputy vehicle may overshoot the target point because it is not able to reduce its speed properly in time. The effects of zoning are seen in the transition between the formation keeping and acquisition phases. The maximum separation distances are identical for all three cases. Zoning cutbacks performed further away from the vehicle yields a longer transition phase. The time of the acquisition phase for both the doubled and halved zoning distance are 119 seconds and 116 seconds. Due to the longer duration with a slower speed produced by the doubled zoning distance, the formation keeping phase performs worse than the benchmark and half zoned distance. The average and standard deviation of the doubled zoning distance are 0.57 m and 2.25 m respectively. The average and standard deviation of the halved zoning distance are 0.39 m and 1.09 m respectively.

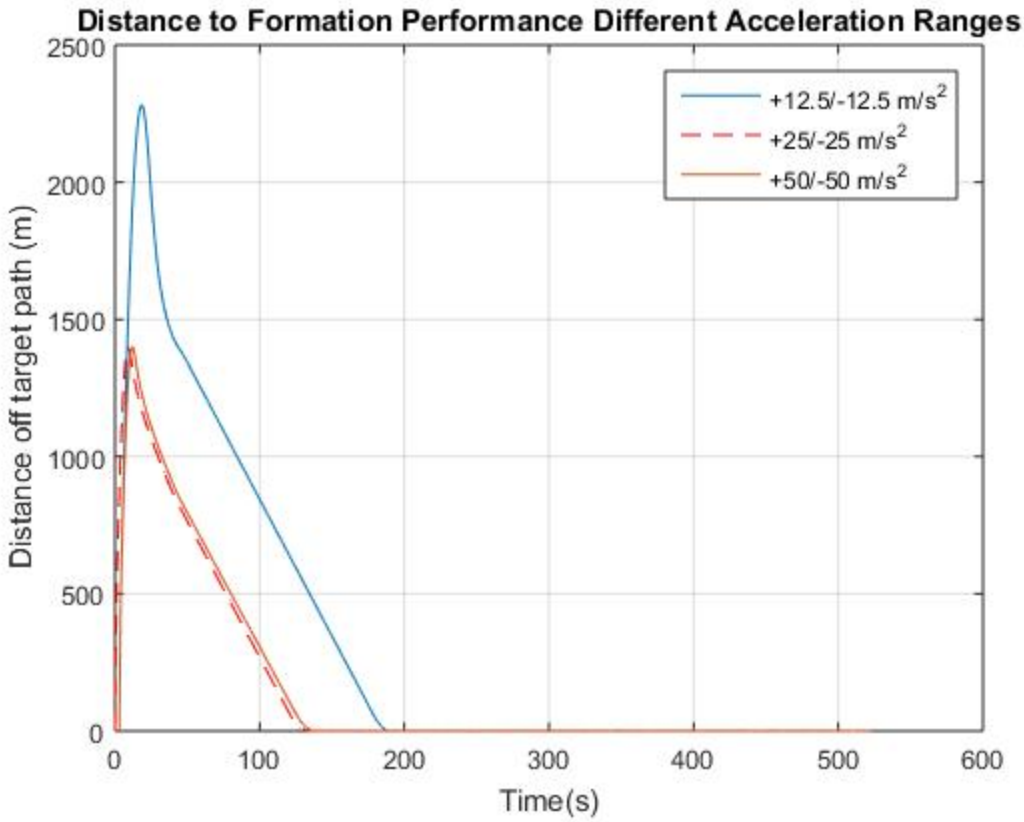

Figure 37:Formation control and keeping performance for different acceleration ranges. Benchmark case is dotted line with initial separation of 1 m and maximum separation of 1.4 km

Figure 37 shows the deputy vehicle performance with different ranges of acceleration. The acceleration is the variable that directly controls the motion of the deputy vehicle. The range of acceleration corresponds to the magnitude limit in the acceleration for each dimension. The acceleration range's upper and lower limits are doubled and halved for this test to examine the performance results. The doubled range makes the target acquisition phase begin with a separation distance of 1.4 km and takes approximately 118 seconds to obtain the formation. The average and standard deviation of the doubled range are 0.85 m and 1.23 m respectively. The halved acceleration range has the opposite effect. The separation distance is over 2280 m and the vehicle takes longer to enter the formation-keeping phase at 165. The acceleration range is crucial for having sufficient formation control and keeping. The average and standard deviation

of the halved acceleration range are 0.32 m and 1.63 m respectively. The accuracy of the formation keeping phase is better during the halved range because the vehicle does not apply massive accelerations to move between points. This allows for a smoother motion profile of the deputy vehicle during simulations. This trade off must be taken into consideration when specifying the acceleration range of the deputy vehicle. The chief's maximum and minimum accelerations used during this scenario are 9.85 m/s^2 and 0.4 m/s^2. If the acceleration range does not cover the accelerations the chief trajectory undergoes, the deputy vehicle will struggle to regain formation. The acceleration ranges that cover maximum and minimum acceleration taken by the chief trajectory allow the deputy to keep up and stay in formation with the chief trajectory or vehicle.

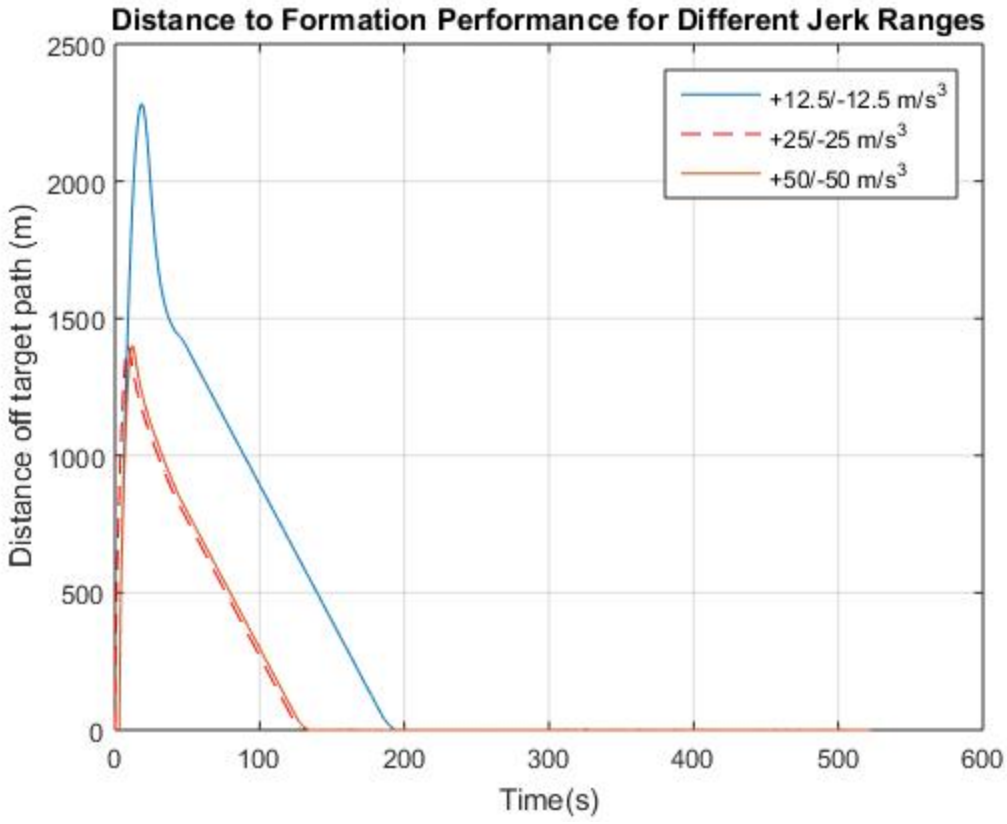

Figure 38: Formation control and keeping for different jerk ranges. Benchmark case is dotted line with initial separation of 1 m and maximum separation of 1.4 km

Figure 38 shows the effects of changing the jerk range the vehicle is allowed to use during a simulation. The jerk determines the change of acceleration a vehicle is permitted to obtain. The upper and lower values of jerk ranges of the original parameters were doubled and halved to display the effects of vehicle formation control and keeping performance. The jerk values higher and lower than the acceleration limits have of course little effect on the performance. The chief's maximum jerk is 4 m/s^3 and minimum jerk is 0 m/s^3. The original and double jerk ranges naturally have the same affect since the maximum acceleration is around 10 m/s^2. The ranges of the original and doubled ranges cover more than the acceleration range provides, however, the halved range does not. The target acquisition phase of the halved jerk range begins with a separation distance of 2280 m and takes 170 seconds to obtain the formation. The average and standard deviation for the formation-keeping phase are 0.52 m and 1.92 m, respectively.

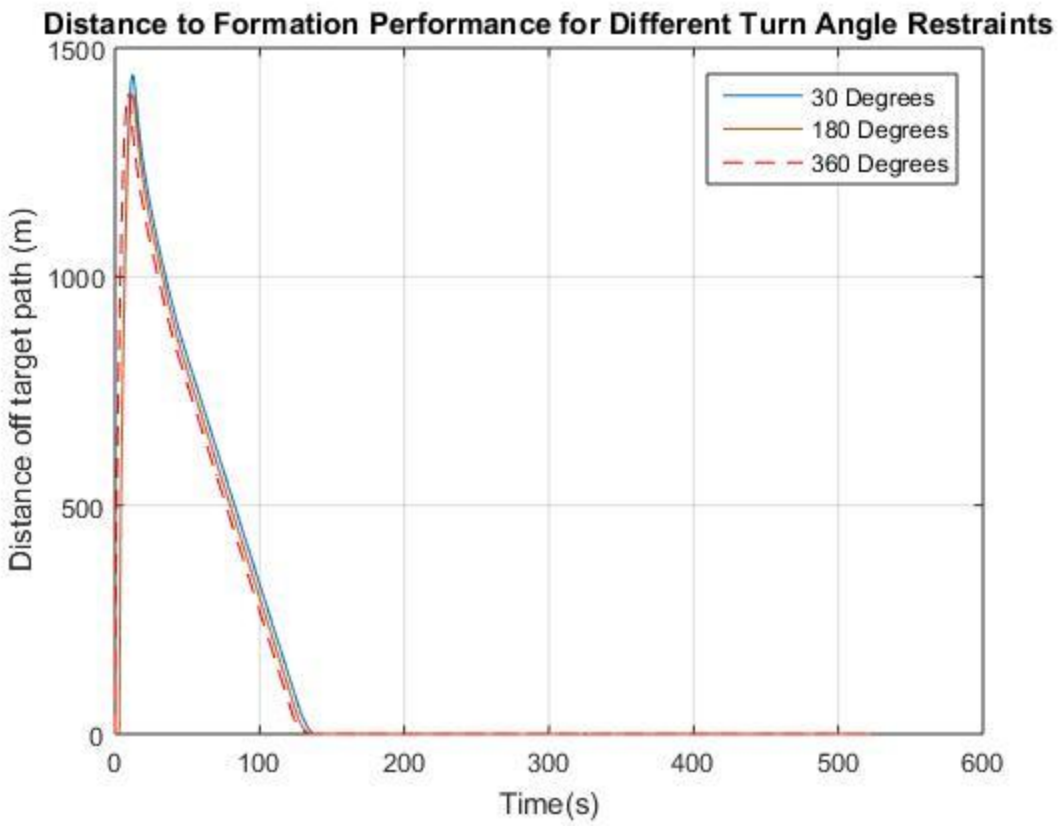

Figure 39: Formation control and keeping performance for turn angle restraints. Benchmark case is dotted line with initial separation of 1 m and maximum separation of 1.4 km

Figure 39 shows the turn angle restraints' effect on performance of a deputy vehicle. The turn angle restraints refer to the directions the acceleration can be applied relative to the vehicle's heading. If a vehicle heading straight has a turn angle restraint of 180 degrees, acceleration can be applied directly to the sides and from behind the vehicle. Accelerations cannot be applied in front of the vehicle to slow it down. In high dynamic scenarios, the turn angle restraint has a significant effect if the chief path shows high agility. The deputy vehicle will not be allowed to move freely and would only be allowed to move in certain directions during travel. The separation distance achieved by the 30 degree turn restraint and 180 degree turn restraint are 1442 m and 1.4 km respectively. The smaller turn angle affects the deputy vehicle's performance more when the target position is not in the window of view defined by the turn angle restraint.

The performance of the restraints in the acquisition and keeping phase are nearly identical. The average offset and standard deviation between the two differs by 0.02 m in average and 0.2 m in standard deviation. The times of the acquisition phases differ by three seconds.

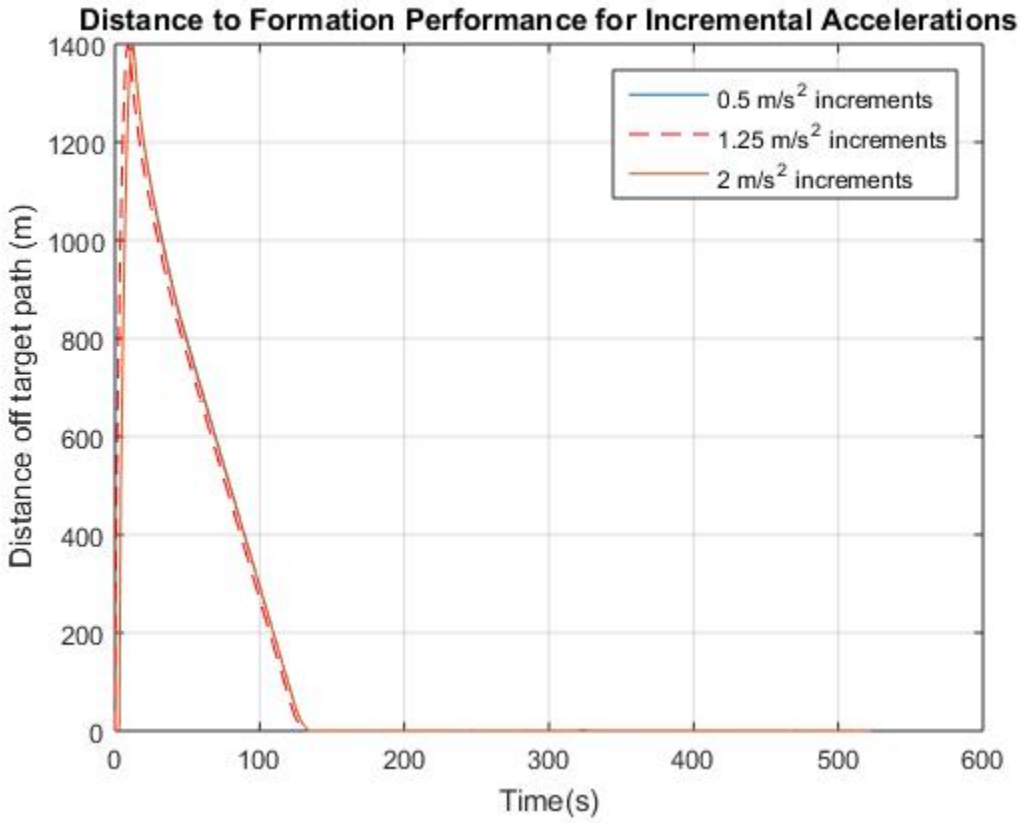

Figure 40: Formation control and keeping performance for incremental accelerations. Benchmark case is dotted line with initial separation of 1 m and maximum separation of 1.4 km

Figure 40 shows the incremental acceleration's effect on the performance of a deputy vehicle. The incremental acceleration parameter refers to the precision of acceleration values a deputy vehicle uses for its range of accelerations. Small increments have higher control of the formation in comparison to large increments. This is seen within the subtle vehicle response differences on the incremental acceleration plot. The average and standard deviation for the smaller increment parameter is 0.27 m and 1.57 m, respectively for the formation-keeping phase. The average and standard deviation for the larger increment parameter is 0.65 m and 1.52 m

respectively for the formation-keeping phase. Finer precision within the range of acceleration choices allows the vehicle to keep the formation better. There is a drawback to increasing the precision. If the precision increases, this amplifies the total number of acceleration choices in three dimensions the vehicle can choose from. Ultimately, this slows down the processing time of the navigation section. The processing time does not depend on the upper and lower limits of the range, but it depends on the possible values within the range the vehicle can choose from. *Table 4* shows the differences in processing time for different sizes of acceleration.

