A GPS and GLONASS L1 Vector Tracking Software-Defined Receiver

by

Tanner M. Watts

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
August 3, 2019

Keywords: GPS, GLONASS, Software-Defined Receiver, Vector Tracking

Approved by

David Bevly, Chair, Bill and Lana McNair Endowed Professor of Mechanical Engineering
Scott Martin, Assistant Research Professor of Mechanical Engineering
Stanley Reeves, Professor of Electrical and Computer Engineering

Abstract

Originally designed for military navigation, GPS has exploded into a modern tool used by government, industrial, and commercial sectors. The increasing demand for navigation has led to the operation of GPS receivers in challenging signal environments that include suburbs, forested areas, and metropolitan cities. In recent years, GPS receivers have also come under attack from commercial jamming and spoofing devices. To assure the integrity of GPS, the ability for receivers to overcome challenging signal environments must be solved.

GPS vectorized signal tracking (vector tracking) has been shown to enhance receiver performance by 2 to 6 dB in poor signal environments over traditional tracking methods that rely on loop filters (scalar tracking). GLONASS, the Russian Federation's equivalent to GPS, is another system that can be used for navigation. Today, many receivers use both GPS and GLONASS with scalar processing. Implementing the constellations into a centralized vector tracking filter gives the opportunity of enhanced navigation capability in challenging areas.

In this thesis, the development and analysis of a software receiver that uses GPS and GLONASS vector tracking is performed. Specifically, the software receiver uses a centralized Vector Delay/Frequency Lock Loop (VDFLL) Kalman filter implementation to track the code and carrier dynamics of the satellite signals. Cascaded Phase Lock Loop (PLL) aiding is applied to the satellite channels to maintain carrier phase lock.

Simulation results showed the software receiver's ability to maintain accurate navigation in GPS or GLONASS jamming environments. In GPS jamming environments, GLONASS was able to maintain accurate tracking replicas of the GPS channels through the VDFLL. Experimental results from forested areas and urban canyons showed that the software receiver performed better with vector tracking than scalar tracking. Depending on the experiment, GPS and GLONASS vector tracking outperformed GPS-only vector tracking. In some environments, GLONASS became degraded, which caused noise sharing issues in the software receiver's vector processing algorithm.

Acknowledgments

First and foremost, I thank my family for their consistent support, motivation, and love. I am truly blessed to have such a great family. I thank my dad, Mike, for helping me collect the data used in this thesis. I thank my sister, Tori, and my mom, Stacey, for proofreading this entire thesis. I also thank my best friend, Dillon, and my brother, Seth, for the encouragement they have given me to complete my engineering studies over the years.

I would like to thank my advisor, Dr. David Bevly, who graciously offered me a position in his research lab. I appreciate the challenges and opportunities Dr. Bevly has given me. I must thank my boss and mentor, Dr. Scott Martin, who accelerated my knowledge of GPS. When I first began graduate school, Dr. Martin gave me a lot of help to kickstart my research, which I cannot appreciate enough. The work in this thesis would have not been possible without Dr. Martin.

I thank Dr. Jeffrey Fergus, who invited me to visit Auburn while I was still studying at Iowa State University. I would like to acknowledge Dr. Stanley Reeves and Dr. John Hung for the their excellent lectures and coursework on signal processing and estimation.

I want to thank all of the members of the GAVLAB. It is an honor to work with the smartest, hardest-working people I know. I thank Ethan Edwards, Logan Cruse, Tanner Ray, Joe Selikoff, Patrick Smith, and Robert Brothers for their friendship and support. I enjoyed all of the reseach, class, and daily life conversations we had. I also want to acknowledge Troupe Tabb, Luke Kamrath, Ryan Shaw, Nick Williams, and Howard Chen for their helpful inputs on the work in this thesis. I thank former GAVLAB members, Matthew Lashley and Benjamin Clark, for their excellent PhD dissertations on vector tracking loops, which were very helpful to me. I wish to thank another former GAVLAB member, Luke Edwards, for the help he has given me on vector tracking implementation.

Finally, I would like to thank my Auburn relatives, Terry and Mari Ley, for welcoming me to Auburn. They have made me feel at home in Alabama.

Table of Contents

List of Figures

List of Tables

List of Acronyms

ADC	Analog to Digital Converter
bps	Bits Per Second
BPSK	Binary Phase Shift Keying
C/A	Coarse Acquisition
C/N_o	Carrier-to-Noise Power Density
CDMA	Code Division Multiple Access
cps	Chips Per Second
CRT	Common Reception Time
CSAC	Chip Scale Atomic Clock
CTT	Common Transmission Time
dB	Decibels
dB-Hz	Decibels Hertz
DLL	Delay Lock Loop
DOP	Dilution of Precision
DVPLL	Differential Vector Phase Lock Loop
ECEF	Earth-Centered Earth-Fixed
ECI	Earth-Centered Inertial
EKF	Extended Kalman Filter
EM	Electromagnetic
ENU	East-North-Up

FAA	Federal Aviation Administration
FDE	Fault Detection and Exclusion
FDMA	Frequency Division Multiple Access
FLL	Frequency Lock Loop
GAVLAB	GPS and Vehicle Dynamics Laboratory
GPS	Global Positioning System
GLONASS	Global Navigation Satellite System (Russian Federation)
GNSS	Global Navigation Satellite System
HOW	Hand-Over Word
Hz	Hertz
ICD	Interface Control Document
IF	Intermediate Frequency
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
ITRF2008	International Terrestrial Reference Frame 2008
lla	Latitude, Longitude, and Altitude
NCO	Numerically Controlled Oscillator
NED	North-East-Down
OCXO	Oven Controlled Crystal Oscillator
PLL	Phase Lock Loop
PRN	Pseudo-Random Noise
PVT	Position, Velocity, and Time
PZ-90	Parametry Zemli 1990

RLS	Recursive Least Squares
RTK	Real-Time Kinematic
RTK VPLL	Real-Time Kinematic Vector Phase Lock Loop
SBAS	Satellite-Based Augmentation System
SDR	Software-Defined Receiver
SNR	Signal-to-Noise Ratio
TCXO	Temperature Controlled Crystal Oscillator
TOD	Time of Day
TOW	Time of Week
UTC	Universal Coordinated Time
VDLL	Vector Delay Lock Loop
VDFLL	Vector Delay/Frequency Lock Loop
VFLL	Vector Frequency Lock Loop
VPLL	Vector Phase Lock Loop
WAAS	Wide-Area Augmentation System
WGS84	World Geodetic System 1984
WRLS	Weighted Recursive Least Squares

Chapter 1

Introduction

1.1 Background and Motivation

The Global Positioning System (GPS) has become an important navigation tool in modern society. Localization and timing services from GPS and other Global Navigation Satellite Systems (GNSS) are used in several commercial, government, and military sectors. Table 1.1 outlines these sectors. Within each of the sectors shown are several specific applications of GPS.

Table 1.1: Applications of GPS and GNSS

Navigation	**Timing**	**Remote Sensing**
Surveying	Wireless Communication	Meterology and Climatology
Robotics	Timekeeping	Atmospheric Science
Pedestrian Tracking	Banking	Space Weather
Precision Farming	Trading and Finance	Agriculture
Land and Marine Navigation	Electrical Power Grids	Geology
Air and Space Navigation	Sensor Fusion	Oceanography and Hydrology

Over time, the demand for GPS in challenging environments by users has grown. These challenging environments include forests, metropolitan cities, and indoor areas. GPS receivers rely on low-power signals transmitted from satellites 20,000 km away in space. Signal quality is important for GPS receivers to accurately navigate. Challenging environments tend to distort, degrade, and weaken the satellite signals, which in turn causes GPS receivers to perform poorly.

In recent years, civilian GPS receivers have come under jamming and spoofing attacks. Jammers generate noisy, high-power broadcasts at the GPS frequency that crowd the signal environment, in turn making it nearly impossible for receivers to process the satellite signals. Originally, jamming was a military issue, but has become a civilian problem through the ease of obtaining commercial jammers. Spoofing is a newer form of GPS degradation, where a device transmits fake GPS signals in attempt to distort or trick a receiver's navigation. There are some commerical spoofers available and several GPS simulators for sale that can be reconfigured as spoofers. Figure 1.1 shows pictures of a GPS jammer (left) and GPS simulator (right) that are commercially available.

Figure 1.1: Jammer [7] and Simulator [17] Devices that can be used to degrade GPS

The motivation for this thesis is to discuss research performed with the intended goal of enhancing a GPS receiver's navigation capabilities in challenging signal environments. The ability for receivers to navigate in degraded signal environments allow GPS use to be expanded to more areas of society, especially urban cities. This also helps assure the long term use of GPS in the presence of jamming and spoofing threats.

1.2 Prior Art

GPS receivers fail in challenging signal conditions because the receiver is unable to process the satellite broadcasts and extract navigation measurements used for positioning. The receiver processing of the satellite signals is known as signal tracking. The performance of any GPS receiver is dependent on whether the receiver can effectively track the satellite signals. Since the late 1970's, there has been significant research and innovation on enhancing GPS signal processing. Specifically, increasing tracking performance in poor signal environments. An overview of this research is given below.

1.2.1 Software Receivers

Originally, GPS research was tested with electrical hardware receivers. Developing and re-designing hardware receivers takes time, material, and the skill of building advanced circuits. Today, the majority of GPS tracking research takes place with software defined receivers. Software receivers allow tracking algorithms to be coded onto a computer instead of built into a circuit. This saves significant time in development and has accelerated GPS research.

The use of software receivers for development was discussed for decades and became feasible in the 1990's due to innovations in computer technology. A thorough discussion of GNSS signal processing with a software defined receiver was published by Dennis Akos in his PhD dissertation at Ohio University [12]. Research with software defined receivers became more popular with the publication of *A Software Defined GPS and Galileo Receiver: A Single Frequency Approach* in 2007 [26]. Today, there is discussion and research on moving GNSS software receivers into the commercial market.

Figure 1.2 shows a diagram of a typical GPS software receiver. The receiver tracks each satellite signal with an individual feedback control loop using an internal replica. The receiver-generated replica is updated and corrected with discriminator feedback on each satellite channel. The tracking loops then extract measurements that are inputted into the navigation processor to estimate the receiver's position, velocity, and time (PVT). This form of signal processing is known as scalar tracking.

The origin of the tracking loop design used in GPS receivers dates back to early signal analysis in the 1930's. Eventually, the Phase Lock Loop (PLL) became prevalent for signal tracking loops in many applications during the 1950's [23]. Since then, there have been many contributions to scalar tracking loops. Advances in loop filter design were originally published by Phillip Ward [50]. In the early 2000's, replacing traditional loop filters with Kalman filters for GPS signal tracking became popular [29], [9], and [30].

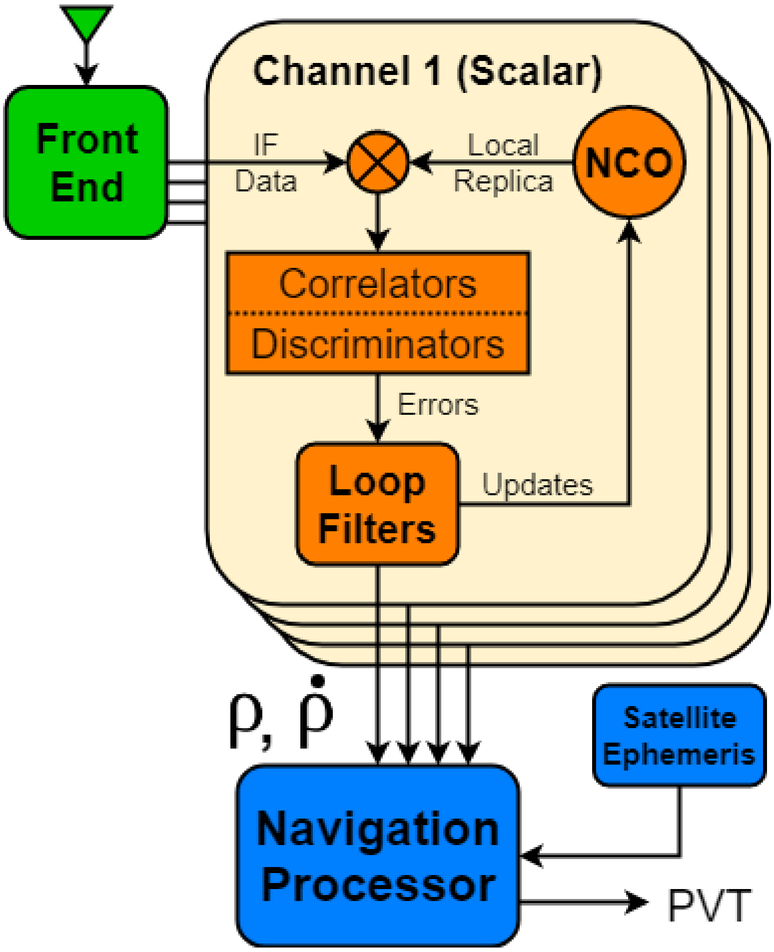

Figure 1.2: Diagram of a Software Defined GPS Scalar Tracking Receiver

In most situations, scalar tracking loops perform well. Scalar loops become limited in performance when the receiver is introduced to degraded signal environments. This occurs because each satellite channel's tracking loop is fully dependent on its discriminator feedback. When a satellite signal is distorted by the environment, its discriminator feedback becomes a very poor representation of tracking error. As with any feedback control loop, when the feedback is no longer accurate, the loop fails. Signal tracking failure results in the navigator using distorted measurements that ultimately produces inaccurate positioning and timing.

1.2.2 Vector Tracking

An effective way to maintain tracking of a weak/noisy satellite channel is to provide feedback tracking error from another source, which is typically the receiver's navigation solution. In this type of implementation, control loops on each satellite channel are replaced with the navigation estimator. The receiver's navigation estimator is used to couple the satellite channels together. The channels work together to track each others' signals, which allows stronger channels to aid the tracking of weaker channels. This solution is known as vector tracking, because the signal processing has been "vectorized" amongst all the satellite channels. Figure 1.3 provides a diagram of a vector tracking GPS receiver.