Table 4: Processing times for increment choices

Evenly Space Increments for Each Dimension	Matrix Size of Acceleration Possibilities	Average Processing Time on First Iteration (s)	Average Processing Time on Successive Iterations (s)
10	1000	0.002	0.001
50	125000	0.13	0.07
100	1000000	0.58	0.58
150	3375000	1.96	1.92

5.2.5 System Component Testing Summary

The purpose of component testing is to establish the performance and capability of the testbed. Timing and error quantities have shown the system's performance, and specific methods to use certain sections resulted.

Table 5: Error for receiver and raw measurement section

Section	Position Error (m)	Velocity Error (m/s)
Remote Control Tolerance	0.2	0.1
Raw Measurement (RM)	3.48E-04	0.53
Stationary Receiver +RM	1.80E-01	0.54
Low Dynamics Receiver +RM	3.03E-02	4.94
Medium Dynamics Receiver +RM	1.20E-01	6.52
High Dynamics Receiver +RM	1.47E+00	520.53

Table 5 shows that the receiver and raw measurement subsystem have errors in either position or velocity above the remote control system tolerances when coupled together. These errors prevent

the receiver being used on a deputy vehicle from producing motion for itself. The receiver can only be used for synchronized time or to measure a chief vehicle to provide waypoints for a deputy vehicle.

The processing time is also another critical system constraint to measure. Depending on the frequency of injected motion, the processing time of the entire test bed needs to be less than the time step interval. For example, if 41 evenly spaced acceleration increments are used to meet processing time intervals of 1 second, the processing time can be less than the time interval as shown in *Table 6*.

Table 6: Processing time example

Section	First Iteration Processing Times (s)	Successive Iteration Processing Times (s)
Raw Measurement	0.02	0.015
Navigation (41 Increments)	0.08	0.031
Total	0.1	0.046

This table shows that the processing time of the system can be calibrated to meet the time step parameter and cause no testbed loop breaks. The processing time of the testbed depends heavily on the acceleration range's precision.

The other initial parameters allow the user to create any deputy vehicle to follow a chief trajectory. Advanced scenarios are demonstrated in the next section.

5.3 Advanced Vehicle Scenarios

This section demonstrates the versatility of the testbed and the wide range of simulations that can be created. The HWIL testbed is modifiable and can be adjusted for any simulation.

5.3.1 Low Dynamics Scenario: Car Travel at Variable Speeds with Waypoints

The first advanced scenario is a car traveling from Blacksburg, VA to Chesapeake Bay between D.C. and Maryland. The car roughly follows the highway with speeds varying from 35 mph to 80 mph. The total time of travel is two hours.

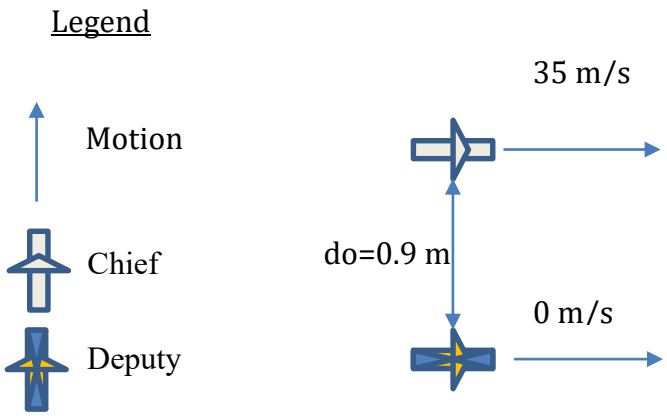

Figure 41:Initial Setup for Low Dynamic Scenario

This path travels at higher speed limits than the real path to save computational time. In this test, the car is given waypoints to reach during its travel and raw data is collected for the vehicle. The G-12 receiver is used for synchronized time and observed satellites only.

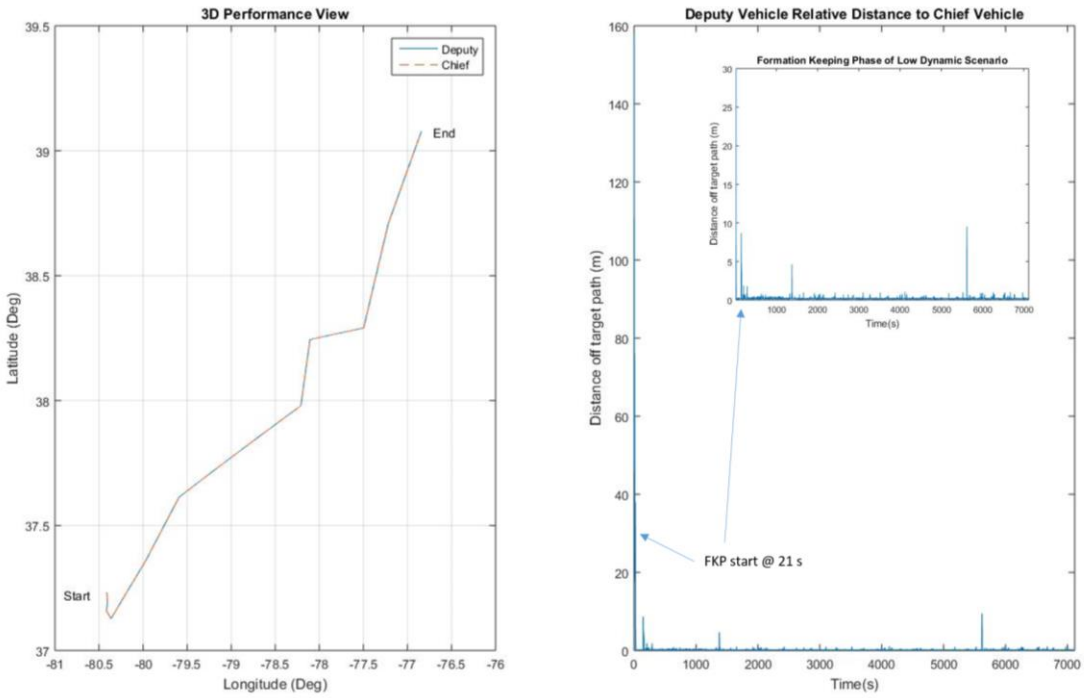

Figure 42: Performance and path of car following waypoints

Figure 42 shows the performance of the car following a path to Chesapeake Bay. The deputy vehicle parameters are shown in *Table 7*.

Table 7: Deputy initial parameters for low dynamic simulation

Time Step	Maximum Velocity	Zone 1 Cutback	Zone 2 Cutback	Maximum Acceleration	Minimum Acceleration	Max. /Min. Jerk	Turn Angle	ΔA
1 s	85 m/s	<50m, 82.45 m/s	<20m, 80.75 m/s	20 m/s^2	-10 m/s^2	+20/-20 m/s^3	360°	0.25 m/s^2

The chief path started at a velocity of 35 mph while the deputy vehicle started with no velocity. This velocity difference generated a 157 m separation distance. The formation acquisition phase lasted for 21 seconds due to the vehicle being allowed to have any velocity up to 85 m/s. Speed limit controls for different areas can be added to the testbed but is not currently in the system. The formation-keeping phase for this scenario has an average of 0.21 m offsets and a standard deviation of 0.44 m. These results prove that the deputy vehicle can follow the chief path even under changing velocity during travel. There are some larger offsets within the

formation-keeping phase around 10 m that occur during sharp or sudden turns. However, the deputy vehicle quickly recovers and stays within the formation.

The raw data displayed in *Figure 43*, *Figure 44*, and *Figure 45* demonstrate the raw data measured from the receiver during testing. All data is accurate within the tolerances described in section 5.2.3. This information is presented to show the capability of measuring raw data during testing.

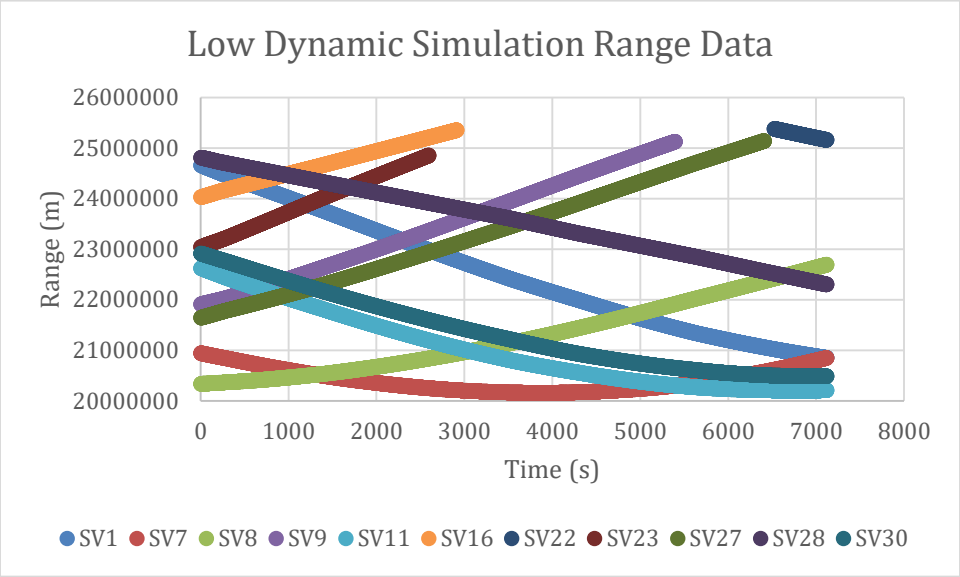

Figure 43: Low dynamic simulation range data

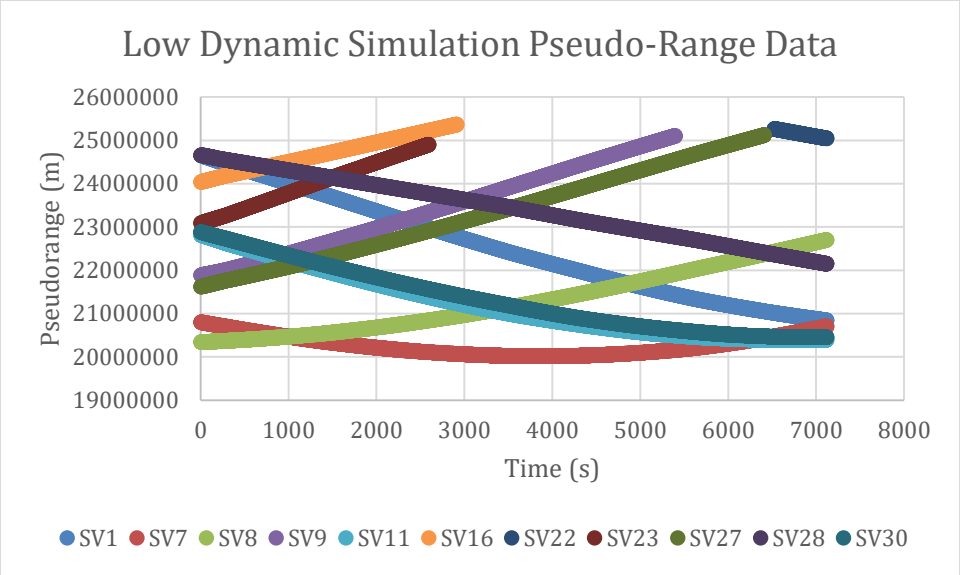

Figure 44: Low dynamic simulation pseudo-range data

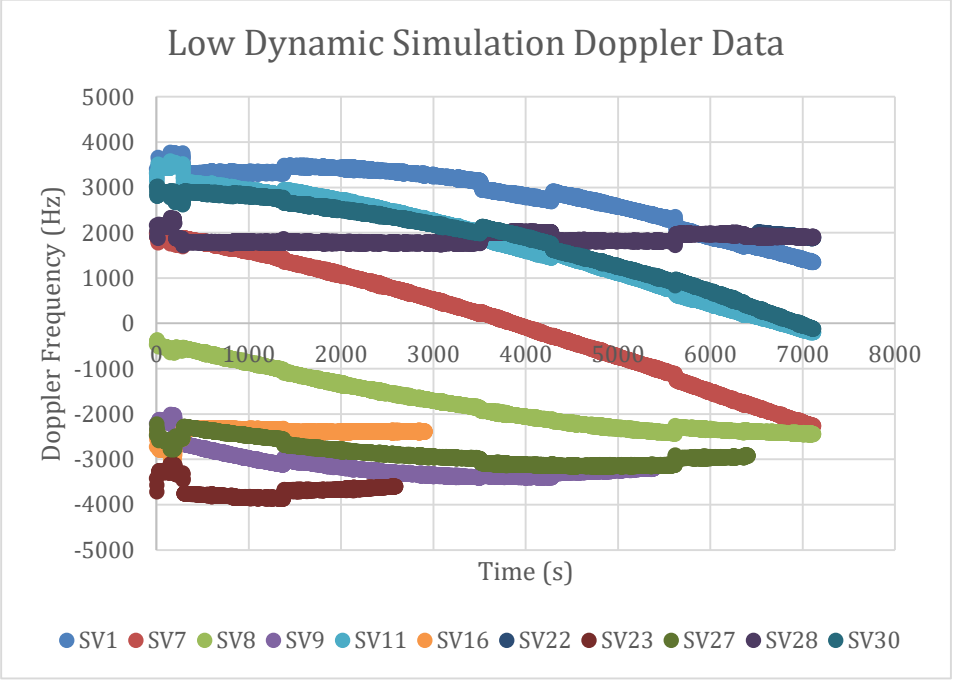

Figure 45: Low dynamic simulation Doppler data

This scenario has been created to demonstrate the autonomous vehicle applications that the new generation automobiles are investigating. Simulations for these applications can easily

be tested through this testbed before being built and applied. Testing the scenarios on the testbed will save time and money.

5.3.2 Medium Dynamics Scenario: Airplane in Formation with Chief Airplane travelling at Constant Speeds

The second advanced scenario is of a deputy airplane following live measurement position data from a chief vehicle. The chief vehicle is travelling at a constant 500 m/s and endures two turns of 0.1 g of lateral acceleration. The duration of the simulation lasted for two hours. In this test, a deputy airplane's performance is tested under live position data from a receiver that has some error. The deputy vehicle is positioned 2772 m above the starting position of the chief vehicle with a velocity direction perpendicular to the chief. The initial setup of the chief and deputy are shown in Figure 46.

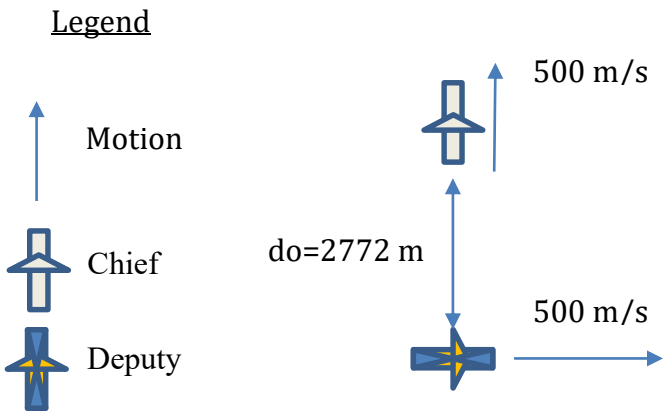

Figure 46: Initial setup for vehicles in medium dynamic scenario

The deputy vehicle follows the chief vehicle at a constant 100 m separation distance during the first hour. The separation distance grows at a rate of 0.1 m/s in altitude. Final separation distance at the end of the second hour is approximately 480 m. The trajectories between the chief and deputy vehicles are shown in Figure 47.

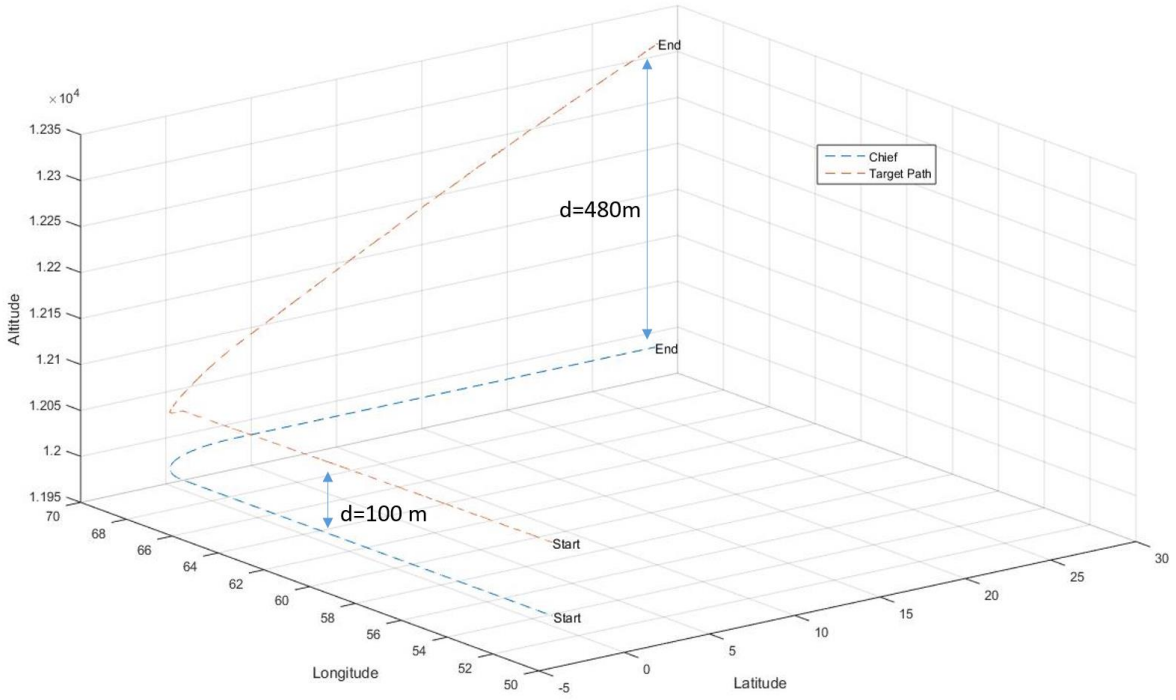

Figure 47: Chief and deputy vehicle target path

Future positions of the chief are estimated from velocity dynamic information provided by the G-12 receiver. The receiver is used for synchronized time, location, velocity, and SVs observed. The initial parameters of the deputy plane are shown in *Table 8*.

Table 8: Medium dynamic simulation deputy vehicle initial parameters

Time Step	Maximum Velocity	Zone 1 Cutback	Zone 2 Cutback	Maximum Acceleration	Minimum Acceleration	Max. /Min. Jerk	Turn Angle	ΔA
1 s	510 m/s	<50m, 505.4 m/s	<20m, 501.3 m/s	35 m/s^2	-20 m/s^2	+35/-35 m/s^3	360°	1.38 m/s^2

Figure 48 shows the performance and path of the deputy vehicle following the chief path using the receiver measurements.

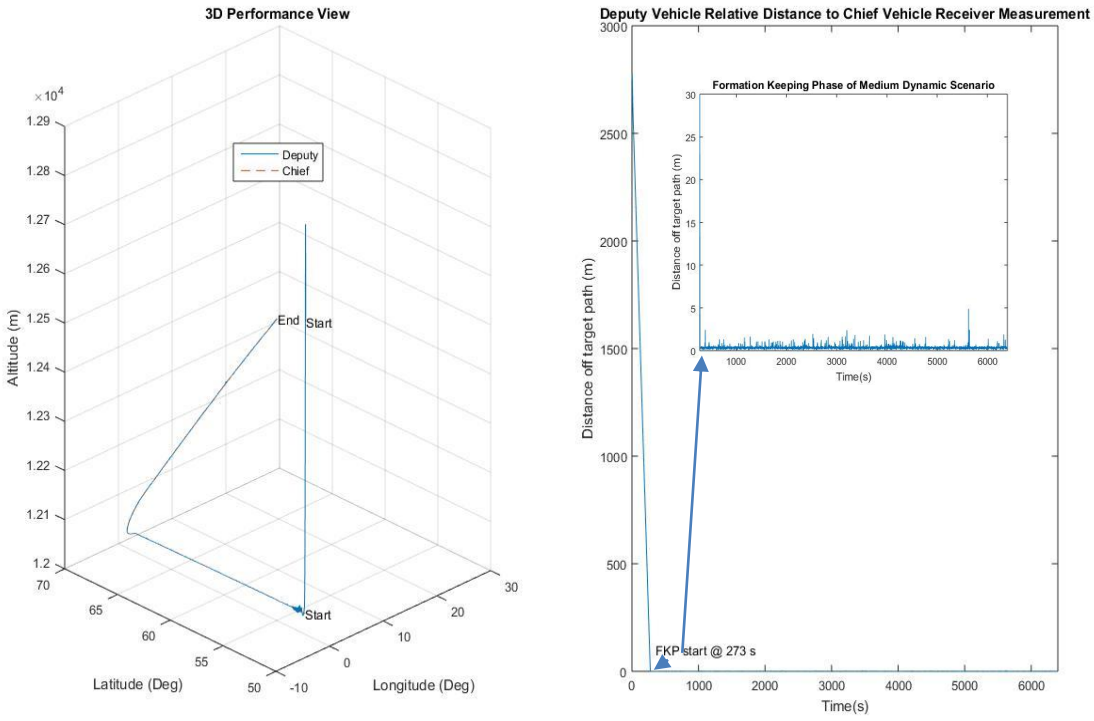

Figure 48: Performance and path of medium dynamic simulation. Initial separation distance 800 m.

The performance graph indicates how well the deputy vehicle follows the motion outlined by the receiver. The deputy vehicle quickly corrects its trajectory with two turns within 30 seconds. The maximum separation distance is approximately 2772 km during the vehicle correction phase. The formation acquisition phase lasts for 273 seconds due to a large maximum acceleration and maximum velocity being 10 m/s higher than the chief vehicle. The graph on the left shows the deputy following the receiver measurements of the chief position and velocity. The keeping phase has an average of 0.35 m offsets with a standard deviation of 0.36 m.

There are higher errors in this simulation compared to the low dynamic simulation due to the deputy vehicle following an unstable path established by receiver measurements. The initial parameters can be altered to reduce the offsets lower. This testbed simulation demonstrates the precision of a medium dynamic scenario involving an airplane using a live chief motion

trajectory. This simulation also validates the effectiveness of testing drones or other autonomous vehicle applications in the real world.

5.3.3 High Dynamics Scenario 1: Two Satellites in Formation with Chief LEO Satellite

The third advanced scenario is of two satellites flying in formation with a chief satellite. The chief satellite orbits earth at altitudes between 200 km and 2000 km similar to a LEO satellite. The chief satellite moves at speeds between 5.9 km/s and 7.8 km/s. The total time of travel is 12 hours. In this test, the deputy satellites are monitored to see if they can maintain and keep formation under high dynamics. The formation of the chief is known and the deputy vehicle will follow the chief trajectory. The initial setup of the vehicles in this scenario is shown in Figure 49.

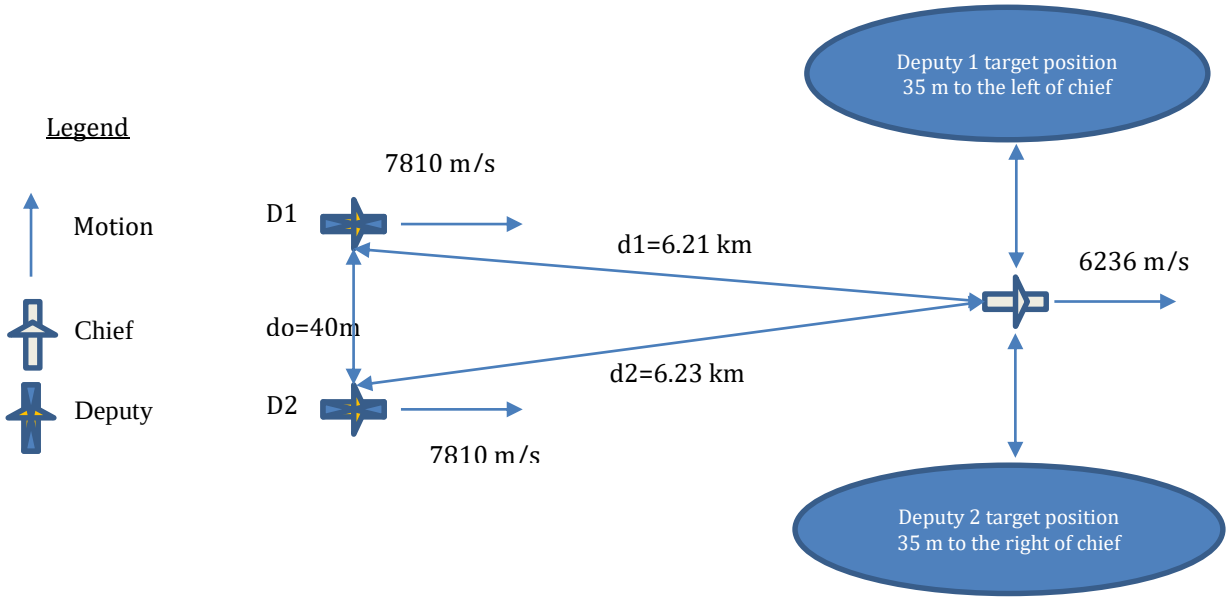

Figure 49: Initial setup of vehicles for high dynamics scenario 1

The receiver is only used for synchronized time for the navigation section to apply motion commands at the current time and the following time step. The initial parameters of the deputy vehicles are shown in *Table 9*.

Table 9: Deputy satellite initial parameters for high dynamic scenario 1

Time Step	Maximum Velocity	Zone 1 Cutback	Zone 2 Cutback	Maximum Acceleration	Minimum Acceleration	Max. /Min. Jerk	Turn Angle	ΔA
1 s	7810 m/s	<50m, 7807 m/s	<20m, 7802 m/s	10 m/s^2	-10 m/s^2	+10/-10 m/s^3	360°	0.5 m/s^2

The acceleration of a typical LEO satellite's thruster is 0.002 m/s^3. However, this does not account for the gravitational acceleration and other solar acceleration that are placed on a space vehicle. The acceleration for the deputy vehicles is set to 10 m/s^2 to account for these accelerations. Deputy vehicle 1 follows the formation at 20 meters in each dimension away from the chief position in ECEF. Deputy vehicle 2 follows the formation a -20 meters in each dimension away from the chief dimension in ECEF. Both vehicles are started at 40 m apart from each other. The performance and path of the satellite formation can be viewed in *Figure 50*.

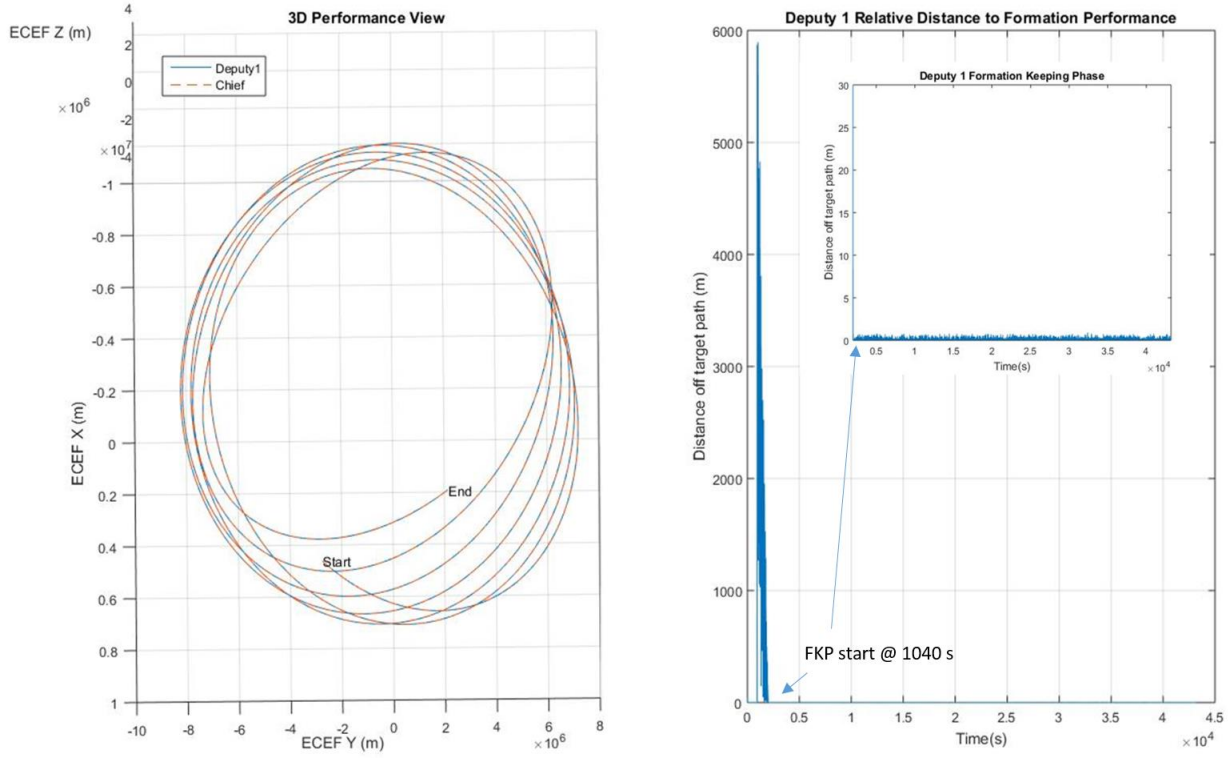

Figure 50: Performance and path of deputy satellite 1 formation for high dynamic scenario 1

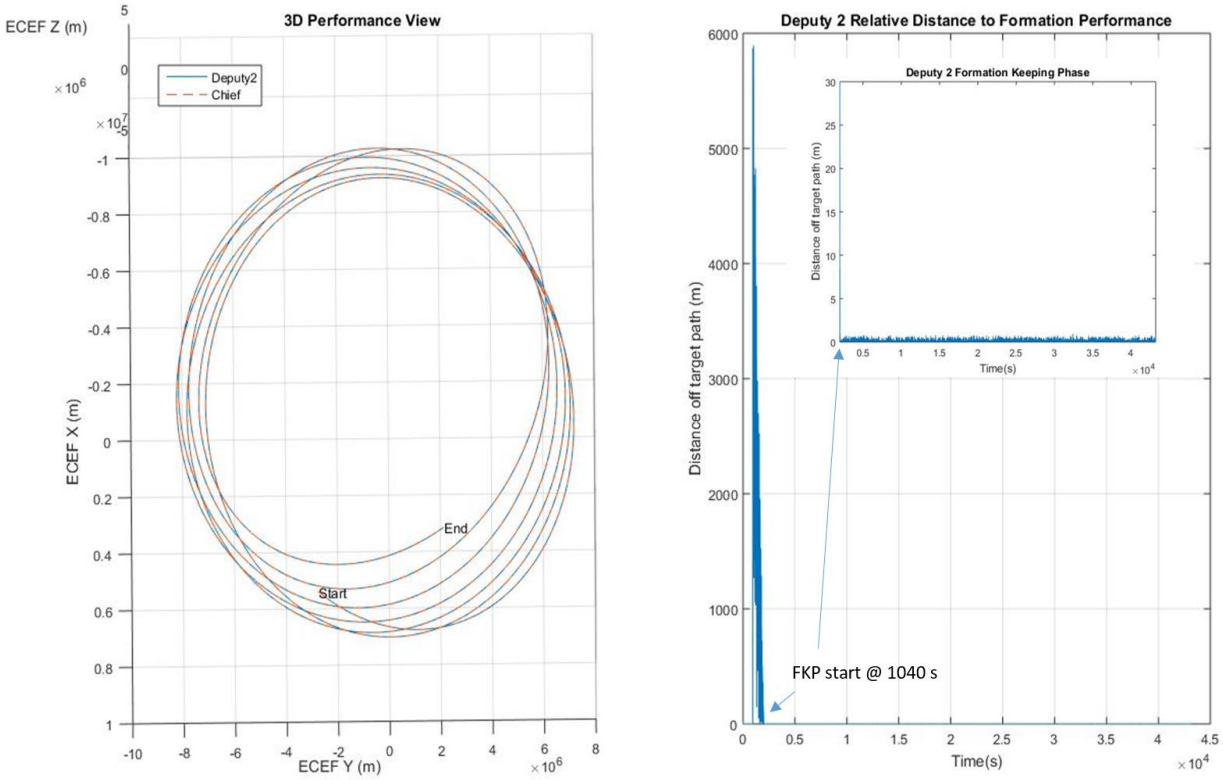

Figure 51: Performance and path of deputy satellite 2 formation for high dynamic scenario 1

The graph of the path shows that the formation is well kept during travel. The deputy vehicles endure a separation distance of approximately 6 km before the formation acquisition phase begins. The acquisition phase lasts for 1040 seconds for each vehicle. Since the vehicles have the same initial parameters and their respective target paths are moving identically, their performance is the same. The average offset in the formation-keeping phase is 0.13 m and the standard deviation is 0.11 m. Lower offset occurs in the satellite scenario because the satellite travels in a direction without sudden or sharp turns.