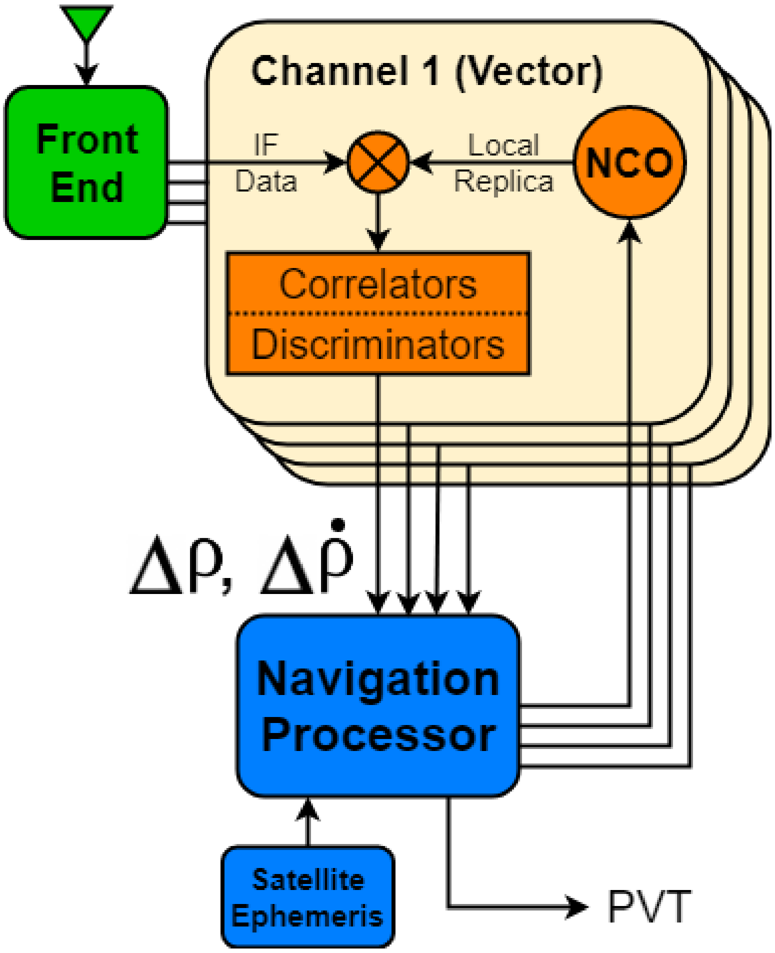

Figure 1.3: Diagram of a Software Defined GPS Vector Tracking Receiver

Figure 1.4 illustrates the advantage of vector tracking over scalar tracking. The figure shows the frequency for a GPS signal over time using scalar and vector processing. In the first half of the plot, the satellite channel is in a strong signal environment that results in good signal frequencies for both tracking architectures. In the second half of the plot, the satellite channel becomes degraded due to entering a poor signal environment. The scalar frequency becomes very noisy because the loop filter is fully reliant upon distorted feedback from the degraded channel. The vector processing reduces the noise significantly in the bad signal conditions by using feedback from other channels through the navigation estimator. The other channels' discriminators do not contain distorted feedback, which makes them useful for aiding the degraded channel's frequency estimation. The scalar loop filter could lower its bandwidth to reject more noise, but it then may no longer be able to track the degraded channel's signal dynamics.

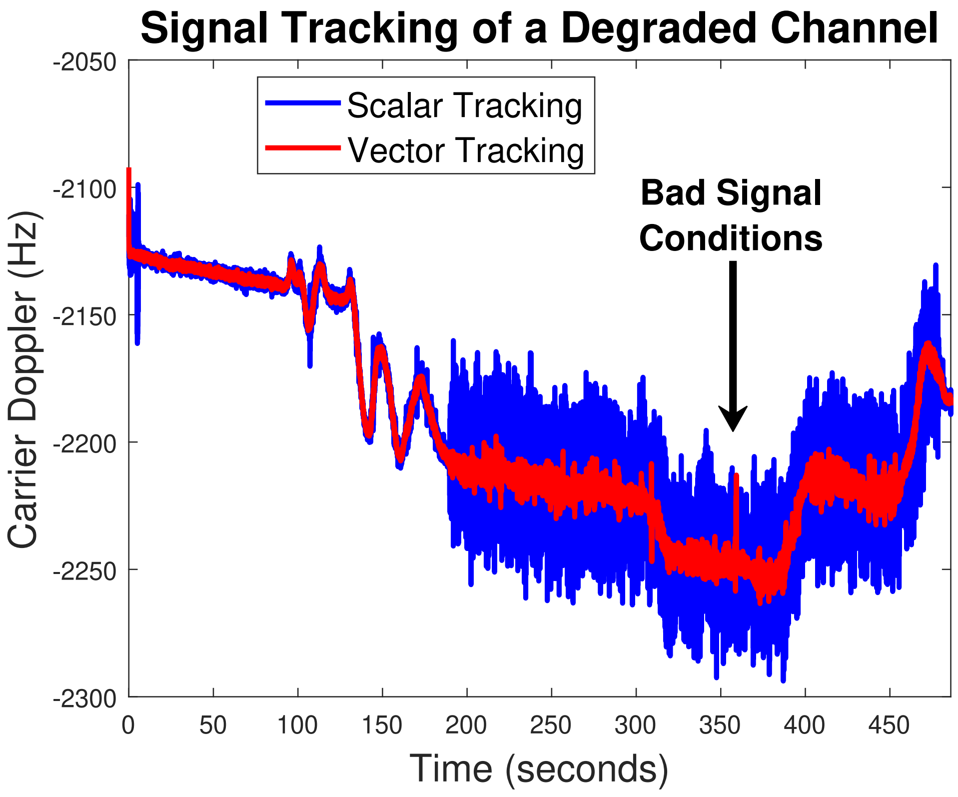

Figure 1.4: Scalar and Vector Tracking for a GPS Channel in a Degraded Environment

The inception of vector tracking came from the publication, *Optimal Processing of GPS Signals*, in 1980 [14], where the authors described a centralized Kalman filter estimator for code tracking. During the 1980's, U.S funded defense companies and academic institutions continued research into vector processing of GPS signals. In the 1990's, vector tracking patents were produced by Jim Sennott and Dave Senffner from Bradley University [24], and James Spilker from Stanford University [21].

Due to the complexity of vector tracking and selective availability, there was no significant performance analysis of vector loops in the 80's and 90's, only theoretical descriptions. At the turn of the century, with improvements in computer technology, vector tracking analysis was initiated by Don Benson from MITRE Corporation [11]. Significant contributions in vector tracking analysis were made by Lashley and Bevly [32], [34], [36], [37], and [39]. They focused on comparisons between vector and scalar tracking loops. Their research concluded that vector processing had between 2-6 dB improvement in tracking performance over scalar processing, depending on the receiver dynamics. Since the publication of Matt Lashley's PhD dissertation on the Vector Delay/Frequency Lock Loop (VDFLL) at Auburn University [31], there has been several papers on GPS vector tracking. Most of these papers discuss different applications of the algorithm [66] and [27].

Although vector processing can improve GPS navigation in challenging signal conditions, the algorithm does have disadvantages. In general, vector tracking is more complicated and time consuming than scalar tracking, which is why it is not a common algorithm used in commercial receivers. Vector tracking also suffers from noise sharing where weaker channels degrade tracking operations on stronger channels. Like all GPS navigation algorithms, vector tracking still requires at least 4 strong satellite channels to navigate. In the case where all GPS signals are severely degraded, vector tracking will perform just as poorly as scalar tracking.

Figure 1.5 illustrates vector tracking performance in the presence of GPS jamming. In the figure, receiver channel signal powers are shown before, during, and after a simulated jamming test. Prior to the jamming test, the signal powers are high, which indicates the receiver is accurately tracking the satellite signals. During the jamming test, the signals are so degraded the receiver cannot track, which results in low signal powers. After the jamming test, the signal powers remain low because the vector tracking loop lost lock of all the signals. When jamming occurred, the vector receiver did not have any information from the signals to produce a navigation solution, hence it could not track the signals. If the vector receiver had information from another navigation source, the signal powers would have returned to normal values after the jamming ended.

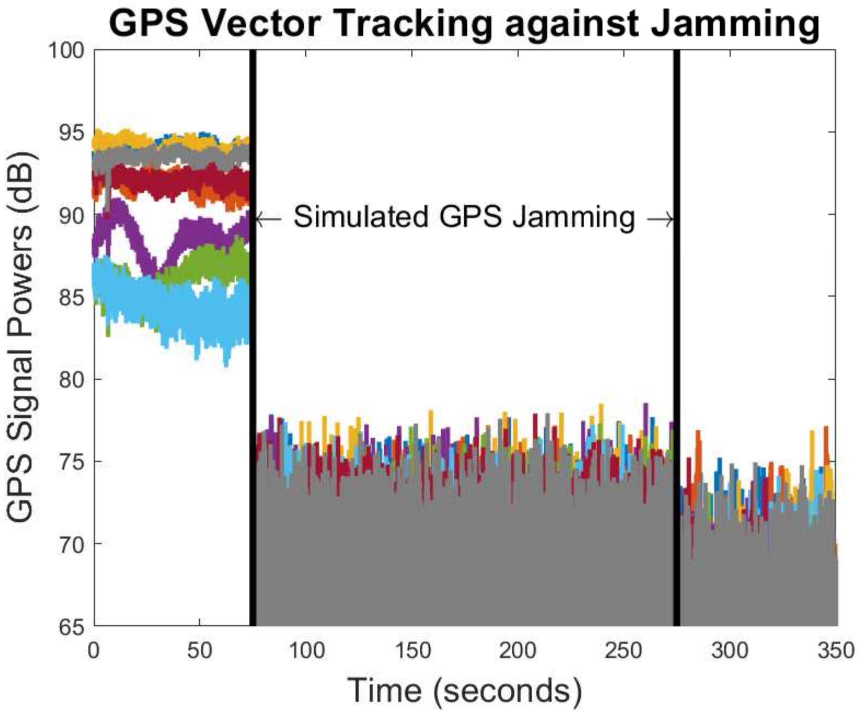

Figure 1.5: GPS Vector Tracking Satellite Channel Signal Powers Before and After Jamming

1.2.3 Deeply Integrated GPS/INS

A common way to improve vector tracking is to aid the algorithm with another navigation sensor, which is generally an inertial measurement unit (IMU). This form of vector tracking is known as deeply integrated GPS/INS. In general, IMU's contain noisy navigation measurements that make their helpfulness to GPS receivers limited. In the deeply integrated algorithm, the GPS receiver and the IMU form a symbiotic relationship where they aid each other's navigation. Figure 1.6 shows a diagram of a deeply integrated GPS/INS receiver. The IMU sends measurements to the receiver's navigation processor that estimates receiver position as well as corrections for the noisy IMU. When GPS is not degraded, the receiver provides corrections to the IMU to keep its measurements accurate. When GPS is degraded, the receiver can rely on the IMU for a limited period to navigate.

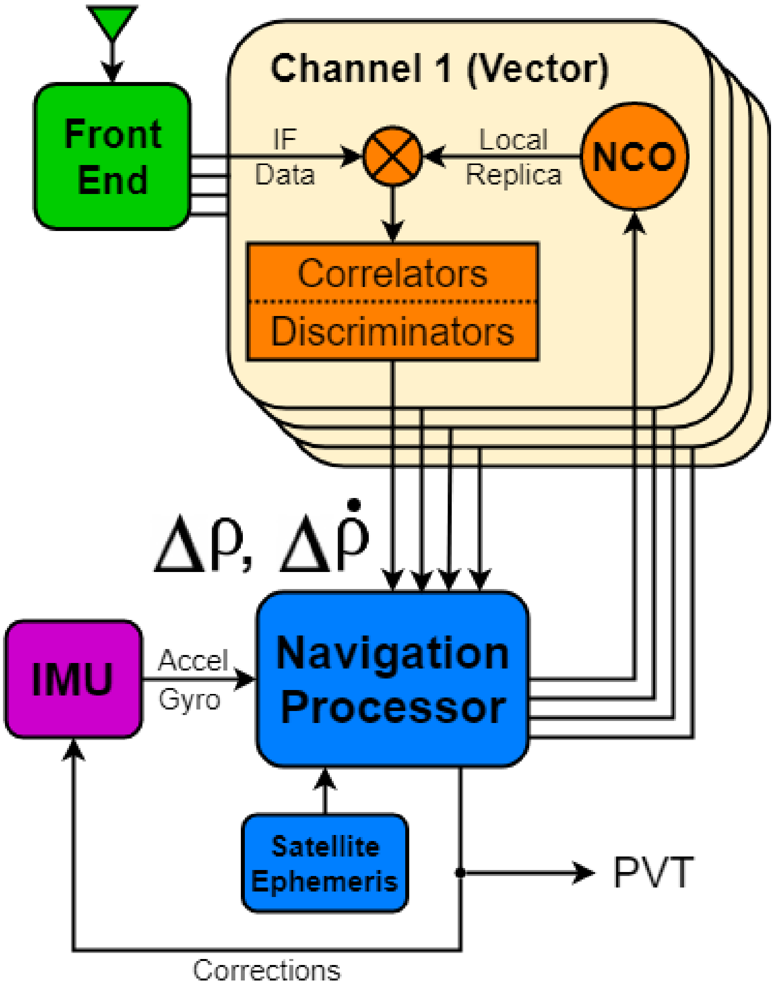

Figure 1.6: Diagram of a Software Defined Receiver using Deeply Integrated GPS/INS

The idea of deeply integrated GPS/INS has been around as long as vector tracking. In 1999, Jeff Horslund and Jonathan Hooker from Raytheon patented a deeply integrated receiver for GPS jamming immunity [25]. In 2003, Anthony Abbott and Walter Lillo from the The Aerospace Corporation patented an ultra tightly coupled GPS/INS receiver that provided improvements to deeply integrated GPS/INS with new filtering techniques [4]. Since the publication of these patents, there have been many papers on analysis and performance of deeply integrated GPS/INS with varying IMU grades.

Some papers have shown that deeply integrated GPS/INS improves navigation at very high dynamics [33] and [38]. Others showed that deeply integrated GPS/INS can navigate when GPS is jammed, for a period of time that depends on the quality of the IMU [10] and [15]. However, deeply integrated GPS/INS has some disadvantages. The algorithm is very complex.

These complexities include the synchronziation of GPS and IMU data, estimation of attitude, and estimation of IMU measurement error. The issue of data synchronization is discussed in [71]. Ben Clark analyzed the application of deeply integrated GPS/INS for vehicle navigation in his PhD dissertation at Auburn University [5]. His results concluded that deeply integrated GPS/INS performed worse than vector tracking when using an automotive grade IMU. For the GPS/INS algorithm to perform better than vector tracking, the navigation processor would need to rely on a higher quality IMU, which becomes expensive. Currently, deeply integrated GPS/INS is more suited for military applications. For general, low-acceleration civilian navigation in challening signal environments, low-grade IMU's will likely deteriorate the position solution further.