This scenario proves the testbed is capable of handling vehicles under high dynamics. The applications of these tests can be expanded to satellites that provide data collection or rendezvous missions for restocking spacecraft or space stations.

5.3.4 High Dynamics Scenario 2: Two Satellites in a Leader-Follower Formation with Chief Vehicle

The fourth advanced scenario is of two satellites flying in a leader-follower formation with a chief satellite. The chief satellite orbits earth at altitudes between 200 km and 2000 km similar to a LEO satellite. The chief satellite moves at speeds between 5.9 km/s and 7.8 km/s. The total time of travel is approximately four hours. Deputy vehicle 1 follows the chief trajectory 30 seconds behind the chief and deputy vehicle 2 follows the trajectory 60 seconds behind the chief. In this test, the deputy satellites are monitored to see if they can maintain and keep formation under high dynamics. A diagram of the vehicles' initial setup is displayed in Figure 52.

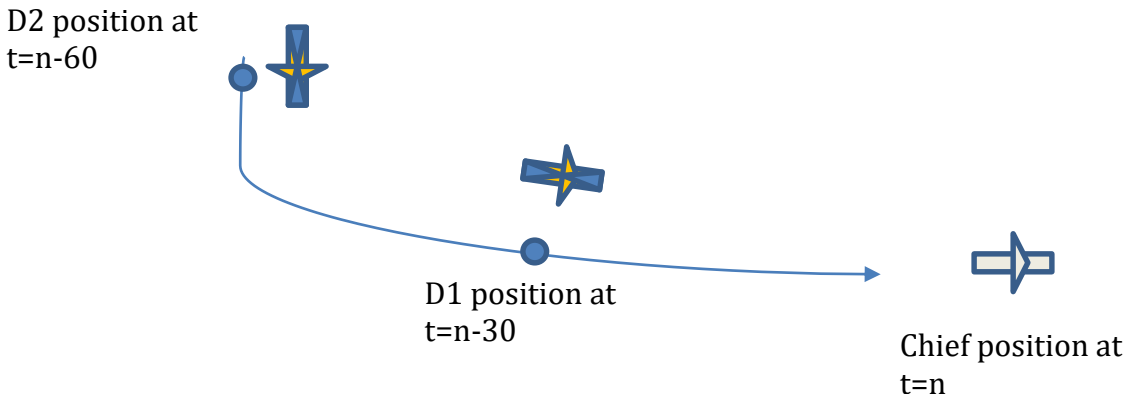

Figure 52: Initial Setup for vehicles in high dynamic scenario 2

The orbit of the chief is known and the deputy vehicles will follow the chief trajectory. The receiver is only used for synchronized time to apply motion using the current time and following time step. The initial parameters of the deputy vehicles are shown in *Table 10*.

Table 10: Deputy satellite initial parameters for high dynamic scenario 2

Time Step	Maximum Velocity	Zone 1 Cutback	Zone 2 Cutback	Maximum Acceleration	Minimum Acceleration	Max. /Min. Jerk	Turn Angle	ΔA
1 s	7815 m/s	<50m, 7810 m/s	<20m, 7805 m/s	15 m/s^2	-15 m/s^2	+15/-15 m/s^3	360°	0.75 m/s^2

The performance and path of the deputy vehicles are shown in *Figure 53* and Figure 54.

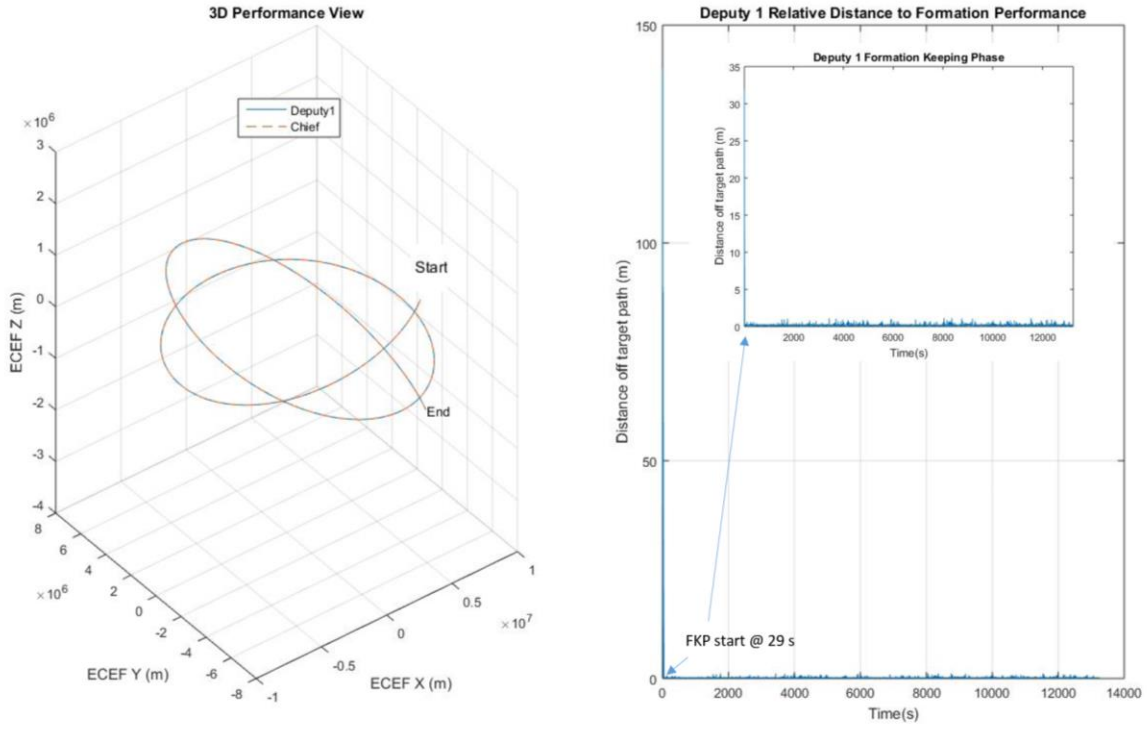

Figure 53: Performance and path of deputy vehicle 1 in high dynamic scenario 2

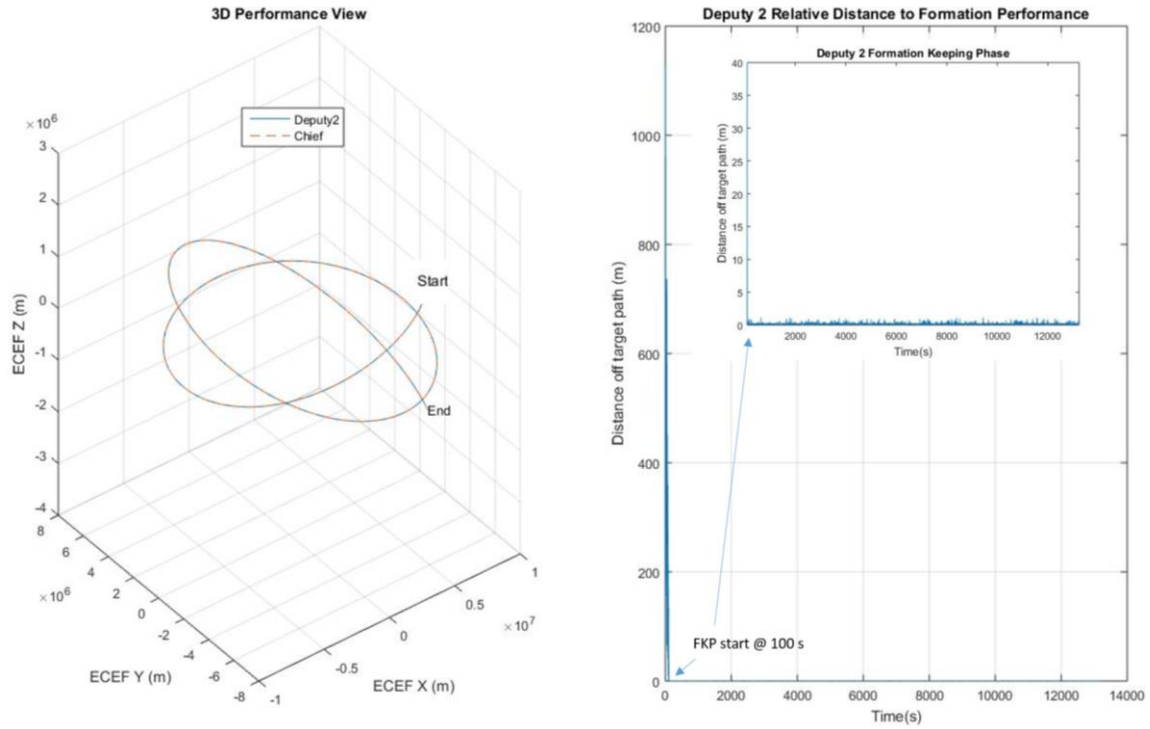

Figure 54: Performance and path of deputy vehicle 2 in high dynamic scenario 2

The path shown on the left for both vehicles shows almost no offset during the simulation. The satellites are positioned relatively close to their starting points that they need to obtain along the chief path. Deputy vehicle 1 starts with a separation distance of 149 m and deputy vehicle 2 starts with a separation distance of 1.1 km. The formation acquisition phase of deputy vehicle 1 lasts for 29 seconds and deputy vehicle 2's acquisition phase lasts for 1 minute and 40 seconds. The average offset in formation keeping phase of deputy vehicle 1 and deputy vehicle 2 are 0.2 m and 1.9 m, respectively. The standard deviation of the offset for deputy vehicle 1 and deputy vehicle 2 are 0.36 and 0.2 m, respectively.

This simulation shows the different applications that can be applied to the testbed. The application is analogous to satellites taking swath measurements of the geography of Earth for mapping or reconnaissance. The testbed can be adapted to fit any scenario to simulate vehicle performance.

5.3.5 High Dynamics Scenario 3: Two Satellites in a Leader-Follower Formation with Chief Vehicle at Significant Separation Distance

The fifth advanced scenario simulates two satellites flying in a leader-follower formation with a chief satellite that starts with a significant separation distance from the deputy vehicles. The chief satellite motion follows a circular equatorial orbit at a constant altitude of 1003 km. The chief travels between speeds of 6329 m/s and 6309 m/s. The total time of travel is approximately three hours. Deputy vehicle 1 follows the chief trajectory 30 seconds behind the chief and Deputy vehicle 2 follows the trajectory 60 seconds behind the chief. In this test, the deputy satellites are monitored to see if they can acquire, maintain, and keep formation under high dynamics with a significant separation distance. The initial setup is identical to the previous high dynamic scenario with a larger separation distance between each vehicle. The initial separation of deputy vehicle 1 is 143000 m and 6810 m for deputy vehicle 2. The orbit of the chief is known and the deputy vehicles will follow the chief trajectory. The receiver is only used for synchronized time to apply motion using the current time and following time step. The initial parameters of the deputy vehicles are shown Table 11.

Table 11:Initial parameters for deputy vehicles for high dynamic scenario 3

Vehicle	Time Step	Maximum Velocity	Zone 1 Cutback	Zone 2 Cutback	Maximum Acceleration	Minimum Acceleration	Max. /Min. Jerk	Turn Angle	ΔA
Deputy 1	1 s	6835 m/s	<50m, 6831 m/s	<20m, 6829 m/s	10 m/s^2	-15 m/s^2	+15/-15 m/s^3	360°	0.31 m/s^2
Deputy 2	1 s	6835 m/s	<50m, 6831 m/s	<20m, 6829 m/s	10 m/s^2	-10 m/s^2	+10/-10 m/s^3	360°	0.25 m/s^2

The performance and path for both vehicles are shown in Figure 55.

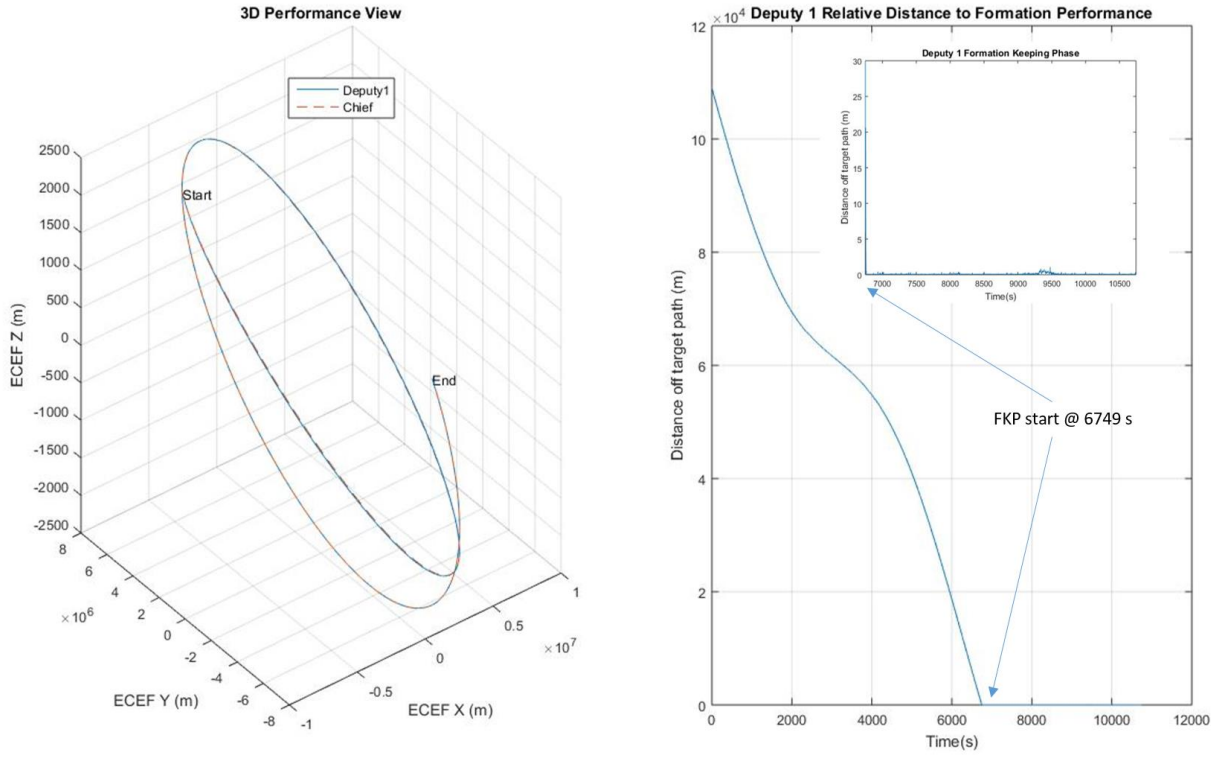

Figure 55: Performance and path of deputy vehicle 1 in high dynamic scenario 3. Initial separation distance

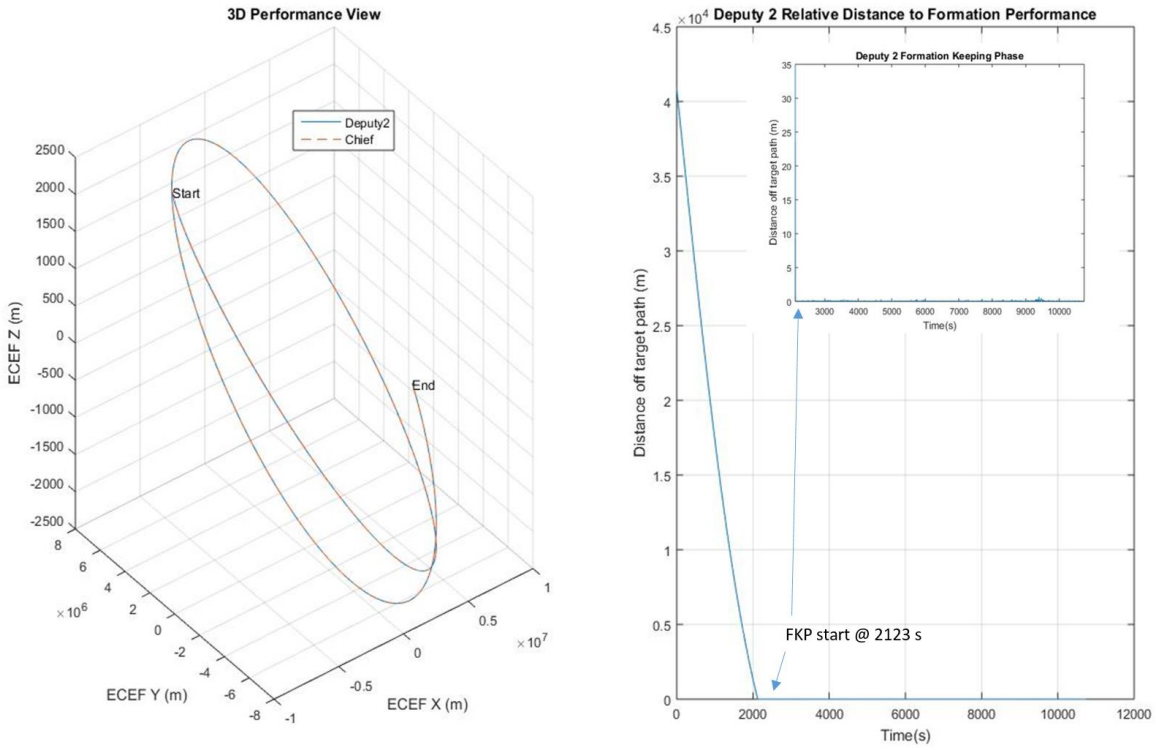

Figure 56: Formation and performance of deputy vehicle 2 in high dynamic scenario 2

Deputy 1 starts with an initial separation distance over 100 km. Deputy 2 starts with an initial separation distance over 40 km. This performance shows that even at significant separation distances, the testbed has the capability of achieve desired formations and trajectories. Deputy vehicle 1's acquisition phase lasts almost two hours before the formation is kept. Deputy vehicle 2's acquisition phase lasts for approximately 35 minutes before its keeping phase begins. The average offset and standard deviation of the keeping phase for deputy vehicle 1 is 0.07 m and 0.37 m, respectively. The average offset and standard deviation for the keeping phase of vehicle 2 is 0.12 m and 0.73 m, respectively. The initial condition of the vehicles can be altered to enhance the performance.

The motion produced during all scenarios have discrepancy with following a realistic path. Although the separation distance between any vehicle and its trajectory path is driven to zero during simulations, the current dynamic model overcorrects for a vehicle's motion.

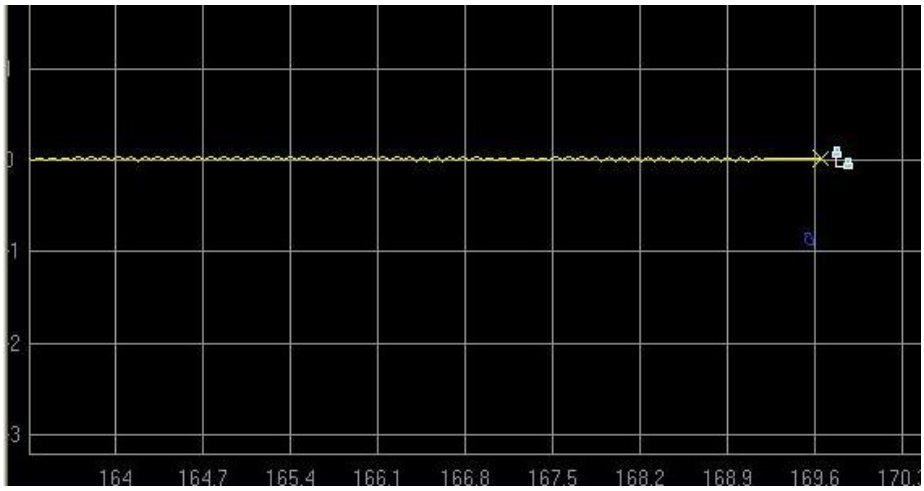

Figure 57: Vehicle 1 ground track during high dynamic scenario 3

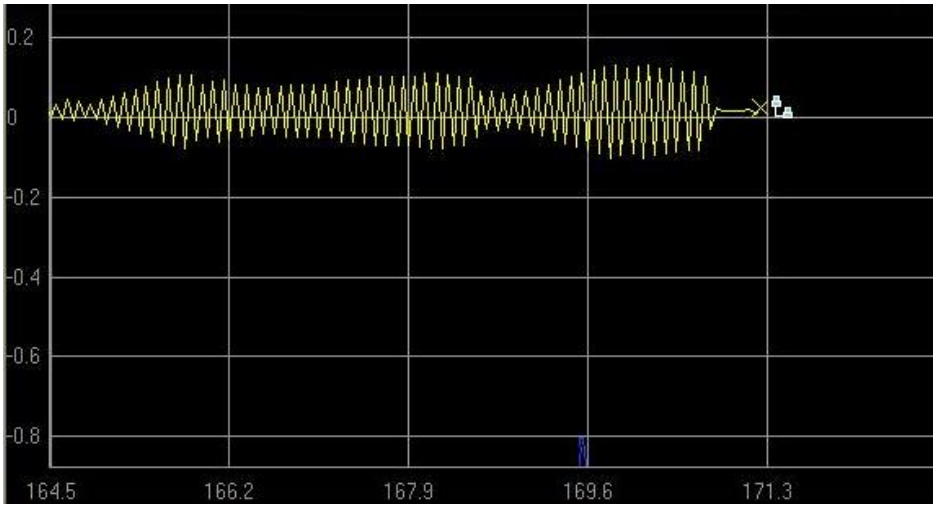

Figure 58: Vehicle 2 ground track during high dynamic scenario 3

Figure 57 and Figure 58 show the ground track for both deputy vehicles during simulation. The acceleration applied oscillates between time steps. This shows that the vehicle either does not quite reach its required waypoint or is overshooting the desired waypoint. In either case, the vehicle tries to recover and overcorrect its motion on the next step. This phenomenon is more

apparent in high dynamic scenarios due to the magnitude of the velocities being used. Although this behavior is occurring, formation acquisition and keeping phases of all vehicle scenarios have demonstrated that the goals of the testbed are met. The consequence of this behavior is that unnecessary motion is being applied which costs increased fuel consumption during an actual mission. The deceleration of deputy vehicle 1 was applied to try and reduce the oscillatory behavior, and it worked. The frequency and magnitude of oscillation for deputy vehicle 1 is less than vehicle two. The behavior also occurs less often over the entire simulation. A future upgrade should be directed to the navigation section to smooth out the applied motion, which is more realistic and less costly. A filtering algorithm may be able to improve the system performance.

Chapter 6: Conclusion and Future Recommendations

The HWIL testbed is capable of simulating a wide variety of vehicle motion or formation. Its versatility allows for modeling the initial parameters of vehicles and can be modified to incorporate other trajectory controls and features also. There is a plethora of ways that the chief motion can be sent to the deputy vehicle for simulations. Live data streams, pre-planned motion, and waypoints are a few of those ways demonstrated in this work. The deputy vehicles can be set to reduce the offset of the trajectory path to be as low as required. This HWIL testbed is a great way to determine the performance of a vehicle before it is built to meet the user's needs.

Currently the testbed is a basic model derived from basic GPS and physics principles for navigation. More features should be added to the model to incorporate more real world effects on vehicle motion and to fix the oscillation behavior in the applied motion. The G-12 and DG-14 receivers, in their current state, provide too much error for a deputy vehicle to rely on the receivers' measurements for its own location and velocity. Either using the receiver in differential mode or using a more accurate receiver should allow a receiver to be used to measure a deputy vehicle and allow it to be self-aware. The GNSS 8000 simulator causes a 15 second time slip to the receivers after ten to eleven minutes of usage. This time slip can be fixed with a MATLAB script but if the error goes unchecked it can break the loop. Lastly, the MATLAB serial communication commands occasionally times out. A better method for receiver communication should be used during simulations.

This testbed and its algorithm components differ from similar systems in quite a few ways. This testbed is one or more vehicle applications implemented in HWIL. It is a versatile testbed capable of simulating any vehicle or application. This navigation algorithm provides

initial parameters for any vehicle and can be modified to include more vehicle features or applications. Since the testbed is derived from basic principles, it can also be used as an educational tool for any novice to use to develop vehicle trajectory/formation simulation experiments.

The testbed created for this thesis serves as a base model for future expansion. It contains all the parts necessary to simulate formations and single vehicles within any precision. The ideal testbed would include smoother motion without oscillation during travel. Attempts to model jerk as inputs were unsuccessful due to its sporadic nature. Advancements in the testbed should include curvilinear motion, a motion estimation filter to eliminate the oscillation behavior, and real world constraints or boundaries. Real world boundaries or constraints, commonly known as restricted or no fly zones, ensure that vehicles do not travel beneath the Earth's surface or restrict to specific geographic locations. The ideal testbed would also use an accurate receiver. There is a Novatel OEM-628 receiver in the GPS lab at Virginia Tech that provides high accuracy but has altitude errors between 20-30 m. If the altitude error can be eliminated, the Novatel receiver can be used to monitor and promote self-contained movement of the deputy vehicle. The current testbed is also not configured for simulations longer than 24 hours. The date is not included within the 24-hour clock format for the time analysis but can be easily added. Obstacle avoidance needs to be a priority expansion to this system as well. The ideal testbed would include all of these updates as well as expansions to increase the models performance and capabilities.

References

[1] C. Q. Yang and Z. Wang, "Formation control: a review and a new consideration," in International Conference on Intelligent Robots and Systems, Edmonton, Canada, 2005.

[2] P. M. Kintner, Jr., Global Positioning System Theory and Design, Ithaca, New York, 1999.

[3] Q. Marji, "Precise Relative Navigation for Satellite Formation Flying Using GPS," Calgary, 2008.

[4] A. Snow, E. Buchen and J. R. Olds, "Trends in Average Earth-Orbiting Spacecraft Launch Mass," SpaceWorks Enterprises Inc., 10 6 2014. [Online]. Available: http://www.spaceworksforecast.com/docs/SpaceWorks_Spacecraft_Mass_Trends_2014.pdf. [Accessed 16 3 2016].

[5] "Mission Statement," Cubesat, 2016. [Online]. Available: www.cubesat.net/index.php/about-us/mission-statement. [Accessed 24 3 2016].

[6] S. Stansbury, "Low Thrust Transfer to GEO: Comparison of Electric and Chemical Propulsion," 10 12 2009. [Online]. Available: http://ccar.colorado.edu/asen5050/projects/projects_2009/stansbury/. [Accessed 11 02 2016].

[7] "The 10 Fastest Aircraft in the World," MIGFLUG, 26 11 2014. [Online]. Available: http://www.migflug.com/jetflights/the-10-fastest-aircraft-in-the-world.html. [Accessed 11 02 2016].

[8] S. Said, "The 10 Fastest Passenger Planes in History," The Richest, 17 05 2013. [Online]. Available: http://www.therichest.com/business/technology/the-10-fastest-passenger-planes-in-history/?view=all. [Accessed 11 02 2016].

[9] B. Hunting, "10 Fastest Acceleration Cars 0-60," Autobytel Inc., [Online]. Available: http://www.autobytel.com/sports-cars/car-buying-guides/10-fastest-acceleration-cars-0-60-120638/. [Accessed 11 2 2016].

[10] J. R. Guinn and R. J. Boain, "Spacecraft Autonomous Navigation for Formation Flying Earth orbiters Using GPS," 1996.

[11] J. Leiner, "A Hardware-in-the-Loop Testbed for Spacecraft Formation Flying Applications," Greenbelt, MD, 2001.

[12] N. E. Johnson, J. A. Calise, R. Sattigeri, Y. Watanabe and V. Madyastha, "Approaches to Vision-Based Formation Control," in IEEE Conference on Decision and Control, Atlantis, Paradise Island, Bahamas, 2004.

[13] R. Alonso, J.-Y. Du, D. Hughes, J. L. Junkins and J. L. Crassidis, "Relative Navigation For Formation Flying of Spacecraft," College Station, TX & Amherst, NY.

[14] D. Lebedev and A. Tkachenko, "Magnetic Attitude Control System for Low-Earth Orbit Satellites," Kluwer Academic Publishers, Kiev, Ukraine, 2001.

[15] J.-I. Park, H.-E. Park, S.-Y. Park and K.-H. Choi, "Hardware-in-the-loop simulations of GPS-based navigation and control for satellite formation flying," The Republic of Korea, 2010.

[16] "Equations of Motion," 11 03 2016. [Online]. Available: https://en.wikipedia.org/wiki/equations_of_motion. [Accessed 22 03 2016].

[17] "Spirent Communications Announces GSS8000 High Dynamics GPS Simulation System," Spirent Federal Systems Inc., 5 05 2008. [Online]. Available: http://www.spirent.com/About-Us/News_Room/Press-Releases/2008/5-5-08. [Accessed 8 4 2016].

[18] "GSS8000 Series," Spirent Federal Systems Inc., 3 3 2013. [Online]. Available:

http://www.spirentfederal.com/GPS/documents/GSS8000_datasheet.pdf. [Accessed 8 4 2016].

[19] "Multi-Channel Fully Flexible GPS/SBAS Simulation System Spirent GSS6560," Spirent Federal Systems Inc., 5 2006. [Online]. Available: http://nubilight.nubicom.co.kr/upload/datasheet/GSS6560%20DS.pdf. [Accessed 8 4 2016].

[20] A. Inc., G12 GPS Board Reference Manual, Sunnyvale, CA, 1997.