1.2.4 GLONASS

The United States is not the only country that operates a satellite system for positioning. Originally developed by the USSR and now owned by the Russian Federation, the Global Navigation Satellite System (GLONASS) is another GNSS constellation. In 1995, the system reached global coverage and many research papers on GLONASS were published. Pratap Misra from Lincoln Laboratory pioneered the integration of GPS and GLONASS into an estimator [51] and [52]. In 1998, GPS and GLONASS were integrated together for precise machine control in mining operations by Lyle Johnson and Frank van Diggelen [28]. A thorough discussion and analysis of integrating GPS and GLONASS was performed by Chao-heh Cheng in his 1998 Master's Thesis at Ohio University [8]. In 1999, due to funding problems in the Russian Federation, GLONASS lost global coverage status and research on the system dropped.

By 2010, the Russian Federation was able to reinstate global coverage for GLONASS and attention to the system resurged. Many papers were published on software receiver architecture that integrated GPS and GLONASS together with scalar tracking loops [58], [62], [47], [22], and [57]. Frank van Diggelen and his team at Google pushed for the integration of GLONASS into smart phones. Today, most smart phones include GLONASS, along with GPS, into their positioning techonology. The Russian Federation plans to continue funding GLONASS as well as adding new satellites to the constellation.

The publications cited previously have shown that adding GLONASS to a receiver does not always improve position accuracy. In many cases, GLONASS can degrade accuracy when integrated with GPS due to signal measurement resolution and error that can occur from its modulation design. However, including GLONASS into a receiver provides diversity in satellite geometry, signal frequency, and constellation use. Diversity in signal frequency makes it more challenging for a receiver to be jammed, and diversity in constellation use makes it more challenging for a receiver to be spoofed. In other words, a GPS and GLONASS receiver will be more resilient to jamming and spoofing threats if only one of the constellations is jammed or spoofed.

Taking advantage of both GPS and GLONASS in challenging environments improves the probability that the receiver will maintain enough satellites to continue navigating. For vector tracking, the use of GPS and GLONASS could provide the potential for the constellations to aid each other's tracking operations. Overall, using both constellations may improve the stability of vector tracking in challenging signal conditions.

There has been some work on GPS and GLONASS vector tracking. The first publication on the topic was written by Javad Positoning Systems in 1998 [43]. They discussed a receiver that improved carrier phase tracking by inputting the discriminator errors into a GPS and GLONASS navigation processor. The technique performed by Javad is not true vector tracking, but is known as cooperative tracking. The paper focused more on the tracking algorithm than the use of both GPS and GLONASS. Recently in 2017 and 2018, Iran University of Technology and Science published work comparing federated (scalar) and vectorized tracking architectures in GPS and GLONASS receivers [2] and [3]. The papers analyzed signal blockage environments and compared their results to the Javad receiver.

There has not been significant work on the analysis of GPS and GLONASS signal processing with purely vector methods, such as Copps' or Lashley's algorithms. There also has not been analysis on GPS and GLONASS vector tracking in challenging signal conditions, such as forested areas. These topics need to be analyzed and discussed before navigators will adopt GPS and GLONASS vector tracking as a tool for GNSS receivers.

1.3 Research Contributions

There have been many contributions to vector tracking algorithms, most of which are associated to GPS signals. The main goal and contribution of this thesis is the development and analysis of a GPS and GLONASS vector tracking software-defined receiver for navigation in challenging GNSS signal environments. The research objectives this thesis contributes to the field of navigation are listed below.

- Provide a thorough discussion of GPS and GLONASS software-defined receivers
- Provide a thorough discussion of GNSS vector signal tracking
- Implementation of a GPS and GLONASS software-defined receiver
- Validation of GPS and GLONASS vector tracking capabilities
- Analysis of different receiver configurations using GPS, GLONASS, scalar tracking, and vector tracking in challenging signal environments

1.4 Thesis Outline

This thesis consists of seven chapters and an appendix. Chapter 1 introduces the problem of navigating in challenging signal environments and the potential solution by integrating GPS and GLONASS into a vector tracking software receiver. Chapter 2 discusses the GPS and GLONASS signals used by the software receiver in detail. Chapter 3 describes GNSS software receiver architecture, as well as a description of scalar signal tracking loops. Chapter 4 analyzes how signal measurements are extracted from the receiver for navigation. The measurement errors induced from challenging signal environments are also explained in the fourth chapter. Chapter 5 discusses vector tracking with an emphasis on the Vector Delay/Frequency Lock Loop (VDFLL). The first half of Chapter 6 explains the integration of GPS and GLONASS into a vector processing receiver. The second half of Chapter 6 analyzes the performance of the vector receiver with respect to other signal tracking architectures in challenging environments. Analysis in Chapter 6 is conducted with simulation and experimental results. Chapter

7 summarizes the findings produced by this thesis and concludes with future work to improve the software receiver's navigation capabilities.

The appendix contains three independent discussions. Appendix A goes over the Extended Kalman Filter (EKF) used in the GPS and GLONASS software receiver. Appendix B provides navigation results for different methods of measurement synchronization for the receiver's vector tracking algorithm. Appendix C analyzes receiver clock bias results when one clock bias is estimated for both constellations and when two clock biases are estimated for each constellation.

Chapter 2

GNSS Signals

Today, there are several global and regional satellite constellations that transmit electromagnetic signals to provide users with navigation and timing services. The oldest constellation is GPS, owned and operated by the United States. The Russian Federation operates GLONASS, the European Union operates Galileo, and China operates Beidou; all satellite constellations that provide global coverage. Japan and India operate regional satellite systems. All the systems' ranging signals are similar in design, but vary in modulation and transmission. In this work, the original GPS and GLONASS civilian signals are used to develop a receiver that performs robust signal tracking and provides accurate navigation results. The rest of this chapter will discuss the GPS L1 C/A signal and the GLONASS L1 signal.

2.1 The GPS L1 C/A Signal

The Global Positioning System (GPS) is a constellation of orbiting satellites that transmit ranging signals to the Earth's surface. Receivers on the Earth use these signals to determine their position, velocity, and time. GPS was the first developed global navigation satellite system (GNSS), launched by the Department of Defense in 1973 and fully operational by 1995. GPS satellites transmit both military and civilian signals. Military signals are only authorized for use by the Department of Defense and are inaccessible to commercial and civilian receivers. Therefore, GPS military signals will not be discussed further in this work. Currently, the GPS constellation transmits three civilian signals; the L1 C/A signal that transmits at 1575.42 Mega-Hertz (MHz), the L2C signal that transmits at 1227.60 MHz, and the L5 signal that transmits at 1176 MHz. All current GPS satellites transmit the L1 C/A signal and some transmit the L2C and L5 signals. The L1 C/A signal is the original GPS signal, known as the legacy signal, most receivers use L1 C/A for navigation and timing. The L2C and L5 signals will be used more in the future as new satellites begin joining the constellation, which transmit these signals. A fourth civilian GPS signal, L1C, is expected to be transmitted by the constellation in the 2020's. The software receiver in this work focuses on GPS L1 C/A and is discussed further in detail.

The GPS L1 signal contains a carrier signal that is modulated to a pseudorandom noise (PRN) code and a navigation data message. The PRN code and navigation data are binary sequences of 1's and 0's used to convey information. The GPS L1 carrier transmits from the satellite at 1575.42 MHz, the GPS L1 PRN code transmits from the satellite at 1.023 megachips per second (Mcps), and the GPS L1 navigation data message transmits from the satellite at 50 bits per second (bps). In general, GPS receivers use the PRN code to estimate their position, the carrier to estimate their velocity, and the navigation data message to determine information about the transmitting satellite's position, velocity, and time. The carrier, code, and data message of the GPS L1 C/A signal will be discussed further in their own subsections.

2.1.1 GPS L1 Carrier

The carrier is a sinusoidal function that "carries" information from the orbiting satellite to the receiver on Earth. Specifically, this information is the PRN code and the navigation data message. The signal's name comes from the fact that the carrier transmits within the L-band. The L-Band is a range of frequencies between 1 GigaHertz (GHz) and 2 GHz, used by all GPS signals. This frequency range is used by GPS due to its benefits. The L-Band allows GPS signals to penetrate the Earth's strong atmosphere with minimum signal delay to the receiver. Increasing or decreasing the carrier frequency from the L-Band would cause much larger atmospheric delays or strong signal attenuation through the atmosphere. The PRN code and navigation data are attached to the carrier using Binary Phase Shift Keying (BPSK) modulation, which cause phase shifts in the carrier. A GPS receiver observes a Doppler shift in the satellite signal's carrier frequency which is used by the receiver for velocity determination.

2.1.2 GPS L1 PRN Code

The GPS L1 signal is modulated with the Coarse Acquisition (C/A) PRN code. The C/A code is a sequence of 1's and 0's, known as chips. GPS satellites transmit the C/A code at 1.023 Mcps. The C/A code is a repetitive sequence of chips that is known to the receiver. Although the term PRN may suggest the code is random, it is in-fact a patterned sequence meant to appear random for good correlation properties. The C/A code repeats every 1,023 chips and has a period of 1 millisecond. Every GPS satellite transmits a different C/A code, this allows the receiver to know which GPS satellite is transmitting the received signal. This type of system used by GPS is referred to as Code Division Multiple Access (CDMA). The receiver has knowledge of every satellite's C/A PRN code, which comes from the publicly accessible GPS L1 interface control document (ICD) [44]. The receiver uses the C/A PRN code to synchronize to GPS satellite time and generate ranging measurements for position determination.

The C/A PRN codes are Gold codes, which were developed by Robert Gold. Gold codes are used by GPS to ensure a receiver will never mistake one satellite signal for another. Gold codes also ensure a receiver will never mistake a satellite signal with a time delay (chip delay), within the code period. Gold codes are generated in a simple way where each satellite's PRN code sequence is random enough that it cannot be mistaken for another satellite's PRN code, nor mistaken with the same code sequence that has a chip offset (chip delay or chip advance). This is also true when noise is present on the satellite signal's code.

Figure 2.1 illustrates these special properties of a GPS Gold code. In the figure, two PRN codes are correlated together with differing chip offsets. In blue, two PRN codes from the same satellite are correlated together (autocorrelation). In red, two PRN codes from different satellites are correlated together (crosscorrelation). The autocorrelation is a maximum of 1 when no chip offset occurs and nearly 0 at all other chip offsets. The autocorrelation shows that a replica PRN will not mistake a satellite's PRN with a significant chip delay or chip advance from the true code location. The crosscorrelation is approximately 0 at all chip offsets, showing that PRN codes from different satellites will not provide sufficient correlation for a match. Hence, a receiver will not mistake one satellite for another.

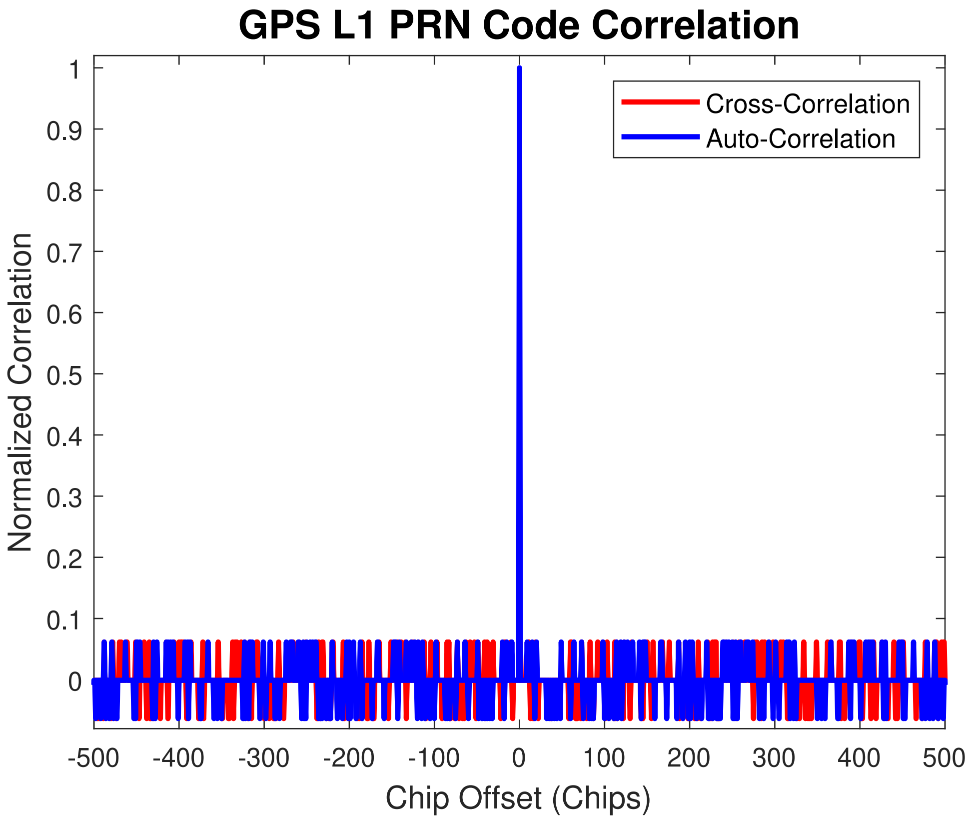

Figure 2.1: Autocorrelation and Crosscorrelation of the GPS L1 C/A PRN Code

Figure 2.2 shows the PRN autocorrelation function from Figure 2.1 zoomed-in near the zero chip offset mark. When the chip offset (chip error) is 0, the autocorrelation is 1. Between chip error magnitudes of 0 to 1 (chip offsets between -1 and 1), the autocorrelation is equal to the chip error magnitude. When the chip error magnitude is greater than 1, the autocorrelation is assumed to be 0. This autocorrelation function concept becomes very important in signal tracking, which is discussed in the next chapter. The concept to note here is that when the chip error between a signal and receiver replica is greater than 1, the GPS receiver cannot effectively track the satellite signal because the code correlation will be nearly 0. Equation (2.1) describes the PRN autocorrelation $R(\epsilon)$ as a function of chip error ϵ.

$$R(\epsilon) = \begin{cases} 1 - |\epsilon| & \epsilon \leq 1 \\ 0 & \epsilon > 1 \end{cases} \tag{2.1}$$

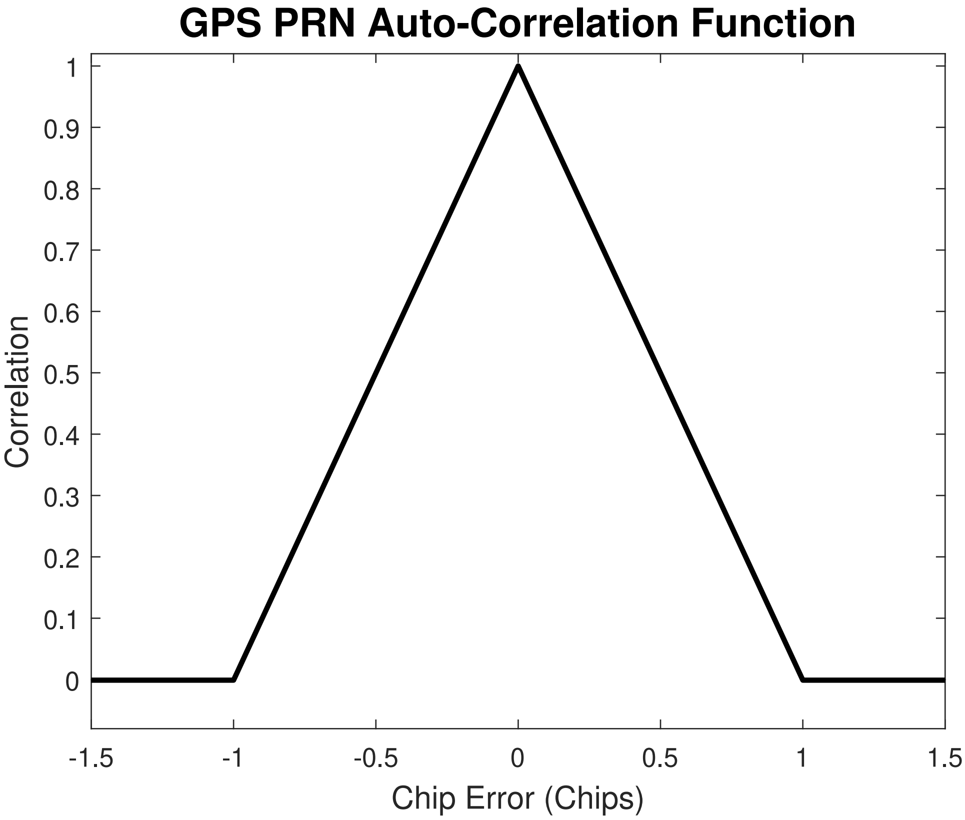

Figure 2.2: Modeled Autocorrelation Function of the GPS L1 C/A PRN Code

The C/A PRN code is generated as a 10-stage Gold code. Figure 2.3 illustrates how the code is generated by the GPS satellite. In the figure, the circular-cross symbols represent modulo-2 summations (even summations become 0 and odd summations become 1). PRN code generation uses two 10-element registers known as Generator 1 (G1) and Generator 2 (G2), these are also known as Polynomial 1 and Polynomial 2. G1 and G2 are both initialized as all 1's in their elements at the beginning of a code generation period. The modulo-2 summation of G1's element three and element ten values are inputted into G1. All the elements in G1 shift to the right, where the value in G1's element ten becomes the output value of G1 for the current epoch. The G2 output is the modulo-2 sum of two of its elements. These elements used for G2's output depends on the GPS satellite. This ensures that every GPS satellite has a unique code sequence. The modulo-2 summation of G2's elements two, three, six, eight, and nine are inputted into G2. The element values are then shifted to the right in G2, where G2's element ten value is thrown out for the current epoch. Finally, the G1 and G2 outputs are modulo-2 summed together to generate the chip value in the PRN code sequence for the current epoch.

After 1,023 iterations, the PRN sequence period is finished, and the generator elements are all reinitialized to 1. Instead of storing the full 1,023 chip sequence for all GPS satellites, receivers generally use this algorithm to develop the PRN code sequences for satellite identification as needed. This allows the receiver to save memory space.

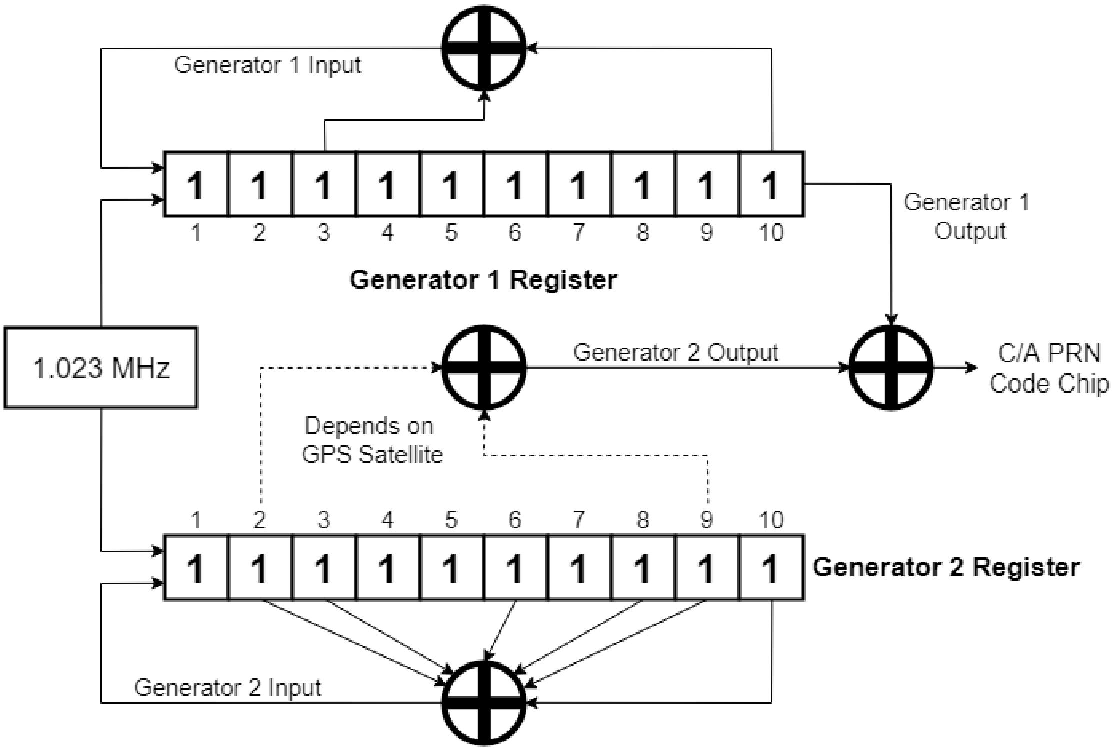

Figure 2.3: Diagram of the Generation of the GPS L1 Coarse Acquisition PRN Code

Since the C/A PRN code is a sequence of binary symbols, the PRN code is a square wave. For a GPS satellite to generate the PRN, the satellite must use a series of several sinusoidal waves operating at different frequencies added together to develop the code. Note that square waves contain discontinuities that can only be generated through a series of sinusoidal functions (Fourier Series). This means the PRN code contains a large frequency content, which is another important property of the code. Figure 2.4 illustrates this property, where the magnitude frequency spectrum of the GPS carrier and the GPS carrier BPSK modulated with the PRN code is shown. Since the GPS carrier is a sinusoid operating at a single frequency, it only has magnitude at that frequency. On Earth, there are several electromagnetic waves propagating through the atmosphere that come from commerical, government, and military communication sources as well as natural sources. If these sources operate at frequencies in the L-band, they can potentially interfere with GPS. There may also be intentional electromagnetic threats to GPS. The carrier is very easy to interfere with because a source transmitting the L1 frequency is all that is needed to disturb the carrier. When the carrier is modulated with the code sequence, the magnitude frequency spectrum appears as a sinc function, where more frequency content is present, and the signal magnitude is spread across these frequencies. The modulation of the code sequence on the carrier drops the magnitude at the L1 frequency and spreads it to additional frequencies nearby. With the modulation, the GPS signal becomes more challenging to interfere with because the source must now transmit a band of frequencies.

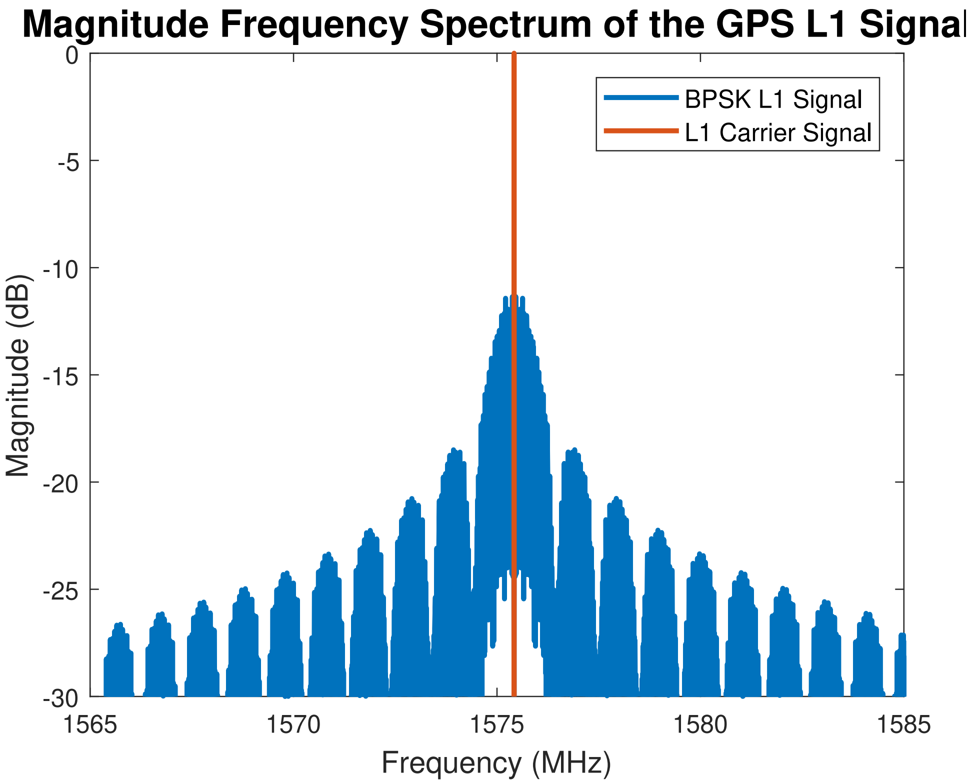

Figure 2.4: Magnitude Frequency Spectrum of the GPS L1 C/A Code

2.1.3 GPS L1 Navigation Data Message

The navigation data message is a binary code modulated to the GPS carrier that provides critical information to the receiver about the transmitting satellite. This information includes the ephemeris, time, clock corrections, atmospheric corrections, and the almanac (information about other satellites in the GPS constellation). The ephemeris is a set of parameters (ephemerides) that allow the receiver to calculate the satellite's position and velocity at a given time. Without this information, a receiver cannot estimate its own position. The satellite transmits the data message at 50 bits per second, much slower than the C/A code and the carrier. This slow transmission rate allows receivers to track the data message through averaging when the signal is weak. One navigation data bit is associated to 20 PRN code periods, where these periods can be used to average out the data bit value.

Figure 2.5 shows the structure of the GPS navigation data for the L1 C/A signal. A complete navigation message from a satellite consists of 25 frames. Each frame consists of 5 subframes, and each subframe consists of 10 words. A word is a sequence of 30 bits that gives information to the receiver in binary form. In the figure, the total navigation message takes 12.5 minutes to be completed, for a total of 37,500 bits per message. The navigation message may seem long, however, portions of the message repeat faster than 12.5 minutes. As stated before, every frame contains 5 subframes. Subframe 1 provides satellite clock correction terms and GPS time information. Subframes 2 and 3 provide ephemeris data for the broadcasting satellite. Subframe 4 provides ionospheric data, coordinated universal time (UTC), and almanac data, and subframe 5 also shows almanac data. Figure 2.6 illustrates the breakdown of the subframes. The information in subframes 1, 2, and 3 is critical for the receiver to generate a navigation solution. These subframes are repeated in every frame with the same data and are updated by the satellite as needed. The receiver reads this critical information every 30 seconds. The data in subframes 4 and 5 is not critical, but helpful to the receiver. The almanac data is a large set of ephemerides that cannot be expressed in two subframes. Subframes 4 and 5 provide almanac data on different satellites in every frame. The full almanac message in subframes 4 and 5 is repeated to the receiver every 12 minutes and 30 seconds.