[21] NovAtel, "Receivers OEM628," 03 2016. [Online]. Available: http://www.novatel.com/assets/Documents/Papers/OEM628.pdf. [Accessed 8 4 2016].

Appendix

Appendix A: Raw Measurement Section Receiver Code

Script 1: Constants

```
%Uploaded GPS satellite positions during scenario
SATposition= xlsread('Satellite Position file');
%Upload ephemeris file for day of scenario
Eph= xlsread('Ephemeris file');
SATpositionsize=size(SATposition);

%Constants for converting UTC time to GPS time of week
Sunday=0; Monday=1; Tuesday=2; Wednesday=3; Thursday=4; Friday=5; Saturday=6;
% Select day of scenario
Day=Thursday;

%GPS Constants
%'muk' : G*Me, the "gravitational constant" for orbital motion
% about the Earth
 muk = 398600.5e9;      % meters^3/second^2
%'AA' : the semi-major axis of the reference ellipsoid (WGS-84)
 AA = 6378137.00000;     % meters
%'BB' : the semi-minor axis of the reference ellipsoid (WGS-84)
 BB = 6356752.31425;     % meters
%'esquare' : the square of the Earth's orbital eccentricity
 esquare=(AA^2 - BB^2) / AA^2;
%'OmegaE' : the sidereal rotation rate of the Earth (WGS-84)
 OmegaE = 7.292115e-5;    % radians/second
%'c' : the speed of light (meters/second)
 c = 299792458;       % meters/second
%'degrad' : a constant used for converting degrees to radians
degrad = pi/180.0;
%'leapSeconds': the number of leap seconds currently for the GPS system
leapSeconds = 16;      % seconds
%'f0' : the fundamental frequency for the GPS system
 f0 = 10.23e6;       % Hertz
%'f' : the L1 carrier frequency
 f = 154 * f0;       % Hertz
%lambda' : the L1 carrier wave length
 lambda = c / f;      % meters

 %Direction associated with WGS-84 component directions
 S=-1; N=1; W=-1; E=1;

%UTC time of satellites extracted from excel file (Not Needed if file
%contains UTC time importable by Matlab)
Exceltime=datestr(SATposition(:,2),'HH:MM:SS');
```

```
Hourdigit1=Exceltime(:,1); Hourdigit2=Exceltime(:,2); Minutedigit1=Exceltime(:,4);
Minutedigit2=Exceltime(:,5); Seconddigit1=Exceltime(:,7); Seconddigit2=Exceltime(:,8);
UTCtime=str2num(strcat(Hourdigit1,Hourdigit2,Minutedigit1,Minutedigit2,...
    Seconddigit1,Seconddigit2));

%Sorted ECEF (X,Y,& Z) positions of GPS satellites
SVID1=[SATposition(:,3),SATposition(:,35),SATposition(:,67)];
SVID2=[SATposition(:,4),SATposition(:,36),SATposition(:,68)];
SVID3=[SATposition(:,5),SATposition(:,37),SATposition(:,69)];
SVID4=[SATposition(:,6),SATposition(:,38),SATposition(:,70)];
SVID5=[SATposition(:,7),SATposition(:,39),SATposition(:,71)];
SVID6=[SATposition(:,8),SATposition(:,40),SATposition(:,72)];
SVID7=[SATposition(:,9),SATposition(:,41),SATposition(:,73)];
SVID8=[SATposition(:,10),SATposition(:,42),SATposition(:,74)];
SVID9=[SATposition(:,11),SATposition(:,43),SATposition(:,75)];
SVID10=[SATposition(:,12),SATposition(:,44),SATposition(:,76)];
SVID11=[SATposition(:,13),SATposition(:,45),SATposition(:,77)];
SVID12=[SATposition(:,14),SATposition(:,46),SATposition(:,78)];
SVID13=[SATposition(:,15),SATposition(:,47),SATposition(:,79)];
SVID14=[SATposition(:,16),SATposition(:,48),SATposition(:,80)];
SVID15=[SATposition(:,17),SATposition(:,49),SATposition(:,81)];
SVID16=[SATposition(:,18),SATposition(:,50),SATposition(:,82)];
SVID17=[SATposition(:,19),SATposition(:,51),SATposition(:,83)];
SVID18=[SATposition(:,20),SATposition(:,52),SATposition(:,84)];
SVID19=[SATposition(:,21),SATposition(:,53),SATposition(:,85)];
SVID20=[SATposition(:,22),SATposition(:,54),SATposition(:,86)];
SVID21=[SATposition(:,23),SATposition(:,55),SATposition(:,87)];
SVID22=[SATposition(:,24),SATposition(:,56),SATposition(:,88)];
SVID23=[SATposition(:,25),SATposition(:,57),SATposition(:,89)];
SVID24=[SATposition(:,26),SATposition(:,58),SATposition(:,90)];
SVID25=[SATposition(:,27),SATposition(:,59),SATposition(:,91)];
SVID26=[SATposition(:,28),SATposition(:,60),SATposition(:,92)];
SVID27=[SATposition(:,29),SATposition(:,61),SATposition(:,93)];
SVID28=[SATposition(:,30),SATposition(:,62),SATposition(:,94)];
SVID29=[SATposition(:,31),SATposition(:,63),SATposition(:,95)];
SVID30=[SATposition(:,32),SATposition(:,64),SATposition(:,96)];
SVID31=[SATposition(:,33),SATposition(:,65),SATposition(:,97)];
SVID32=[SATposition(:,34),SATposition(:,66),SATposition(:,98)];

%Sorted Eph constants from ephemeris file needed to calculate raw
%data of range, pseudo-range and doppler (toe, af0, af1, af2, tgd, toc)
ephSV1=[Eph(1,4),Eph(1,20),Eph(1,21),Eph(1,22),Eph(1,23),Eph(1,24)];
ephSV2=[Eph(2,4),Eph(2,20),Eph(2,21),Eph(2,22),Eph(2,23),Eph(2,24)];
ephSV3=[Eph(3,4),Eph(3,20),Eph(3,21),Eph(3,22),Eph(3,23),Eph(3,24)];
ephSV4=[Eph(4,4),Eph(4,20),Eph(4,21),Eph(4,22),Eph(4,23),Eph(4,24)];
ephSV5=[Eph(5,4),Eph(5,20),Eph(5,21),Eph(5,22),Eph(5,23),Eph(5,24)];
```

```
ephSV6=[Eph(6,4),Eph(6,20),Eph(6,21),Eph(6,22),Eph(6,23),Eph(6,24)];
ephSV7=[Eph(7,4),Eph(7,20),Eph(7,21),Eph(7,22),Eph(7,23),Eph(7,24)];
ephSV8=[Eph(8,4),Eph(8,20),Eph(8,21),Eph(8,22),Eph(8,23),Eph(8,24)];
ephSV9=[Eph(9,4),Eph(9,20),Eph(9,21),Eph(9,22),Eph(9,23),Eph(9,24)];
ephSV10=[Eph(10,4),Eph(10,20),Eph(10,21),Eph(10,22),Eph(10,23),Eph(10,24)];
ephSV11=[Eph(11,4),Eph(11,20),Eph(11,21),Eph(11,22),Eph(11,23),Eph(11,24)];
ephSV12=[Eph(12,4),Eph(12,20),Eph(12,21),Eph(12,22),Eph(12,23),Eph(12,24)];
ephSV13=[Eph(13,4),Eph(13,20),Eph(13,21),Eph(13,22),Eph(13,23),Eph(13,24)];
ephSV14=[Eph(14,4),Eph(14,20),Eph(14,21),Eph(14,22),Eph(14,23),Eph(14,24)];
ephSV15=[Eph(15,4),Eph(15,20),Eph(15,21),Eph(15,22),Eph(15,23),Eph(15,24)];
ephSV16=[Eph(16,4),Eph(16,20),Eph(16,21),Eph(16,22),Eph(16,23),Eph(16,24)];
ephSV17=[Eph(17,4),Eph(17,20),Eph(17,21),Eph(17,22),Eph(17,23),Eph(17,24)];
ephSV18=[Eph(18,4),Eph(18,20),Eph(18,21),Eph(18,22),Eph(18,23),Eph(18,24)];
ephSV19=[Eph(19,4),Eph(19,20),Eph(19,21),Eph(19,22),Eph(19,23),Eph(19,24)];
ephSV20=[Eph(20,4),Eph(20,20),Eph(20,21),Eph(20,22),Eph(20,23),Eph(20,24)];
ephSV21=[Eph(21,4),Eph(21,20),Eph(21,21),Eph(21,22),Eph(21,23),Eph(21,24)];
ephSV22=[Eph(22,4),Eph(22,20),Eph(22,21),Eph(22,22),Eph(22,23),Eph(22,24)];
ephSV23=[Eph(23,4),Eph(23,20),Eph(23,21),Eph(23,22),Eph(23,23),Eph(23,24)];
ephSV24=[Eph(24,4),Eph(24,20),Eph(24,21),Eph(24,22),Eph(24,23),Eph(24,24)];
ephSV25=[Eph(25,4),Eph(25,20),Eph(25,21),Eph(25,22),Eph(25,23),Eph(25,24)];
ephSV26=[Eph(26,4),Eph(26,20),Eph(26,21),Eph(26,22),Eph(26,23),Eph(26,24)];
ephSV27=[Eph(27,4),Eph(27,20),Eph(27,21),Eph(27,22),Eph(27,23),Eph(27,24)];
ephSV28=[Eph(28,4),Eph(28,20),Eph(28,21),Eph(28,22),Eph(28,23),Eph(28,24)];
ephSV29=[Eph(29,4),Eph(29,20),Eph(29,21),Eph(29,22),Eph(29,23),Eph(29,24)];
ephSV30=[Eph(30,4),Eph(30,20),Eph(30,21),Eph(30,22),Eph(30,23),Eph(30,24)];
ephSV31=[Eph(31,4),Eph(31,20),Eph(31,21),Eph(31,22),Eph(31,23),Eph(31,24)];
ephSV32=[Eph(32,4),Eph(32,20),Eph(32,21),Eph(32,22),Eph(32,23),Eph(32,24)];

%Database for storing optional raw data
database_range=[UTCtime, zeros(SATpositionsize(1),32)];
database_pseudorange=[UTCtime zeros(SATpositionsize(1),32)];
database_Doppler=[UTCtime zeros(SATpositionsize(1),32)];

run(' USB_receive_and_send.m')
```

Script 2: USB_receive_and_send

```
%Establishing connectivity to receiver for data stream in
serobj=serial('Serial Port of Receiver');
fopen(serobj);

%Sending to navigation section
serobj1=serial('COM4');
fopen(serobj1);

%Start looping of testbed
%%
```

```
str='start';
while str>0
   %Receive Message of POS and GSN
   two_message_retrieve
end
```

Script 3: two_message_retrieve

```
%Seperates and categorizes two incoming messages by title
firstmessage=fgetl(serobj);
secondmessage=fgetl(serobj);

if firstmessage(2)=='P' & secondmessage(2)=='P'
   return
elseif 0==isequal(firstmessage(2),'P') | 0==isequal(secondmessage(2),'G')
   return
else

%%
if firstmessage(2)=='P'
   Positioncommand=firstmessage;
elseif firstmessage(2)=='G'
   Satellitecommand=firstmessage;
end

if secondmessage(2)=='P'
   Positioncommand=secondmessage;
elseif secondmessage(2)=='G'
   Satellitecommand=secondmessage;
end
end

run(' ASHTECH_POS_command.m');
```

Script 4: ASHTECH_POS_command

```
%Data from receiver parse and values and defined by variable names
ASHTECHposCOMMAND=Positioncommand;
commaseparator=find(ASHTECHposCOMMAND==',');
%Detection of viewable satellites for interpretting navigation meassage
TotalSV=str2double(ASHTECHposCOMMAND(commaseparator(3)+1:commaseparator(4)-1));

if TotalSV<4
   fprintf(serobj1,'NotAvailable');
   return;
```

```
elseif TotalSV>=4
 %Data parsing and defining
 %Compressed current time
   curTime=str2double(ASHTECHposCOMMAND(commaseparator(4)+1:commaseparator(5)-
1));
   Hour=str2double(ASHTECHposCOMMAND(commaseparator(4)+1:commaseparator(4)+2));

Minute=str2double(ASHTECHposCOMMAND(commaseparator(4)+3:commaseparator(4)+4));

Second=str2double(ASHTECHposCOMMAND(commaseparator(4)+5:commaseparator(4)+6));
%Latitude

latdeg=str2double(ASHTECHposCOMMAND(commaseparator(5)+1:commaseparator(5)+2));

latmin=str2double(ASHTECHposCOMMAND(commaseparator(5)+3:commaseparator(5)+4));

latfractionofminute=str2double(ASHTECHposCOMMAND(commaseparator(5)+5:commasepar
ator(6)-1));
   latdirection=(ASHTECHposCOMMAND(commaseparator(6)+1));
   if latdirection=='N'
      latdirection=N;
   elseif latdirection=='S'
      latdirection=S;
   end
   anglelat=(latdeg+latmin/60+latfractionofminute*(60/3600))*latdirection; % Degrees
%Longitude

longdeg=str2double(ASHTECHposCOMMAND(commaseparator(7)+1:commaseparator(7)+3));

longmin=str2double(ASHTECHposCOMMAND(commaseparator(7)+4:commaseparator(7)+5))
;

longfractionofminute=str2double(ASHTECHposCOMMAND(commaseparator(7)+6:commasep
arator(8)-1));
   longdirection=(ASHTECHposCOMMAND(commaseparator(8)+1));
   if longdirection=='E'
      longdirection=E;
   elseif longdirection=='W'
      longdirection=W;
   end
   anglelong=(longdeg+longmin/60+longfractionofminute*(60/3600))*longdirection; %Degrees
%Altitude
   alt=str2double(ASHTECHposCOMMAND(commaseparator(9)+1:commaseparator(10)-1));
%meters
%TrueTrack
```

```
   TT=str2double(ASHTECHposCOMMAND(commaseparator(11)+1:commaseparator(12)-1));
%degrees
%Speed over Ground
   gv=str2double(ASHTECHposCOMMAND(commaseparator(12)+1:commaseparator(13)-1));
%knots
%Vertical Velocity
   VV=str2double(ASHTECHposCOMMAND(commaseparator(13)+1:commaseparator(14)-1));
%m/s
%DOP data
   PDOP=str2double(ASHTECHposCOMMAND(commaseparator(14)+1:commaseparator(15)-
1));
   HDOP=str2double(ASHTECHposCOMMAND(commaseparator(15)+1:commaseparator(16)-
1));
   VDOP=str2double(ASHTECHposCOMMAND(commaseparator(16)+1:commaseparator(17)-
1));
   TDOP=str2double(ASHTECHposCOMMAND(commaseparator(17)+1:commaseparator(18)-
1));
%Variables made from receiver data is sent to primary command
   run('PrimaryFunction.m');
end
```