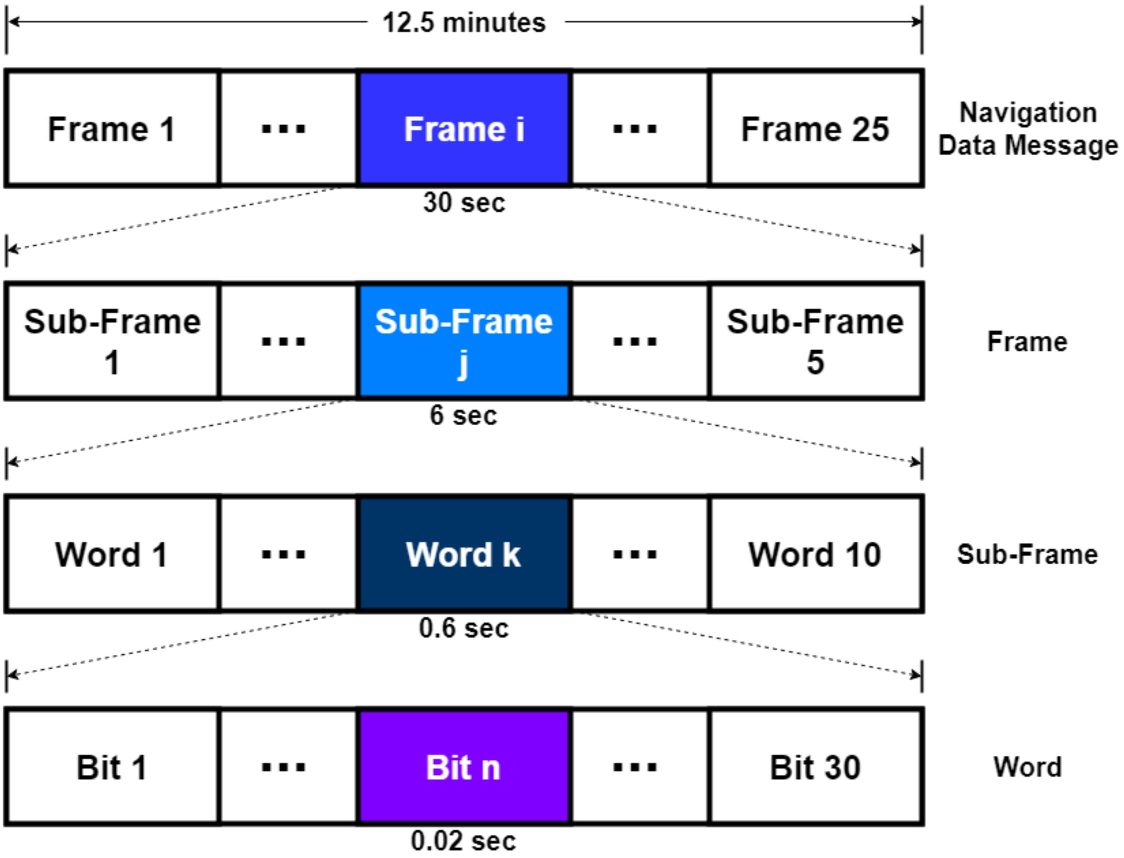

Figure 2.5: Structure of the GPS Navigation Message for the GPS L1 C/A Signal

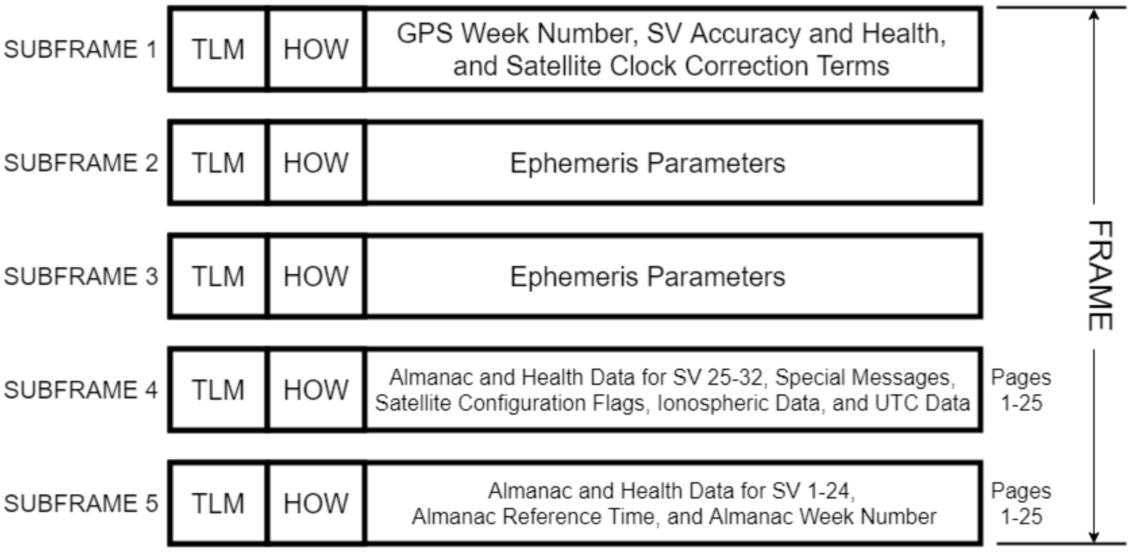

Figure 2.6: Subframe Breakdown in each GPS Navigation Data Message Frame

From Figure 2.6, it is seen that every subframe starts with a telemetry word (TLM) and a handover word (HOW). The TLM is required so the receiver can perform synchronization with the navigation data. The first 8 bits of every TLM contains a preamble that tells the receiver a subframe is beginning. For GPS, the preamble is listed as 10001011 or 01110100 in the case of a 180-degree phase shift in the replica's carrier during signal tracking. The HOW synchronizes the receiver with GPS time by providing the receiver with the time of week (TOW) information. TOW provides the receiver with the transmission time at the rising edge of the first navigation data bit of the preamble for the next subframe in GPS time. The transmission time is given in seconds since the beginning of the week. TOW is crucial for a receiver to accurately navigate. All the GPS satellites transmit their navigation data messages synchronously, within the accuracy of the on-board atomic clocks. Hence, TOW (and the structure of the data message) should be the same at the receiver for all in-view satellite broadcasts at a given synchronous PRN code period. The HOW also provides the receiver with the subframe ID, allowing the receiver to know what subframe it is decoding. Every word in the navigation data uses its last 6 bits for parity. Parity is an additional check for the receiver to validate the navigation message and ensure that a true preamble was located in the data.

2.1.4 GPS L1 Signal Modulation

As stated previously, the C/A code and navigation data are sewn to the carrier through BPSK modulation. Figure 2.7 illustrates the process of modulating the PRN code and navigation data message onto the carrier. First, the C/A code and navigation data are modulo-2 summed together. The summation is then multiplied to the carrier signal, this signal multiplication is the BPSK modulation. At every point where there is a transition from 0 to 1 or 1 to 0 in the binary stream, a 180-degree phase shift occurs in the carrier. Mathematically, the GPS L1 signal transmitted by the satellite $s(t)_{SV_{GPS}}$ can be expressed as shown by Equation (2.2).

$$s(t)_{SV_{GPS}} = \sqrt{2P_{L1}}D(t)C(t)\sin(2\pi f_{L1_{GPS}}t) \tag{2.2}$$

In the equation, P_{L1} is the signal power, $D(t)$ is the navigation data bit value at time t, $C(t)$ is the C/A code chip value, and $f_{L1_{GPS}}$ is the GPS L1 transmission frequency.

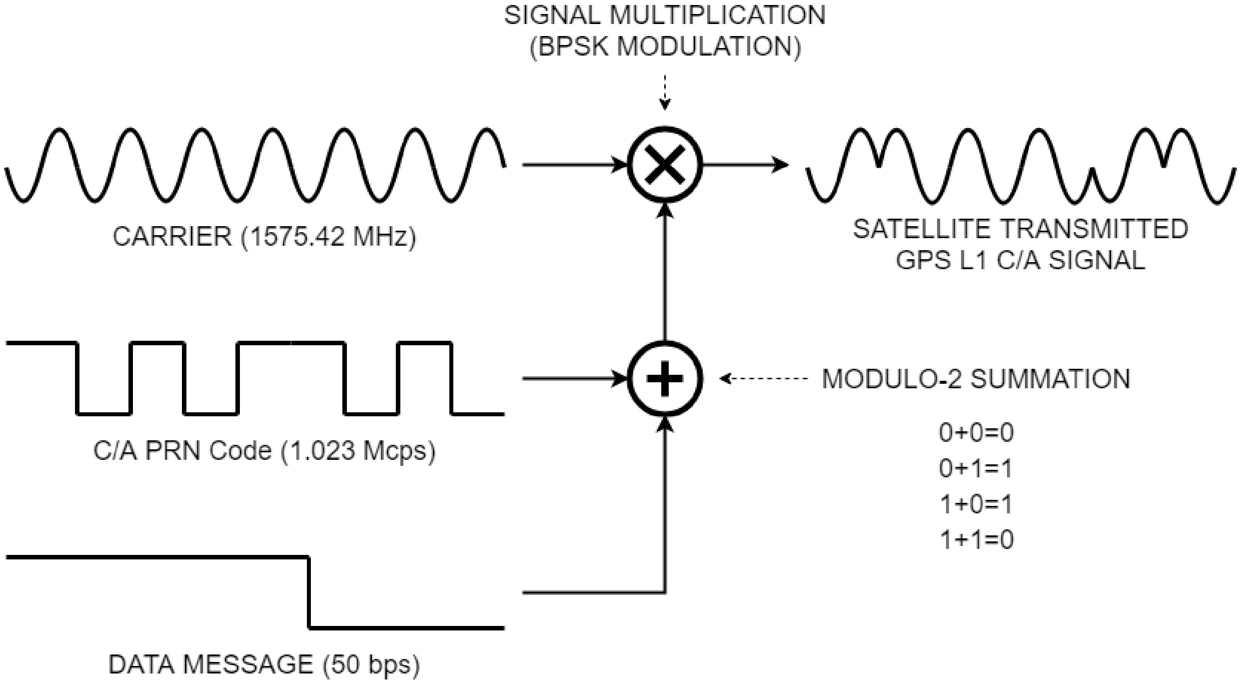

Figure 2.7: BPSK Modulation of the GPS L1 C/A Signal

2.2 The GLONASS L1 Signal

Like GPS, the Global Navigation Satellite System (GLONASS) is a constellation of orbiting satellites that transmit ranging signals to Earth for positioning and timing services. Originally developed by the USSR, GLONASS is now owned and operated by the Russian Federation. As with GPS, GLONASS satellites transmit both civilian and military signals. The GLONASS L1 civilian signal is accessible to the public and the most commonly used signal from the system. The Russian Federation is developing new signals for the GLONASS constellation that will be used in the near future. The receiver in this work focuses on the GLONASS L1 signal. There are many similarities between the GLONASS L1 and GPS L1 C/A signals with a few differences. The GLONASS L1 signal comprises of a carrier, a PRN code, a meander code, and a navigation data message. Each of these signal parts will be discussed further.

2.2.1 GLONASS L1 Carrier

The GLONASS L1 carrier is a sinusoidal function that carries the PRN code, meander sequence, and navigation data message through BPSK modulation. The carrier transmits in the L-Band for the same advantages discussed in the GPS L1 carrier subsection. Unlike GPS, the transmitted carrier frequency is different on each GLONASS satellite. GLONASS L1 uses Frequency Division Multiple Access (FDMA) instead of CDMA to allow receivers to distinguish between different satellites. Since the FDMA system is used for GLONASS L1, the transmitted PRN code sequence on each satellite is the same. New GLONASS signals will begin using the CDMA system like GPS. The main advantage of CDMA over FDMA is that the receiver front-end is not required to take in a larger band of carrier frequencies. CDMA receivers have narrower bandwidths than FDMA receivers, allowing received signals to have less noise. Differing PRN code sequences (CDMA) is also a more robust way of distinguishing satellites than differing carrier frequencies (FDMA). However, after the front-end operations, software receiver operations take less computation for FDMA signals over CDMA signals since only one PRN code sequence needs to be generated.

Equation (2.3) describes how the carrier frequency $f_{L1_{GLO}}$ is transmitted for GLONASS satellite K.

$$f_{L1_{GLO}} = 1602\,\text{MHz} + K(0.5625\,\text{MHz}),\ K = [-7, 6] \tag{2.3}$$

The number K identifies the broadcasting GLONASS satellite. K takes on the values of -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, and 6; which counts to a total of 14 possible broadcasting satellites. The constellation contains more than 14 satellites. Satellite pairs that are always on opposing sides of the Earth that can never both be in-view by the same receiver use the same K value.

It can also be seen that the GLONASS L1 center frequency in Equation (2.3) is offset from the GPS L1 frequency by 26.58 MHz. This offset ensures that the GLONASS L1 and GPS L1 signals do not interfere with each other. This allows receivers to use both signals for navigation and timing services. Figure 2.8 shows the magnitude frequency spectrum of the GLONASS

L1 signal at all 14 different carrier frequencies and the magnitude frequency spectrum of the GPS L1 signal. In the figure, GLONASS L1 is shown in different shades of red to distinguish the main lobes of each transmitted carrier frequency, and GPS L1 is shown in blue. The high magnitude frequency components for each L1 signal are far enough apart that the signals will have very little interference with each other in a receiver front-end. The figure also shows that GLONASS L1 center magnitudes are higher than the GPS L1 center magnitude, and the GPS L1 has a larger frequency spread than a single GLONASS L1 lobe. These properties come from the PRN code characteristics of GPS L1 and GLONASS L1. The GPS L1 code is transmitted at twice the rate the GLONASS L1 code is transmitted, this causes more frequency spread in GPS L1 and hence, a smaller center frequency magnitude when compared to GLONASS L1. The GLONASS L1 code will be discussed further in the next section.

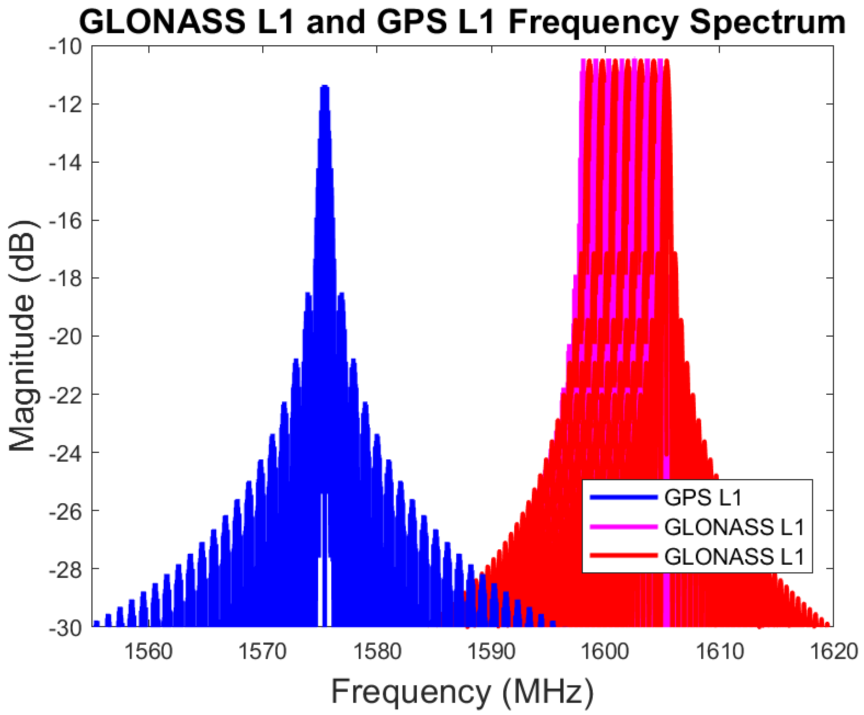

Figure 2.8: Magnitude Frequency Spectrum of the GLONASS L1

2.2.2 GLONASS L1 PRN Code

Although the GLONASS L1 signal uses the transmitted carrier frequency for satellite identification, the PRN code is required for frequency spectrum spreading and ranging purposes.

As stated previously, each satellite transmits the same PRN code. The GLONASS L1 code is transmitted at 511 kHz, with a PRN sequence of 511 chips and a period of 1 millisecond. This PRN code is transmitted at half the chipping rate of the GPS L1 signal, which results in less frequency spread as see in Figure 2.8. The PRN code generation for GLONASS L1 is shown in Figure 2.9. The code is generated using a nine-stage Gold code where all elements are initialized to 1 at the beginning of a code generation period. At each epoch, element seven of the nine-stage register is outputted as the current code chip. Elements five and nine are modulo-2 summed together and inputted into the register, all element values shift to the right where the element nine value is thrown out. After 511 chips have been generated, the elements of the nine-stage register are reinitialized to all 1's, and a new code period begins.

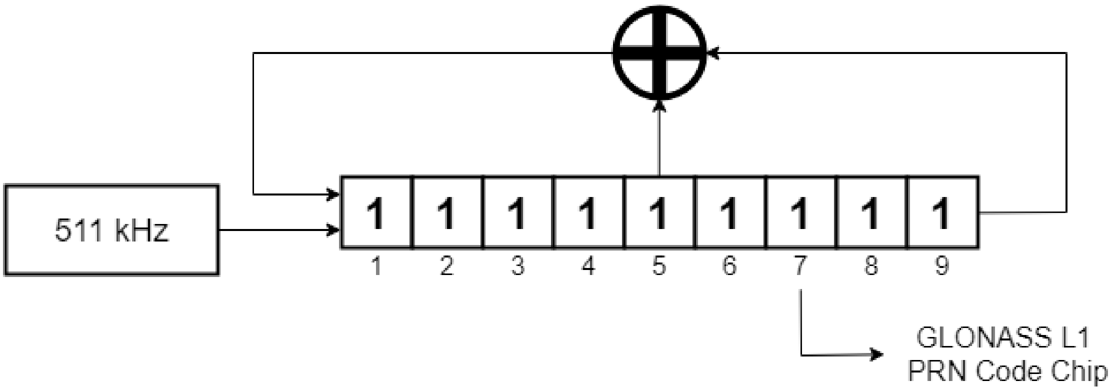

Figure 2.9: Diagram of the GLONASS L1 PRN Code Generator

Since the GLONASS PRN is a Gold code, like GPS, it has the same properties of the C/A code. A receiver will not mistake the GLONASS code with a chip delay or chip advance from the true initial chip starting location. The autocorrelation function $R(\epsilon)$ of the C/A code from Equation (2.1) described in the GPS L1 PRN code subsection, also applies to the GLONASS L1 code. Again, this autocorrelation function is important in receiver signal tracking operations for both the GPS and GLONASS L1 signals.

2.2.3 GLONASS L1 Navigation Data Message and Meander Sequence

Like GPS, the GLONASS L1 navigation data message provides satellite ephemeris, time, satellite clock, and atmospheric information to the receiver. The binary message is transmitted at

50 bps, however, there is also a meander sequence modulated to the signal. This meander sequence is transmitted at 100 cps. The meander sequence is a type of Manchester code, which is a square wave that oscillates between 1 and 0. The GLONASS L1 signal uses the meander sequence to transmit the time mark, which is the equivalent to the GPS preamble. The time mark allows the receiver to synchronize to the data message for decoding. GLONASS transmits the time mark with 10-millisecond chips from the meander sequence instead of 20-millisecond data bits as GPS does with its preamble. The time mark is given by Equation (2.4), which consists of 30 chips, resulting in a time period of 0.3 seconds.

$$\text{GLONASS Time Mark} = 111110001101110101000010010110 \tag{2.4}$$

Note that the time mark can be inverted at the receiver due to 180-degree ambiguity phase shifts from carrier tracking.

Other than when the time mark is being transmitted, the meander sequence is transmitted as a square wave oscillating between chip values of 1 and 0. From a periodic perspective, the meander sequence period contains 200 chips with a time length of two seconds. The first 30 chips of the period is the time mark from Equation (2.4), and the last 170 chips of the period are oscillating 1's and 0's.

The structure of the GLONASS L1 data message is given by Figure 2.10. A data message consists of five frames with a total period of 2.5 minutes. Each frame has a period of 30 seconds and contains 15 strings. The first 5 strings in every frame contains critical information about time, clock corrections, and satellite ephemeris. The time of day (TOD) is transmitted in the first string. The GLONASS TOD is similar to the GPS TOW. The TOD is the satellite transmission time at the rising edge of the first navigation data bit of the next GLONASS frame in seconds since the beginning of the day. Strings 6 through 15 provide almanac and atmospheric data. Different frames provide almanac information about different satellites in the constellation. Like GPS, GLONASS provides information needed for navigation and timing services every 30 seconds in the data message.

Each GLONASS string lasts 2 seconds - the length of the meander sequence period previously described. The first 77 navigation data bits of the string provide unique constellation information as described in the paragraph above. The next 8 data bits are a hamming code used to authenticate the string as part of the GLONASS message, this is similar to the GPS parity. The last part of the string contains 30 meander chips which represent the time mark. Note that data bits have a period of 20 milliseconds, and meander chips have a period of 10 milliseconds. It's also noted that the first 85 bits of the string (information and hamming code) are represented in relative code and modulated to the meander sequence. The first bit in every GLONASS string is an idle chip that is used to start the relative code transformation. As navigation data bits are generated by the satellite, they are transmitted as the modulo-2 summation of the current data bit and previous relative data bit to provide relative code representation. The first ten milliseconds of a data bit are modulated with a meander chip value of 1, and the last 10 milliseconds of a data bit are modulated with a meander chip value of 0. To correctly decode the navigation message, the meander sequence and the relative code representation need to be removed. The time mark at the end of the string is not affected by the meander sequence, nor relative code representation. In the GLONASS ICD [55] and [56], the bit numbers are represented in the opposite direction where the first navigation data bit of the string (idle chip) is listed as bit 85.

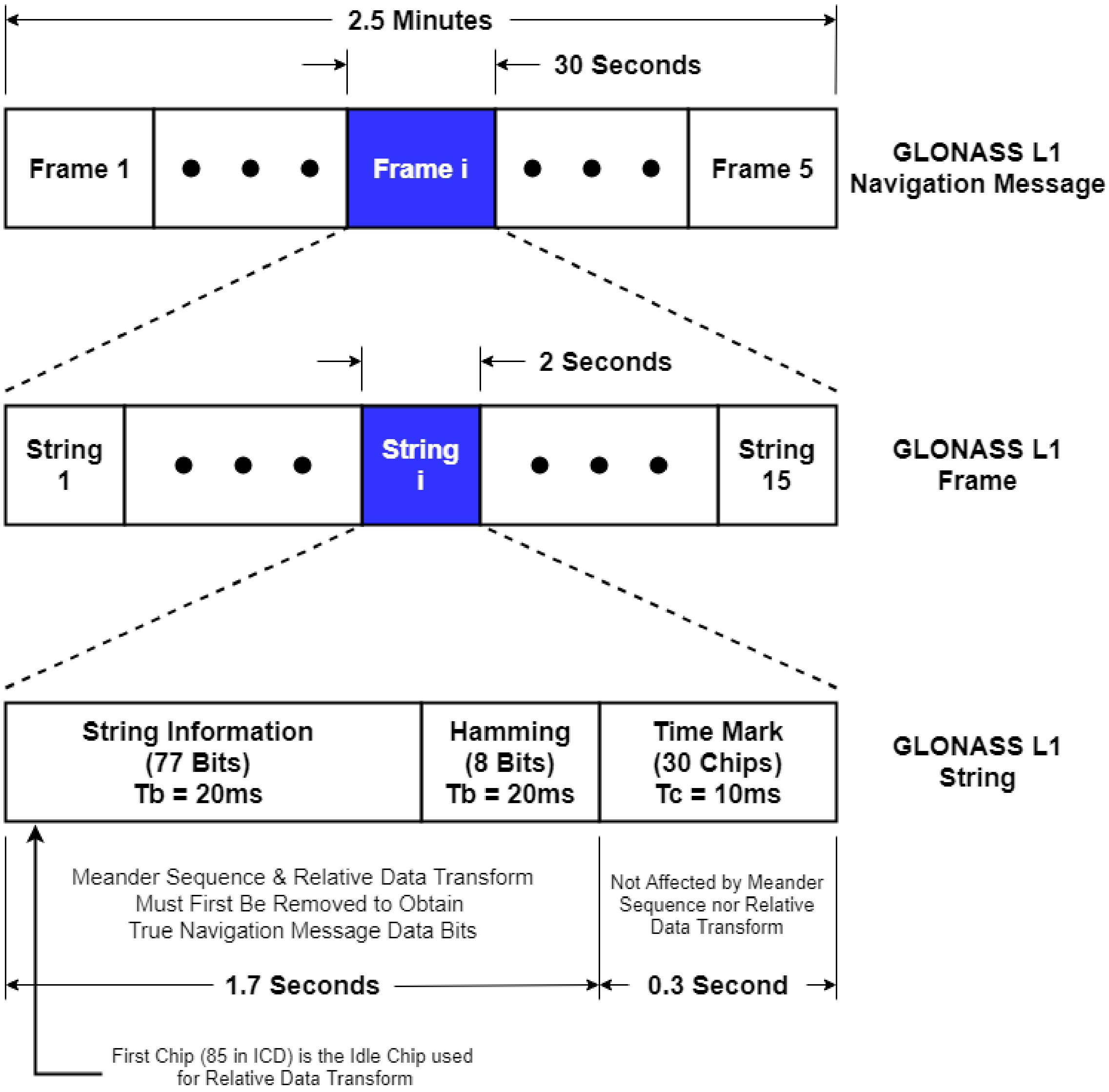

Figure 2.10: Structure of the GLONASS L1 Navigation Data Message

2.2.4 GLONASS L1 Signal Modulation

Like GPS, the GLONASS L1 signal uses BPSK modulation where binary data is represented as 180-degree phase shifts in the carrier. Due to the structure of the GLONASS string in the navigation message, the modulation of the signal is more complicated than GPS L1. Figure 2.11 provides a diagram of the modulation scheme. The navigation data is transformed into relative code through the modulo-2 summation of the current navigation data bit with the previous relative data bit. This relative code transformation starts at the beginning of each GLONASS string with the first data bit that is known as the idle bit. The relative navigation data bits are then modulo-2 summed with the meander sequence. The first half of every relative data bit is modulo-2 summed with a meander chip of value 1, and the second half of every relative data bit is modulo-2 summed with a meander chip of value 0. This sequence of relative data bits modulated to a meander sequence is then organized into the first 1.7 seconds of a GLONASS string. The last 0.3 seconds of the GLONASS string is provided by the time mark. The full string is then modulo-2 summed with the incoming PRN code chips. At this point, the binary data is multiplied to the carrier wave to develop the 180-degree phase shifts. The signal is then transmitted from the satellite to Earth.

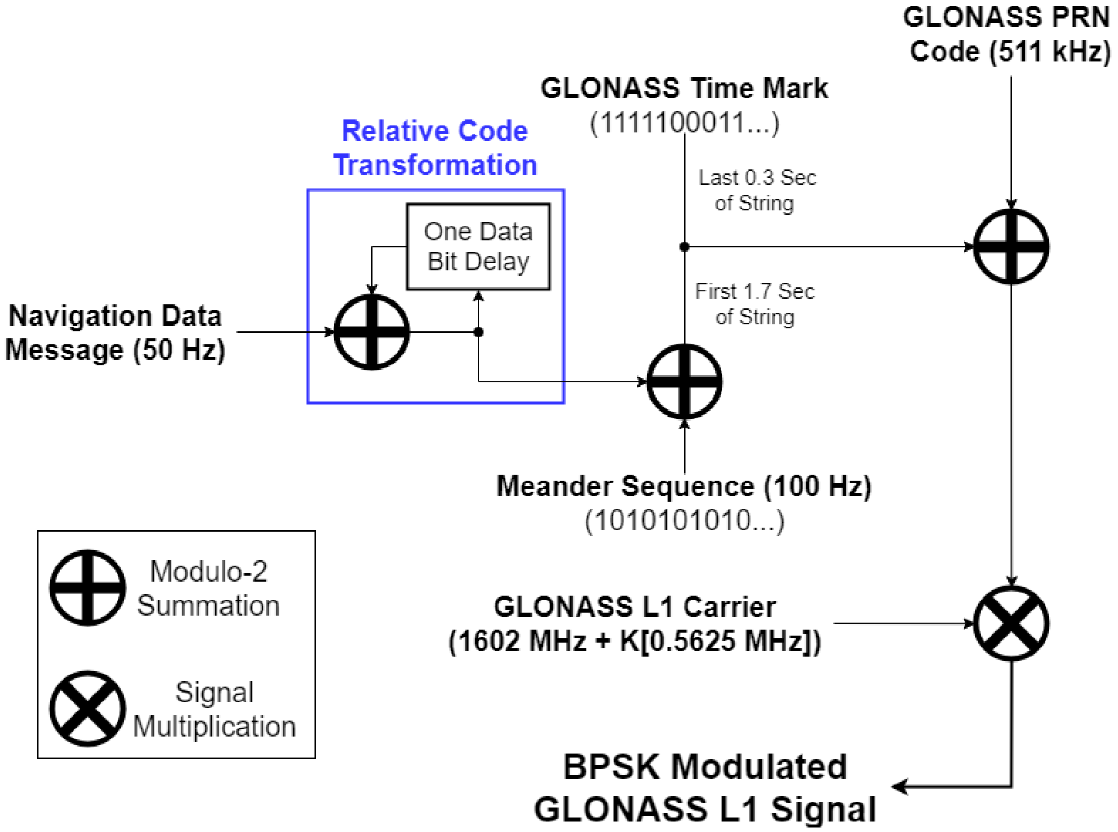

Figure 2.11: GLONASS L1 Signal Modulation

The transmitted GLONASS signal $s(t)_{SV_{GLO}}$ is mathematically modeled by Equation (2.5).

$$s(t)_{SV_{GLO}} = \sqrt{2P_{L1}}D(t)C(t)M(t)\sin(2\pi f_{L1_{GLO}}t) \tag{2.5}$$

The model is very similar to the transmitted GPS signal in Equation (2.2) where P_{L1} is the signal power, $D(t)$ is the navigation data bit value at time t, and $C(t)$ is the PRN code chip value. The GLONASS signal model includes the meander sequence chip value $M(t)$ and the GLONASS L1 transmission frequency $f_{L1_{GLO}}$ which is a function of the GLONASS satellite number K from Equation (2.3).

2.3 Conclusion

This chapter described the design of the GPS L1 C/A and GLONASS L1 signals, which are processed by the software receiver in this work for navigation and timing services. The chapter showed that these signals are BPSK modulated L-Band carriers with a PRN code and a navigation data message. The similarity of the signals allows them to be processed by same receiver algorithms to provide measurements for navigation and timing. The GPS and GLONASS operating frequencies are also far enough apart that both signals can be operated in the same receiver without interference issues. The next chapter discusses the general software receiver operations required to process these GNSS signals for receiver position, velocity, and time estimation.

Chapter 3

GNSS Software Receiver Architecture

Most GNSS receivers used in civil and military applications make use of hardware receivers that acquire, track, and generate navigation solutions from processing satellite signals. Hardware receivers operate in real time, providing position and timing services to many applications that require fast navigation updates. The use of hardware receivers is limited in researching and testing new estimation algorithms, as it takes time, material, and skill to replace electronic circuitry. The academic universities and research labs use GNSS use software receivers as a means of efficient research. Software receivers make use of computer software to process satellite signals into navigation solutions. Processing is generally not operating in real time, which keeps software receivers away from the commercial sector. Tracking algorithms, estimation filters, error mitigation techniques, and other new GNSS receiver research can be implemented into software receivers through coding. The use of sofware receivers in research was made popular by [12] and [26].