Script 5: PrimaryFunction

```
%Variables from ASHTECH_POS_command script are sent to this script
%Conversion of WGS-84 units(latitude, longitude, and altitude) to ECEF
%units (X,Y, and Z)
ECEFpos=lla2ecef([anglelat,anglelong,alt]);

%Conversion of Body Frame units (true track, ground velocity and vertical
%velocity) to ECEF units (Vx,Vy,and Vz)
%Degree conversions to radians
LatRads=anglelat*((2*pi)/360); %rads
LongRads=anglelong*((2*pi)/360); %rads
TTrads=(TT)*((2*pi)/360); %rads
%Ground Velocity conversion from knots to m/s
GV=gv*0.51444444444444; %m/s
%Ground velocity vector components into Eastward and Northward velotiy
%Velocity East
Ve = GV.*sin(TTrads);   %m/s
%Velocity North
Vn = GV.*cos(TTrads);   %m/s
%Body Frame Conversion
NorthEastVerticle = [VV; Ve; Vn];
  unitN=[cos(LongRads).*sin(LatRads),sin(LongRads).*sin(LatRads),cos(LatRads)];
  unitE=[sin(LongRads),cos(LongRads),0];
  unitV=[cos(LongRads).*cos(LatRads),-sin(LongRads).*cos(LatRads),-sin(LatRads)];
```

```
NEVrot=[unitV;unitE;unitN];
ECEFVelocityxyz=NEVrot*NorthEastVerticle;
ECEFvx=ECEFVelocityxyz(1); %m/s
ECEFvy=ECEFVelocityxyz(2); %m/s
ECEFvz=ECEFVelocityxyz(3); %m/s
ECEFvel=[ECEFvx,ECEFvy,ECEFvz];
%Dynamics of vehicle with current time
DeputyKinematics=[ECEFpos ECEFvel];
initcond=[curTime DeputyKinematics];
%Packaging time-based dynamics for export
initcondsampled=num2str(initcond);
reduct1=strrep(initcondsampled,' ',',');
reduct2=strrep(reduct1,',,,,,',',');
reduct3=strrep(reduct2,',,,,',',');
reduct4=strrep(reduct3,',,,',',');
reduct5=strrep(reduct4,',,,',',');
reduct6=strrep(reduct5,',,',',');
reduct7=strrep(reduct6,',,',',');
reduct8=strrep(reduct7,',,',',')
%Export time based data
fprintf(serobj1,reduct8);

run('ASHTECH_GSN_command.m');
```

Script 6: ASHTECH_GSN_command

```
%The NMEA GSN message is sent to this script. This is an optional step.
%Data parsing of GSN command by satellite number only
ASHTECHgsnCOMMAND=Satellitecommand;
commaseparator2=find(ASHTECHgsnCOMMAND==',');

if length(commaseparator2)==10
%TotalSV=4
   SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
       str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
       str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
       str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1))];
 elseif length(commaseparator2)==12
 %TotalSV=5
   SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
       str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
       str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
       str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
```

```
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1))];
 elseif length(commaseparator2)==14
 %TotalSV=6
    SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(12)+1:commaseparator2(13)-
1))];
  elseif length(commaseparator2)==16
  %TotalSV=7
    SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(12)+1:commaseparator2(13)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(14)+1:commaseparator2(15)-
1))];
  elseif length(commaseparator2)==18
  %TotalSV=8
    SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(12)+1:commaseparator2(13)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(14)+1:commaseparator2(15)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(16)+1:commaseparator2(17)-
1))];
 elseif length(commaseparator2)==20
 %TotalSV=9
    SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
```

```
        str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(12)+1:commaseparator2(13)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(14)+1:commaseparator2(15)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(16)+1:commaseparator2(17)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(18)+1:commaseparator2(19)-
1))];
 elseif length(commaseparator2)==22
 %TotalSV=10
    SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(12)+1:commaseparator2(13)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(14)+1:commaseparator2(15)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(16)+1:commaseparator2(17)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(18)+1:commaseparator2(19)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(20)+1:commaseparator2(21)-
1))];
  elseif length(commaseparator2)==24
  %TotalSV=11
    SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(12)+1:commaseparator2(13)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(14)+1:commaseparator2(15)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(16)+1:commaseparator2(17)-
1)),...
```

```
        str2double(ASHTECHgsnCOMMAND(commaseparator2(18)+1:commaseparator2(19)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(20)+1:commaseparator2(21)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(22)+1:commaseparator2(23)-
1))];
 elseif length(commaseparator2)==26
 %TotalSV=12
    SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(12)+1:commaseparator2(13)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(14)+1:commaseparator2(15)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(16)+1:commaseparator2(17)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(18)+1:commaseparator2(19)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(20)+1:commaseparator2(21)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(22)+1:commaseparator2(23)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(24)+1:commaseparator2(25)-
1))];
 elseif length(commaseparator2)==28
 %TotalSV=13
    SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(12)+1:commaseparator2(13)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(14)+1:commaseparator2(15)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(16)+1:commaseparator2(17)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(18)+1:commaseparator2(19)-
1)),...
```

```
        str2double(ASHTECHgsnCOMMAND(commaseparator2(20)+1:commaseparator2(21)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(22)+1:commaseparator2(23)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(24)+1:commaseparator2(25)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(26)+1:commaseparator2(27)-
1))];
 elseif length(commaseparator2)==30
 %TotalSV=14
    SVs=[str2double(ASHTECHgsnCOMMAND(commaseparator2(2)+1:commaseparator2(3)-
1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(4)+1:commaseparator2(5)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(6)+1:commaseparator2(7)-1)), ...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(8)+1:commaseparator2(9)-1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(10)+1:commaseparator2(11)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(12)+1:commaseparator2(13)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(14)+1:commaseparator2(15)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(16)+1:commaseparator2(17)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(18)+1:commaseparator2(19)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(20)+1:commaseparator2(21)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(22)+1:commaseparator2(23)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(24)+1:commaseparator2(25)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(26)+1:commaseparator2(27)-
1)),...
        str2double(ASHTECHgsnCOMMAND(commaseparator2(28)+1:commaseparator2(29)-
1))];
end

   run('SecondaryFunction.m');
```

Script 7: SecondaryFunction

```
%Find the row of data for each GPS satellite corresponding to the current time
replaceindex=find(UTCtime==curTime);
if replaceindex>=0
```

```
%Take Each SV position and observables and feed it through range pseudorange and doppler
equation for database replacement
for i=1:length(SVs)
    SVID=SVs(i);
    if SVID==1
        SatECEFprev=SVID1(replaceindex-1,:);
        SatECEFcur=SVID1(replaceindex,:);
        SatECEFfut=SVID1(replaceindex+1,:);
        toe=ephSV1(1);
        af0=ephSV1(2);
        af1=ephSV1(3);
        af2=ephSV1(4);
        tgd=ephSV1(5);
        toc=ephSV1(6);
        SecondaryFunctionEquations
    elseif SVID==2
         SatECEFprev=SVID2(replaceindex-1,:);
        SatECEFcur=SVID2(replaceindex,:);
        SatECEFfut=SVID2(replaceindex+1,:);
        toe=ephSV2(1);
        af0=ephSV2(2);
        af1=ephSV2(3);
        af2=ephSV2(4);
        tgd=ephSV2(5);
        toc=ephSV2(6);
        SecondaryFunctionEquations
    elseif SVID==3
         SatECEFprev=SVID3(replaceindex-1,:);
        SatECEFcur=SVID3(replaceindex,:);
        SatECEFfut=SVID3(replaceindex+1,:);
        toe=ephSV3(1);
        af0=ephSV3(2);
        af1=ephSV3(3);
        af2=ephSV3(4);
        tgd=ephSV3(5);
        toc=ephSV3(6);
        SecondaryFunctionEquations
    elseif SVID==4
         SatECEFprev=SVID4(replaceindex-1,:);
        SatECEFcur=SVID4(replaceindex,:);
        SatECEFfut=SVID4(replaceindex+1,:);
        toe=ephSV4(1);
        af0=ephSV4(2);
        af1=ephSV4(3);
        af2=ephSV4(4);
        tgd=ephSV4(5);
```

```
    toc=ephSV4(6);
    SecondaryFunctionEquations
elseif SVID==5
     SatECEFprev=SVID5(replaceindex-1,:);
    SatECEFcur=SVID5(replaceindex,:);
    SatECEFfut=SVID5(replaceindex+1,:);
    toe=ephSV5(1);
    af0=ephSV5(2);
    af1=ephSV5(3);
    af2=ephSV5(4);
    tgd=ephSV5(5);
    toc=ephSV5(6);
    SecondaryFunctionEquations
elseif SVID==6
     SatECEFprev=SVID6(replaceindex-1,:);
    SatECEFcur=SVID6(replaceindex,:);
    SatECEFfut=SVID6(replaceindex+1,:);
    toe=ephSV6(1);
    af0=ephSV6(2);
    af1=ephSV6(3);
    af2=ephSV6(4);
    tgd=ephSV6(5);
    toc=ephSV6(6);
    SecondaryFunctionEquations
elseif SVID==7
     SatECEFprev=SVID7(replaceindex-1,:);
    SatECEFcur=SVID7(replaceindex,:);
    SatECEFfut=SVID7(replaceindex+1,:);
    toe=ephSV7(1);
    af0=ephSV7(2);
    af1=ephSV7(3);
    af2=ephSV7(4);
    tgd=ephSV7(5);
    toc=ephSV7(6);
    SecondaryFunctionEquations
elseif SVID==8
     SatECEFprev=SVID8(replaceindex-1,:);
    SatECEFcur=SVID8(replaceindex,:);
    SatECEFfut=SVID8(replaceindex+1,:);
    toe=ephSV8(1);
    af0=ephSV8(2);
    af1=ephSV8(3);
    af2=ephSV8(4);
    tgd=ephSV8(5);
    toc=ephSV8(6);
SecondaryFunctionEquations
```

```
elseif SVID==9
    SatECEFprev=SVID9(replaceindex-1,:);
   SatECEFcur=SVID9(replaceindex,:);
   SatECEFfut=SVID9(replaceindex+1,:);
   toe=ephSV9(1);
   af0=ephSV9(2);
   af1=ephSV9(3);
   af2=ephSV9(4);
   tgd=ephSV9(5);
   toc=ephSV9(6);
   SecondaryFunctionEquations
elseif SVID==10
    SatECEFprev=SVID10(replaceindex-1,:);
   SatECEFcur=SVID10(replaceindex,:);
   SatECEFfut=SVID10(replaceindex+1,:);
   toe=ephSV10(1);
   af0=ephSV10(2);
   af1=ephSV10(3);
   af2=ephSV10(4);
   tgd=ephSV10(5);
   toc=ephSV10(6);
   SecondaryFunctionEquations
elseif SVID==11
   SatECEFprev=SVID11(replaceindex-1,:);
   SatECEFcur=SVID11(replaceindex,:);
   SatECEFfut=SVID11(replaceindex+1,:);
   toe=ephSV11(1);
   af0=ephSV11(2);
   af1=ephSV11(3);
   af2=ephSV11(4);
   tgd=ephSV11(5);
   toc=ephSV11(6);
   SecondaryFunctionEquations
elseif SVID==12
    SatECEFprev=SVID12(replaceindex-1,:);
   SatECEFcur=SVID12(replaceindex,:);
   SatECEFfut=SVID12(replaceindex+1,:);
   toe=ephSV12(1);
   af0=ephSV12(2);
   af1=ephSV12(3);
   af2=ephSV12(4);
   tgd=ephSV12(5);
   toc=ephSV12(6);
   SecondaryFunctionEquations
elseif SVID==13
    SatECEFprev=SVID13(replaceindex-1,:);
```

```
    SatECEFcur=SVID13(replaceindex,:);
    SatECEFfut=SVID13(replaceindex+1,:);
    toe=ephSV13(1);
    af0=ephSV13(2);
    af1=ephSV13(3);
    af2=ephSV13(4);
    tgd=ephSV13(5);
    toc=ephSV13(6);
    SecondaryFunctionEquations
elseif SVID==14
     SatECEFprev=SVID14(replaceindex-1,:);
    SatECEFcur=SVID14(replaceindex,:);
    SatECEFfut=SVID14(replaceindex+1,:);
    toe=ephSV14(1);
    af0=ephSV14(2);
    af1=ephSV14(3);
    af2=ephSV14(4);
    tgd=ephSV14(5);
    toc=ephSV14(6);
    SecondaryFunctionEquations
elseif SVID==15
     SatECEFprev=SVID15(replaceindex-1,:);
    SatECEFcur=SVID15(replaceindex,:);
    SatECEFfut=SVID15(replaceindex+1,:);
    toe=ephSV15(1);
    af0=ephSV15(2);
    af1=ephSV15(3);
    af2=ephSV15(4);
    tgd=ephSV15(5);
    toc=ephSV15(6);
    SecondaryFunctionEquations
elseif SVID==16
     SatECEFprev=SVID16(replaceindex-1,:);
    SatECEFcur=SVID16(replaceindex,:);
    SatECEFfut=SVID16(replaceindex+1,:);
    toe=ephSV16(1);
    af0=ephSV16(2);
    af1=ephSV16(3);
    af2=ephSV16(4);
    tgd=ephSV16(5);
    toc=ephSV16(6);
    SecondaryFunctionEquations
elseif SVID==17
     SatECEFprev=SVID17(replaceindex-1,:);
    SatECEFcur=SVID17(replaceindex,:);
    SatECEFfut=SVID17(replaceindex+1,:);
```

```
    toe=ephSV17(1);
    af0=ephSV17(2);
    af1=ephSV17(3);
    af2=ephSV17(4);
    tgd=ephSV17(5);
    toc=ephSV17(6);
    SecondaryFunctionEquations
elseif SVID==18
     SatECEFprev=SVID18(replaceindex-1,:);
    SatECEFcur=SVID18(replaceindex,:);
    SatECEFfut=SVID18(replaceindex+1,:);
    toe=ephSV18(1);
    af0=ephSV18(2);
    af1=ephSV18(3);
    af2=ephSV18(4);
    tgd=ephSV18(5);
    toc=ephSV18(6);
SecondaryFunctionEquations
elseif SVID==19
     SatECEFprev=SVID19(replaceindex-1,:);
    SatECEFcur=SVID19(replaceindex,:);
    SatECEFfut=SVID19(replaceindex+1,:);
    toe=ephSV19(1);
    af0=ephSV19(2);
    af1=ephSV19(3);
    af2=ephSV19(4);
    tgd=ephSV19(5);
    toc=ephSV19(6);
    SecondaryFunctionEquations
elseif SVID==20
     SatECEFprev=SVID20(replaceindex-1,:);
    SatECEFcur=SVID20(replaceindex,:);
    SatECEFfut=SVID20(replaceindex+1,:);
    toe=ephSV20(1);
    af0=ephSV20(2);
    af1=ephSV20(3);
    af2=ephSV20(4);
    tgd=ephSV20(5);
    toc=ephSV20(6);
    SecondaryFunctionEquations
elseif SVID==21
    SatECEFprev=SVID21(replaceindex-1,:);
    SatECEFcur=SVID21(replaceindex,:);
    SatECEFfut=SVID21(replaceindex+1,:);
    toe=ephSV21(1);
    af0=ephSV21(2);
```

```
    af1=ephSV21(3);
    af2=ephSV21(4);
    tgd=ephSV21(5);
    toc=ephSV21(6);
    SecondaryFunctionEquations
elseif SVID==22
     SatECEFprev=SVID22(replaceindex-1,:);
    SatECEFcur=SVID22(replaceindex,:);
    SatECEFfut=SVID22(replaceindex+1,:);
    toe=ephSV22(1);
    af0=ephSV22(2);
    af1=ephSV22(3);
    af2=ephSV22(4);
    tgd=ephSV22(5);
    toc=ephSV22(6);
    SecondaryFunctionEquations
elseif SVID==23
     SatECEFprev=SVID23(replaceindex-1,:);
    SatECEFcur=SVID23(replaceindex,:);
    SatECEFfut=SVID23(replaceindex+1,:);
    toe=ephSV23(1);
    af0=ephSV23(2);
    af1=ephSV23(3);
    af2=ephSV23(4);
    tgd=ephSV23(5);
    toc=ephSV23(6);
    SecondaryFunctionEquations
elseif SVID==24
     SatECEFprev=SVID24(replaceindex-1,:);
    SatECEFcur=SVID24(replaceindex,:);
    SatECEFfut=SVID24(replaceindex+1,:);
    toe=ephSV24(1);
    af0=ephSV24(2);
    af1=ephSV24(3);
    af2=ephSV24(4);
    tgd=ephSV24(5);
    toc=ephSV24(6);
    SecondaryFunctionEquations
elseif SVID==25
     SatECEFprev=SVID25(replaceindex-1,:);
    SatECEFcur=SVID25(replaceindex,:);
    SatECEFfut=SVID25(replaceindex+1,:);
    toe=ephSV25(1);
    af0=ephSV25(2);
    af1=ephSV25(3);
    af2=ephSV25(4);
```

```
    tgd=ephSV25(5);
    toc=ephSV25(6);
    SecondaryFunctionEquations
elseif SVID==26
     SatECEFprev=SVID26(replaceindex-1,:);
    SatECEFcur=SVID26(replaceindex,:);
    SatECEFfut=SVID26(replaceindex+1,:);
    toe=ephSV26(1);
    af0=ephSV26(2);
    af1=ephSV26(3);
    af2=ephSV26(4);
    tgd=ephSV26(5);
    toc=ephSV26(6);
    SecondaryFunctionEquations
elseif SVID==27
     SatECEFprev=SVID27(replaceindex-1,:);
    SatECEFcur=SVID27(replaceindex,:);
    SatECEFfut=SVID27(replaceindex+1,:);
    toe=ephSV27(1);
    af0=ephSV27(2);
    af1=ephSV27(3);
    af2=ephSV27(4);
    tgd=ephSV27(5);
    toc=ephSV27(6);
    SecondaryFunctionEquations
elseif SVID==28
     SatECEFprev=SVID28(replaceindex-1,:);
    SatECEFcur=SVID28(replaceindex,:);
    SatECEFfut=SVID28(replaceindex+1,:);
    toe=ephSV28(1);
    af0=ephSV28(2);
    af1=ephSV28(3);
    af2=ephSV28(4);
    tgd=ephSV28(5);
    toc=ephSV28(6);
SecondaryFunctionEquations
elseif SVID==29
     SatECEFprev=SVID29(replaceindex-1,:);
    SatECEFcur=SVID29(replaceindex,:);
    SatECEFfut=SVID29(replaceindex+1,:);
    toe=ephSV29(1);
    af0=ephSV29(2);
    af1=ephSV29(3);
    af2=ephSV29(4);
    tgd=ephSV29(5);
    toc=ephSV29(6);
```

```
        SecondaryFunctionEquations
    elseif SVID==30
         SatECEFprev=SVID30(replaceindex-1,:);
        SatECEFcur=SVID30(replaceindex,:);
        SatECEFfut=SVID30(replaceindex+1,:);
        toe=ephSV30(1);
        af0=ephSV30(2);
        af1=ephSV30(3);
        af2=ephSV30(4);
        tgd=ephSV30(5);
        toc=ephSV30(6);
        SecondaryFunctionEquations
    elseif SVID==31
        SatECEFprev=SVID31(replaceindex-1,:);
        SatECEFcur=SVID31(replaceindex,:);
        SatECEFfut=SVID31(replaceindex+1,:);
        toe=ephSV31(1);
        af0=ephSV31(2);
        af1=ephSV31(3);
        af2=ephSV31(4);
        tgd=ephSV31(5);
        toc=ephSV31(6);
        SecondaryFunctionEquations
    elseif SVID==32
         SatECEFprev=SVID32(replaceindex-1,:);
        SatECEFcur=SVID32(replaceindex,:);
        SatECEFfut=SVID32(replaceindex+1,:);
        toe=ephSV32(1);
        af0=ephSV32(2);
        af1=ephSV32(3);
        af2=ephSV32(4);
        tgd=ephSV32(5);
        toc=ephSV32(6);
       SecondaryFunctionEquations
    end
end

else
    %do nothing but do not stop
end
```