Software receivers contain four primary operations, the front-end/mixers, the analog to digital converter (ADC), the computer/microprocessor, and the user interface. Like hardware receivers, the software version uses a hardware front-end, operating in continuous-time, to filter and downconvert the signal. These front-end operations are required so the software receiver can effectively use the signal. The ADC is used to convert the analog satellite signals into sampled digital sequences processed by the computer/microprocessor. The computer/microprocessor uses software-written algorithms to acquire and track satellite signals, as well as generate navigation solutions. The use of the computer/microprocessor is the primary difference between hardware and software receivers. The user interface displays navigation and timing information from the computer/microprocessor. The hardware front-end and software receiver algorithms are discussed further in their own sections.

3.1 Receiver Front-End and ADC

Software receivers rely on hardware front-ends to amplify and filter the satellite signals. A hardware front-end typically consists of a series of amplifiers and filters (Low Noise Amplifiers) and a downconverter. In general, the receiver GNSS satellite signal power at the Earth's surface is approximately 10^{-16} W or -160 dBW, which is below the atmospheric noise floor. Amplifiers raise the satellite signals' power, and filters remove some of the noise on the signals. Some GNSS signals are also transmitted with an orthogonal military signal component that cannot be accessed by civilian receivers. Low pass filters with limited frequency bandwidth in the front-end are used to remove these military signals. Figure 3.1 provides a schematic of a receiver front-end.

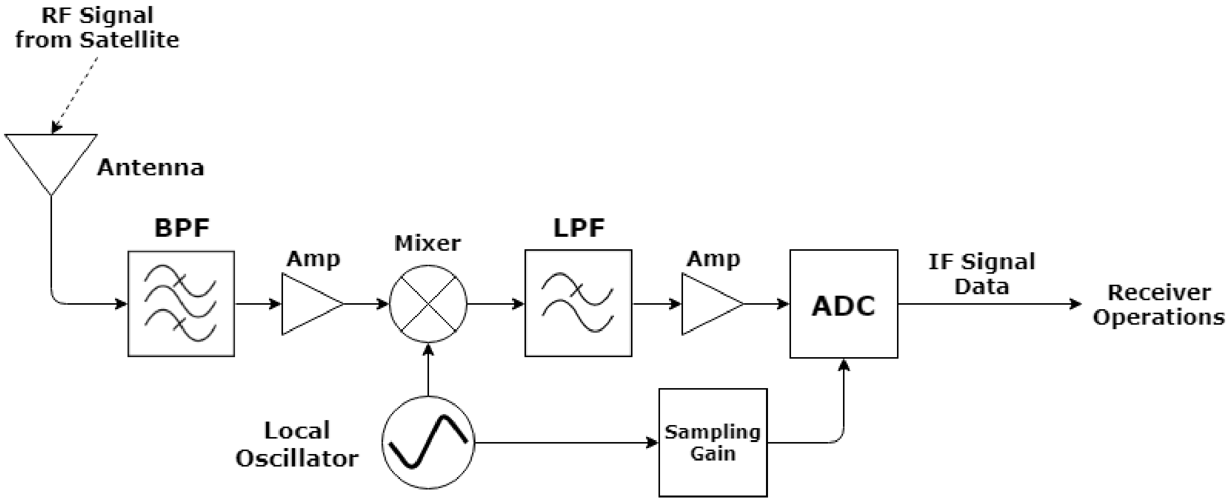

Figure 3.1: Diagram of a Receiver Front-End

A downconverter is used to drop the satellite signal from its transmission frequency to an intermediate frequency (IF) without corrupting or losing any of the signal's information. This process is also known as downmixing. Downconverters are required for receivers since GNSS signals operate in the L-band, which are very high frequencies. Since software receivers use digital samples of the GNSS signals for processing, the signals are required to be up-sampled by at least the Nyquist criterion, which is twice the highest frequency within the signal. Generally, the sampling frequency is four times the signal frequency for GNSS receivers. If the satellite signal is not downconverted to an intermediate frequency, the sampling frequency would be so high that the computer/microprocessor could not process the digital data at an acceptable time rate. Equation (2.2) described a GNSS L1 signal at the point of transmission by the satellite in Chapter 2. Equation (3.1) describes the received L1 signal at the receiver's antenna $s(t)_{antenna}$.

$$s(t)_{antenna} = \sqrt{2P_{L1}}D(t-\tau)C(t-\tau)\sin(2\pi(f_{L1}+f_D)t+\theta) \tag{3.1}$$

In the equation, P_{L1} is the received L1 signal power, $D(t-\tau)$ is the navigation data bit value for a given time, $C(t-\tau)$ is the PRN code chip value at a given time, f_{L1} is the L1 transmitted carrier frequency, f_D is the Doppler frequency induced by dynamics between the satellite and receiver, and θ is the carrier phase of the GNSS signal at the receiver's antenna. For the time variables, t is the current receive time when the signal hits the receiver antenna, and τ is the time delay or transit time between the satellite and receiver. The transmission time is the receive time subtracted from the transit time $(t-\tau)$.

Downmixing starts with multiplying the radio frequency (RF) signal with a signal generated by a local oscillator (LO). In this scenario, the RF signal is the signal received at the antenna with some amplification. The LO signal is a sinusoidal function with some LO frequency f_{LO}. The product of the RF signal and the LO signal produces the addition of two sinusoidal signals, where one sinusoid is the sum of the RF and LO frequencies, and the other sinusoid is the subtraction of the LO frequency from the RF frequency. The process of downmixing ends with using a lowpass filter to remove the high frequency components of the signal. After low-pass filtering, the signal is at an intermediate frequency f_{IF} that is equal to the carrier frequency minus the LO frequency. The signal also operates with a new phase ϕ. Equation (3.2) describes the process of downmixing the L1 RF signal to an IF signal using the front-end's LO signal.

$$s(t)_{IF} = \sqrt{2P_{L1}}D(t-\tau)C(t-\tau)\cos(2\pi(f_{L1}+f_D)t)\sqrt{2P_{LO}}\cos(2\pi f_{LO}t)$$

$$= 2\sqrt{P_{L1}P_{LO}}D(t-\tau)C(t-\tau)[\cos(2\pi(f_{L1}+f_D+f_{LO})t)+ ... \tag{3.2}$$

$$\cos(2\pi(f_{L1}+f_D-f_{LO})t)] = 2\sqrt{P_{L1}P_{LO}}D(t-\tau)C(t-\tau)\cos(2\pi(f_{IF}+f_D)t+\phi)$$

As seen in the equation, crucial information for positioning within the satellite signal is not lost through downmixing.

3.2 Signal Acquisition

After the signals enter the receiver antenna and are converted to a digital sequence that is representative of all in-view satellites for the computer/microprocessor, the software receiver must determine what GNSS satellites are broadcasting to the receiver for each constellation. The receiver discovers what satellites are present in the digital signal using an acquisition algorithm. Acquisition is the process of finding a GNSS satellite signal within digital data through correlating multiple replica signals with the data. A replicated signal is generated by multiplying a replica L1 carrier with a replica PRN code for that GPS/GLONASS satellite. If the correlation between the replica and the signal data is large, then it is probable that a satellite broadcast is present in the signal data.

There are two unknowns when generating a replica GNSS satellite signal, the starting point of the PRN code period and the Doppler frequency shift on the L1 carrier. When a satellite signal initially enters the receiver's antenna, it is unlikely that it will enter at the beginning of a PRN code period. Satellite signal acquisition must determine where a PRN code period begins within the data. Every satellite signal has an induced Doppler frequency associated to its carrier wave due to the net movement between the orbiting satellite and the receiver. This phenomenon is known as the Doppler Effect. Even if the receiver antenna is static, the received signal will still have a Doppler frequency because the transmitting satellite is always moving in its orbit. Along with the PRN code starting point, signal acquisition must also deteremine the carrier Doppler frequency within the data.

Recall from the previous chapter that GPS satellites transmit their own unique C/A code using a PRN number (CDMA). The signal acquisition algorithm must determine what GPS satellites are transmitting to the receiver by searching for each PRN number. GLONASS satellites transmit their own unique carrier frequency using a K-number (FDMA). The signal acquisition algorithm must determine what GLONASS satellites are transmitting to the receiver by searching for each K-number. Overall, the acquisition algorithm provides the receiver with the PRN code starting point, carrier Doppler frequency, and the satellite identification of a broadcast within the signal data. In other words, acquisition is required to search three different parameters for successful determination of a satellite in-view of the receiver. Without these parameters, the receiver cannot move forward towards a navigation solution.

There are not any special equations or algorithms that can accurately and instantaneously estimate the PRN code start point and the Doppler frequency for GNSS signals. For a given GPS/GLONASS satellite, acquisition guesses the code start point and the Doppler frequency, generates a replica for that satellite, and correlates the replica with the digital signal data. Since the code start point and the Doppler frequency are unknown, multiple replicas with various combinations of code start points and Doppler frequencies must be correlated to the signal data. What occurs is a developed search space of correlation values that are associated to these various replicas for a given satellite ID.

Figure 3.2 shows acquisition search spaces where a satellite signal is present (left) and where a satellite signal is not present (right). The search spaces show correlation values with one axis representing code starting points and the other axis representing Doppler frequencies. The peak in the left search space shows high correlation between the replica and signal data at an acquired code and Doppler guess. The right search space appears noisy at all code and Doppler guesses, indicating a satellite signal is not present in the digital data.

3.2.1 Serial Search Algorithm

The most basic signal acquisition algorithm is the serial search algorithm, where a search space is generated by correlating signal data to several replicas with different combinations of code and carrier parameters. Code start points are guessed in increments of chips. Recall from Chapter 2 that the GPS L1 signal contains 1,023 chips in a code period. This results in a code search space size of 1,023 ranging from 0 to 1,022 chips. The GLONASS L1 signal

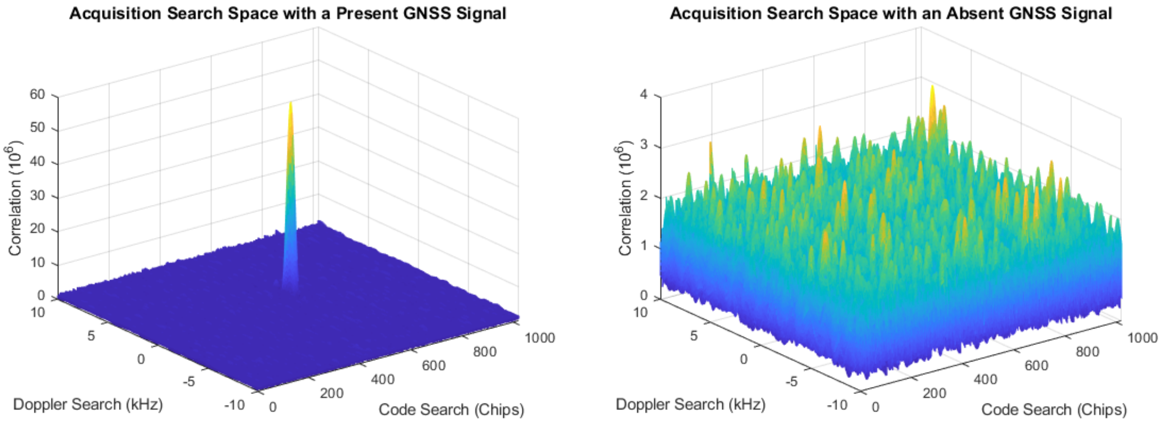

Figure 3.2: Acquisition Search Spaces with and without a GNSS Signal

contains 511 chips in a code period, which results in a code search space size of 511. Doppler frequencies are searched between -5 kHz and 5 kHz for a static receiver and -10 kHz to 10 kHz for a dynamic receiver. Typically, the Doppler frequency is searched in bins of 500 Hz for both GPS and GLONASS. A dynamic GPS receiver will have an acquisition search of 1023 chips x 41 bins, and a dynamic GLONASS receiver will have an acquisition search space of 511 chips x 41 bins.

Acquisition looks for rough estimates of satellite signal parameters; the carrier phase of the signal is not searched for. Without knowledge of the carrier phase, correlation between the receiver replicas and signal data will be very poor. To counteract the unknown carrier phase, two receiver replicas are correlated to the data for a given set of signal parameters. These replicas are known as the in-phase and quadrature. The in-phase replica uses a sine function for the carrier, and the quadrature replica uses a cosine function for the carrier. The sine and cosine functions are 90 degrees apart in phase, which allows the carrier phase to be neglected in the search space as long as both replicas are used in the correlation. Essentially, by using the sine and cosine functions in the replicas, carrier phase correlation will show up in one or both of the in-phase and quadrature correlators. Individual in-phase and quadrature correlators are generated by correlating the signal data to the in-phase replica and the quadrature replica. The calculated correlation placed in the search space is the sum of the magnitudes of the in-phase and quadrature correlator values. Calculating the magnitude of the in-phase and quadrature correlators also removes the navigation data bit ambiguity that can occur in signal acquisition. In other words, the navigation data bit value can make correlation negative. Calculating the magnitude of the in-phase and quadrature correlators ensures all search space values are positive.

Figure 3.3 provides a diagram of the serial search acquisition algorithm. Serial search acquisition is great for beginners to GNSS receivers because of the simplicity of the algorithm. However, the serial search algorithm is not typically used in receivers due to its large computational time. The serial algorithm is required to generate thousands of correlators for each searched satellite signal. More complex acquisition algorithms such as the parallel frequency and parallel code algorithms take advantage of the frequency domain to parallelize either the Doppler frequency search or the code start point search. These algorithms are discussed in more detail next.