Script 8: SecondaryFunctionEquations

```
%This script contains the equations used by the SecondaryFunction script to calculate range,
pseudo-range, and Doppler for each satellite
%Range calculation and storage
```

```
range = sqrt((SatECEFcur(1)-ECEFpos(1)).^2+(SatECEFcur(2)-
ECEFpos(2)).^2+(SatECEFcur(3)-ECEFpos(3)).^2);  %meters
database_range(replaceindex,SVID+1)=range;

%Pseudo-range calculation and storage
GPStime=Day*86400+Hour*3600+Minute*60+Second; %seconds
TEC=((tgd*c)/40.3)*((f^2*(1227.60*10^6)^2)/( f^2-(1227.60*10^6)^2));
ftgd=(40.3*TEC)/(c*f^2); %seconds
deltaS=-(af0 + af1.*(GPStime-toc) + af2.*(GPStime-toc).^2); %seconds
Pseudorange = range + c.*deltaS;  %meters
database_pseudorange(replaceindex,SVID+1)=Pseudorange;

%Doppler Calculation and storraage
SatcurVel=(SatECEFfut-SatECEFprev)/2; %m/s
deltaV=-(SatcurVel-ECEFvel); %m/s
term=SatECEFcur-ECEFpos; %meters
term2=norm(term); %meters
unitP=term/term2;
velR=dot(deltaV,unitP); %m/s
Doppler=velR/.19 %Hertz

database_Doppler(replaceindex,SVID+1)=Doppler;
```

Appendix B: Navigation Section Receiver Code

Script 1: Combinations

```
%File contained target path or chief data is uploaded to script
targetpath = xlsread('Chief or target path file');
tpdata= [targetpath(:,2) targetpath(:,3) targetpath(:,4)];
%Compressed time of chief or target path
ExcelTimeofChief=datestr(targetpath(:,1), 'HH:MM:SS');
Hdigit1=ExcelTimeofChief(:,1);
Hdigit2=ExcelTimeofChief(:,2);
MMdigit1=ExcelTimeofChief(:,4);
MMdigit2=ExcelTimeofChief(:,5);
SSdigit1=ExcelTimeofChief(:,7);
SSdigit2=ExcelTimeofChief(:,8);
TimeofChief=str2num(strcat(Hdigit1,Hdigit2,MMdigit1,MMdigit2,SSdigit1,SSdigit2));
%Initial acceleration value of deputy vehicle
Ao=[0 0 0.0001];
%Motion limits of deputy vehicle
Tstep=1;
MaxVelocity=510;
MaxAccel=10; %m/s^2
MinAccel=-10; %m/s^2
Maxjerk=15; %m/s^3
```

```
Minjerk=-Maxjerk; %m/s^3
%Zoning Parameters
Zone1MaxVelocity=.99*MaxVelocity; %m/s
Zone1upperbound=50; %meters
Zone2MaxVelocity=.985*MaxVelocity; %m/s
Zone2upperbound=20; %meters
%Turning Restraints
thetaval=360; %degrees
thetavalrad=thetaval*((2*pi)/180); %radians
thetalow=6;  %degrees
thetalowrad=thetalow*((2*pi)/180); %radians
thetamed=3;  %degrees
thetamedrad=thetamed*((2*pi)/180);  %radians
thetahigh=1.5;  %degrees
thetahighrad=thetahigh*((2*pi)/180);  %radians
%Creating range of all acceleration possibilites from deputy motion limits
x=linspace(MinAccel,MaxAccel,41); %m/s^2
y=linspace(MinAccel,MaxAccel,41); %m/s^2
z=linspace(MinAccel,MaxAccel,41); %m/s^2
Accelposs=allcomb(transpose(x),transpose(y),transpose(z)); %m/s^2
```

USBreceiveandsend

Script 2: USBreceiveandsend

```
%Establish connectivity to SimGen software
t=tcpip('Ip address', 'Ip address remote port number');
fopen(t);
%Set initial starting point
fprintf(t,'00:00:00,MOT,v1_m1,4063500,-255300,4893100,0,0,0,0,0,0');
fprintf(t,'RU');
display('press button to start')
%Wait command to allow receiver to acquired 4 or more GPS satellites
waitforbuttonpress
display('Simulation Started')
%Establish connectivity to raw measurement section to receive receiver data
serobj=serial('COM3');
fopen(serobj);
%Drain buffer to current data string in case buffer is backed up
for i=1:10
fgetl(serobj)
pause(.2)
end
display('drain successful')
fgetl(serobj);
%Start this sections iteration loop for producing motion
str='start';
```

```
while str>0
   str=fgetl(serobj);
   DeputyTrajectory;
end
```

Script 3: DeputyTrajectory

```
%Inputs of deputy position obtained from raw measurement section
InputfromRaw=str;
%Data parsing of string message
commaseparator=find(InputfromRaw==',');
Nodatacheck=isequal(InputfromRaw,'NotAvailable');
if Nodatacheck==1
  display('Awaiting Data')
   return
else
%%
%Variables of raw measurement data
curTime=str2double(InputfromRaw(1:commaseparator(1)-1));
Dox=str2double(InputfromRaw(commaseparator(1)+1:commaseparator(2)-1));
Doy=str2double(InputfromRaw(commaseparator(2)+1:commaseparator(3)-1));
Doz=str2double(InputfromRaw(commaseparator(3)+1:commaseparator(4)-1));
Vox=str2double(InputfromRaw(commaseparator(4)+1:commaseparator(5)-1));
Voy=str2double(InputfromRaw(commaseparator(5)+1:commaseparator(6)-1));
Voz=str2double(InputfromRaw(commaseparator(6)+1:end));
%Initial derived from receiver
Do=[Dox Doy Doz]; %Initial Input
Vo=[Vox Voy Voz]; %Initial Input
%Savepoint for retaining acceleration produced for acceleration at time 1
savedA=Ao;
%Position and velocity of deputy at the future position
D1=[Do(1)+(Vo(1).*Tstep)+(0.5.*Ao(1).*Tstep^2),Do(2)+(Vo(2).*Tstep)+(0.5.*Ao(2).*Tstep^
2),Do(3)+(Vo(3).*Tstep)+(0.5.*Ao(3).*Tstep^2)];
V1=[Vo(1)+Ao(1).*Tstep,Vo(2)+Ao(2).*Tstep,Vo(3)+Ao(3).*Tstep];
%Positions of chief for successive future positions
Cp1idx=find(TimeofChief==curTime)+1;
Cp2idx=Cp1idx+1;
Cp1=tpdata(Cp1idx,:);
Cp2=tpdata(Cp2idx,:);
%Successive future distances of chief relative to deputy at initial time
Distcur1=sqrt(((Cp1(1)-D1(1))^2)+((Cp1(2)-D1(2))^2)+((Cp1(3)-D1(3))^2));
Distcur2=sqrt(((Cp2(1)-D1(1))^2)+((Cp2(2)-D1(2))^2)+((Cp2(3)-D1(3))^2));
%Position and velocity possibilities generated from position, velocity, and
%the range of possible accelerations
V2possibilities=[V1(1)+Accelposs(:,1).*Tstep,V1(2)+Accelposs(:,2).*Tstep,V1(3)+Accelposs(:,
3).*Tstep];
```

```
P2possibilities=[(D1(1)+(V1(1).*Tstep)+(0.5.*Accelposs(:,1).*Tstep^2)),(D1(2)+(V1(2).*Tstep)
+(0.5.*Accelposs(:,2).*Tstep^2)),(D1(3)+(V1(3).*Tstep)+(0.5.*Accelposs(:,3).*Tstep^2))];
%Distance between cheif at time 2 for every position produced from the acceleration
possibilities
Distproj2=sqrt((Cp2(1)-P2possibilities(:,1)).^2+(Cp2(2)-P2possibilities(:,2)).^2+(Cp2(3)-
P2possibilities(:,3)).^2);
%Logic Filter 1:Restricted by Jerk limits-Acceleration impulses for each
%iteration
Axlimited=find(Accelposs(:,1)<=Ao(1)+Maxjerk & Accelposs(:,1)>=Ao(1)+Minjerk);
Aylimited=find(Accelposs(:,2)<=Ao(2)+Maxjerk & Accelposs(:,2)>=Ao(2)+Minjerk);
Azlimited=find(Accelposs(:,3)<=Ao(3)+Maxjerk & Accelposs(:,3)>=Ao(3)+Minjerk);
Axylimited=intersect(Axlimited,Aylimited);
Axyzlimited=intersect(Axylimited,Azlimited);
%Remaining results from first filter
AccelpossQ=Accelposs(Axyzlimited,:);
V3possibilitiesQ=V2possibilities(Axyzlimited,:);
P3possibilitiesQ=P2possibilities(Axyzlimited,:);
Distproj2Q=Distproj2(Axyzlimited,:);
%Logic Filter 2:possibilities of Filter 1 filtered by velocity limit and
%zoning logic
V3possibilitiesmag=sqrt(V3possibilitiesQ(:,1).^2+V3possibilitiesQ(:,2).^2+V3possibilitiesQ(:,3)
.^2);
if Distcur1>=Zone1upperbound
MaxVLimit=find(V3possibilitiesmag<=MaxVelocity);
elseif Distcur1<Zone1upperbound & Distcur1>=Zone2upperbound
MaxVLimit=find(V3possibilitiesmag<=Zone1MaxVelocity);
display('Zone1')
elseif Distcur1<Zone2upperbound
MaxVLimit=find(V3possibilitiesmag<=Zone2MaxVelocity);
display('Zone2')
end
%Remaining results after filter 2
AccelpossR=AccelpossQ(MaxVLimit,:);
V2possibilitiesR=V2possibilitiesQ(MaxVLimit,:);
P2possibilitiesR=P2possibilitiesQ(MaxVLimit,:);
Distproj2R=Distproj2Q(MaxVLimit,:);
%Logic Filter 3: Restricted by Turning limits
line1=D1-Do;
line2=[(P2possibilitiesR(:,1)-D1(1)),(P2possibilitiesR(:,2)-D1(2)),(P2possibilitiesR(:,3)-D1(3))];
magline1=sqrt(line1(1)^2+line1(2)^2+line1(3)^2);
magline2=sqrt(line2(:,1).^2+line2(:,2).^2+line2(:,3).^2);
maglines=magline1.*magline2;
g=[line1(1).*ones(length(line2(:,1)),1),line1(2).*ones(length(line2(:,2)),1),line1(3).*ones(length(
line2(:,3)),1)];
dotp=dot(g,line2,2);
angle_OtN=acosd(dotp./maglines);
```

```
T=find(angle_OtN<(thetaval));
%Remaining results after filter 3
AccelpossT=AccelpossR(T,:);
V2possibilitiesT=V2possibilitiesR(T,:);
P2possibilitiesT=P2possibilitiesR(T,:);
Distproj2T=Distproj2R(T,:);
%List remaining results from filters with calculated dynamics from time 1
Approvedt1dynamics=[D1(1).*ones(size(T)) D1(2).*ones(size(T)) D1(3).*ones(size(T))
V1(1).*ones(size(T)) V1(2).*ones(size(T)) V1(3).*ones(size(T)) AccelpossT Distproj2T];
%Select one result that generates the smallest distance between the deputy
%and target path
selection=min(Distproj2T);
Finalselection=Approvedt1dynamics(Approvedt1dynamics(:,10)==selection,:);
%Inputs for next time step
NextInputs=Finalselection(1,:);
Nexttime=TimeofChief(Cp1idx);
NextInputsinfo=NextInputs(1:9);
%Package next input data with time and create remote motion command for
%remote control section
info1=ExcelTimeofChief(Cp1idx,1:8);
info2=',MOT,v1_m1,';
NextInputsinfosampled=num2str(NextInputsinfo);
reduct1=strrep(NextInputsinfosampled,' ',',');
reduct2=strrep(reduct1,',,',',');
reduct3=strrep(reduct2,',,,,,',',');
reduct4=strrep(reduct3,',,,,',',');
reduct5=strrep(reduct4,',,,',',');
reduct6=strrep(reduct5,',,',',');
reduct7=strrep(reduct6,',,',',');
reduct8=strrep(reduct7,',,',',');
reduct9=strrep(reduct8,',,',',');
reduct10=strrep(reduct9,',,',',');
reduct11=strrep(reduct10,',,',',');
info3=reduct11;
fullinfo=strcat(info1,info2,info3);
%Send data to SimGen remote control section
fprintf(t,fullinfo);
%Saving process for acceleration for next iteration
A1=NextInputs(7:9);
Ao=A1;
end
```