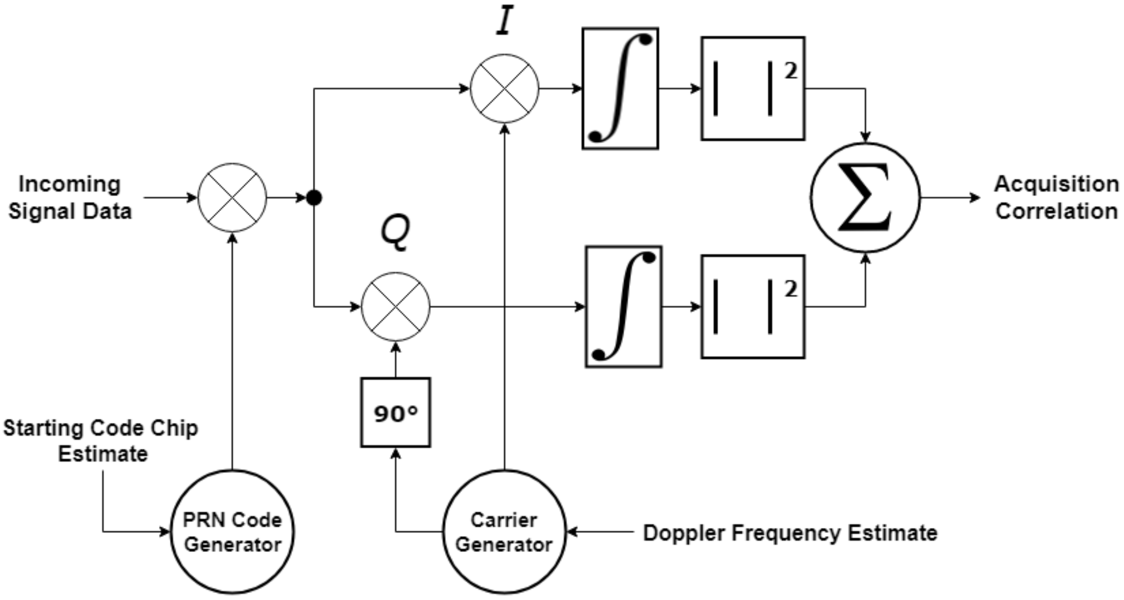

Figure 3.3: Diagram of the Serial Search Acquisition Algorithm

3.2.2 Parallel Frequency Search Acquisition

To reduce the computational time for signal acquisition, the parallel frequency search parallelizes the Doppler frequency search by transforming the product of the satellite signal data and the replica PRN code from the time domain to the frequency domain. This allows the frequency search to be performed in one calculation. Figure 3.4 provides a diagram of the parallel frequency search algorithm. Since the signal data is discrete, a Discrete Fourier Transform (DFT) is used for transformation from the time domain to the frequency domain. The DFT can be replaced with a faster algorithm, the Fast Fourier Transform (FFT). By parallelizing the Doppler frequency search, only the code start point needs to be searched. Hence, PRN codes starting at different chip locations are generated and correlated with the satellite signal data. The correlation signals are then transformed to the frequency domain using the FFT. When an FFT is applied to a real time domain signal, the output is an array of complex numbers. Recall that the serial search algorithm used an in-phase (sine) replica and a quadrature (cosine) replica to account for the carrier phase uncertainty. In the parallel frequency search, the complex numbers represent the in-phase and quadrature replicas. The real component of a complex number represents the in-phase replica, and the imaginary component represents the quadrature replica. The complex numbers are transformed into real, positive magnitudes in the acquisition search space to represent both the in-phase and quadrature correlators.

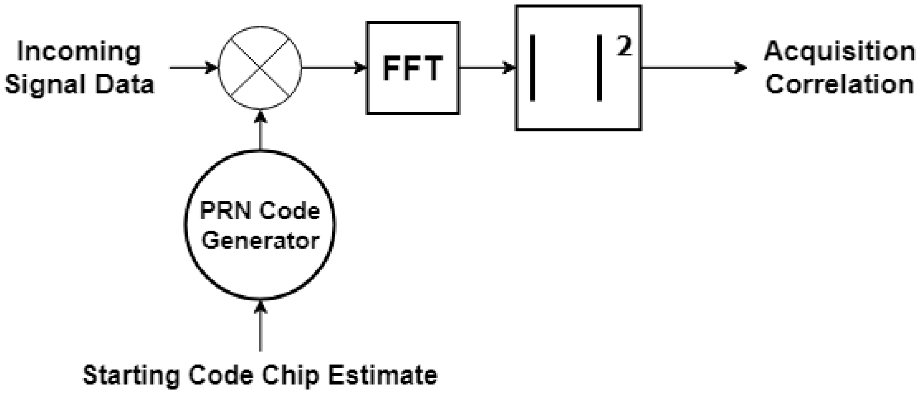

Figure 3.4: Diagram of the Parallel Frequency Search Acquisition Algorithm

A disadvantage of the parallel frequency search over other acquisition algorithms is the Doppler frequency resolution that comes from the output of the algorithm. The algorithm's frequency coverage depends on the sampling rate, and its frequency resolution depends on the time length of the signal data. The sampling rate is typically high enough to cover the span of frequencies the satellite signal's carrier Doppler may take on. Frequency resolution in the FFT is equivalent to the inverse of the time length of the signal data used in acquisition ($f_r = 1/t$). The time length of a code period for GPS and GLONASS is $t = 0.001$ seconds, which results in a frequency resolution of $f_r = 1000$ Hz. Recall that the Doppler frequency search in the serial algorithm was performed in 500 Hz intervals. If the parallel frequency search algorithm is used, an additional frequency search on top of the original search will be required to lower the frequency resolution to 500 Hz. Additional PRN code periods can be used to perform acquisition with a longer integration period, which will result in more frequency resolution.

3.2.3 Parallel Code Search Acquisition

Like the parallel frequency algorithm, the parallel code algorithm uses the frequency domain to reduce the computational time for signal acquisition. However, the parallel code parallelizes the PRN code starting point search instead of the carrier Doppler frequency search. Figure 3.5 provides a diagram of the parallel code search algorithm. The product of the signal data and the carrier replica is transformed from the time domain to the frequency domain using an FFT. The carrier replica uses in-phase and quadrature replicas as the serial search algorithm did, to resolve the carrier phase ambiguity issue. The in-phase replica represents the real component of the signal, and the quadrature replica represents the imaginary component of the signal. This transformed complex signal is then multiplied by the complex conjugate of the FFT of the PRN code replica to generate a combined signal that includes the signal data, the carrier replicas, and the PRN code replica in the frequency domain. The combined signal is transformed back to the to the time domain using the Inverse Discrete Fourier Transform (IDFT). The IDFT can be replaced with a faster algorithm, the Inverse Fast Fourier Transform (IFFT). When the combined signal is transformed back to the time domain, the signal is an array of complex numbers. The complex numbers are squared to represent correlation magnitudes in the acquisition search space.

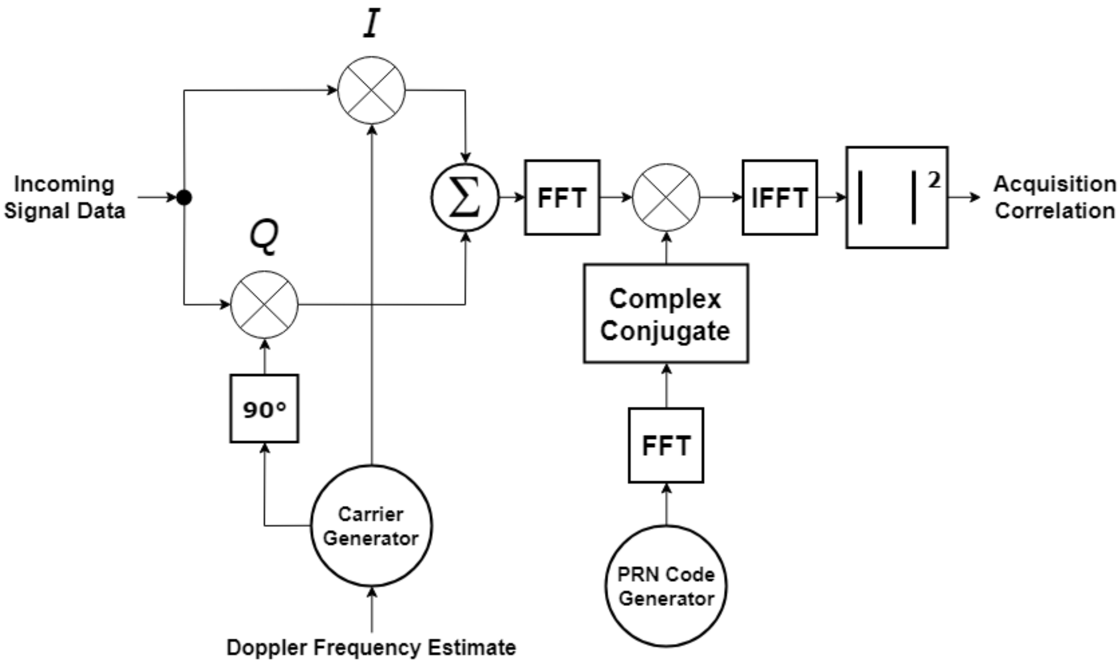

Figure 3.5: Diagram of the Parallel Code Search Acquisition Algorithm

Unlike the parallel frequency algorithm, the parallel code algorithm provides additional resolution in the code starting point search over the serial search algorithm. The code starting location has sample point resolution in the parallel code algorithm instead of chip resolution as in the serial search and parallel frequency algorithms. In general, most GPS and GLONASS software receivers have between 10 to 20 samples to represent a single code chip. Hence, the parallel code algorithm provides a resolution around 1/10 to 1/20 of a PRN code chip. For this thesis, the parallel code search algorithm is chosen for signal acquisition. Since the algorithm is computationally fast, the Doppler frequency bin search is reduced to 100 Hz for additional frequency resolution.

3.2.4 Acquisition Integration Period

Generally, for GPS and GLONASS, the correlators for an acquisition search space are generated using 1 millisecond of signal data because their PRN codes repeat every millisecond. In cases where the signal data is noisy, an integration period may need to be longer to increase correlation, but this period is limited by the structure of the GNSS signals. Recall from the previous chapter that the GPS signal's transmitted code period is 1 millisecond, and its data message is transmitted at 50 bps. This results in 20 code periods for each data bit. Like GPS, the GLONASS signal's transmitted code period is 1 millisecond, and its data message is transmitted at 50 bps, which also pertains to 20 code periods for each data bit. However, the GLONASS signal also contains a meander sequence transmitted at 100 cps, which results in 10 code periods for each meander chip. During signal acquisition, the navigation data (and meander sequence) is generally unknown, and if a data bit transition (or meander chip transition) occurs during an integration period, correlation between the replica and the signal data is destroyed by the integration process. When a navigation data bit or meander chip transition occurs, the signal data becomes inverted (flipped), correlation changes sign, and integration effectively begins subtracting instead of adding, destroying the correlator magnitude.

For noisy signal conditions, an integration period of 10 milliseconds is used for GPS signal acquisition. However, a data bit transition can still occur during 10 milliseconds. To counteract this problem, another 10-millisecond acquisition is conducted on the next corresponding satellite signal data. Because of the GPS signal structure, a data bit transition cannot occur in both 10 millisecond periods, so one of the periods is guaranteed to provide reliable acquisition results. The next navigation data bit in the GPS signal may have the same value as the current data bit, allowing acqusition to be performed with a 20-millisecond integration period. To perform acquisition with integration periods greater than 20 milliseconds, the data message generally needs to be known.

For longer integration periods in GLONASS, the integration period is limited to 5 milliseconds due to the meander sequence. Although the meander sequence provides advantages to the GLONASS signal, this is a disadvantage of modulating a Manchester code onto the signal. Similar to GPS, a meander chip transition cannot occur in two consecutive 5-millisecond integration periods. If the data message is known, the meander sequence can be appropriately lined up, and longer integration periods can be peformed on a GLONASS satellite signal.

3.3 Signal Tracking

Signal tracking is the process of developing accurate replicas of the acquired satellite signals over time through signal correlation and control loops. Signal tracking is the guts of any GNSS receiver. The tracking process extracts code signal measurements (pseudoranges) and carrier signal measurements (pseudorange rates) required to calculate the receiver's position and velocity. Tracking also extracts the satellite navigation data message that is used to decode the ephemeris. The ephemeris is necessary to calculate satellite position and velocity. Without accurate signal tracking, there is not any way to position with GNSS.

In signal acquisition, estimates of an acquired signal's received carrier frequency and code starting point are determined. Since GNSS satellites are always moving in orbit, the received carrier frequency is always changing due to the Doppler effect. This Doppler effect is also induced on the PRN code frequency. Although the Doppler effect is small on the PRN code frequency, it must still be tracked for precision. Specifically, signal tracking estimates a GNSS signal's received carrier frequency and received code frequency. To ensure the receiver's code replica is accurate, the fractional code phase must also be tracked. Tracking the fractional carrier phase is optional; the carrier phase can be tracked to obtain the satellite signal's navigation data message. The fractional carrier and code phases are maintained within the replica from the estimated carrier and code frequencies.

To track the carrier and code frequencies, two feedback control loops are used. The feedback control loops correlate the signal data with the receiver replica, determine the carrier and code errors, and update the replica with filtered errors to better resemble the satellite signal within the data. Ideally, the feedback control loops drive the carrier and code errors to zero, so the receiver replica is a perfect representation of the in-view satellite signal. The feedback control loops used in signal tracking are known as the Phase Lock Loop (PLL) and the Delay Lock Loop (DLL). The PLL tracks the signal's carrier frequency, and the DLL tracks the signal's code frequency.

3.3.1 The Phase Lock Loop

In general, the PLL tracks the frequency and phase of a sinusoidal signal. The PLL calculates a phase error between the true signal and the replica, the phase error is then low-pass filtered and applied to the replica. PLL's are used in a variety of signal processing applications. In GNSS receivers, the PLL tracks a satellite signal's carrier frequency and carrier phase. Figure

3.6 provides a diagram of the PLL. The incoming signal data from the receiver's front-end is multiplied to the receiver's PRN code replica, which is assumed to be perfectly matched with the entering satellite's PRN code. The signal data and code replica are then multiplied to two carrier replicas, an in-phase replica (sine) and a quadrature replica (cosine). The signal data combined with the code replica and carrier replicas are integrated (summed) to develop two correlators, an in-phase correlator and a quadrature correlator. Recall, in signal acquisition that an in-phase correlator and a quadrature correlator were used because the carrier phase was unknown. The PLL is attempting to estimate the carrier phase of the received signal.

The correlators are a representation of how well the receiver's replica carrier matches the satellite's carrier within the data. When the PLL is correctly tracking the carrier, the in-phase correlator will have a high magnitude, and the quadrature correlator will have a low magnitude. During accurate steady-state PLL operation, the in-phase replica will be a representation of the GNSS carrier entering the receiver. The two correlators are then used to calculate a carrier discriminator that gives a representation of the total carrier phase error (carrier frequency error and carrier phase error) between the replica and the signal data. However, when the carrier frequency error and the integration time period of correlation are small, the carrier discriminator represents the carrier phase error. Since there is noise on the signal data, the carrier discriminator will also have noise. A loop filter is applied to the carrier discriminator to remove noise. The filtered carrier discriminator is then used to update the replica's carrier frequency. After the update, the PLL restarts by developing the in-phase and quadrature correlators for the next set of signal data.

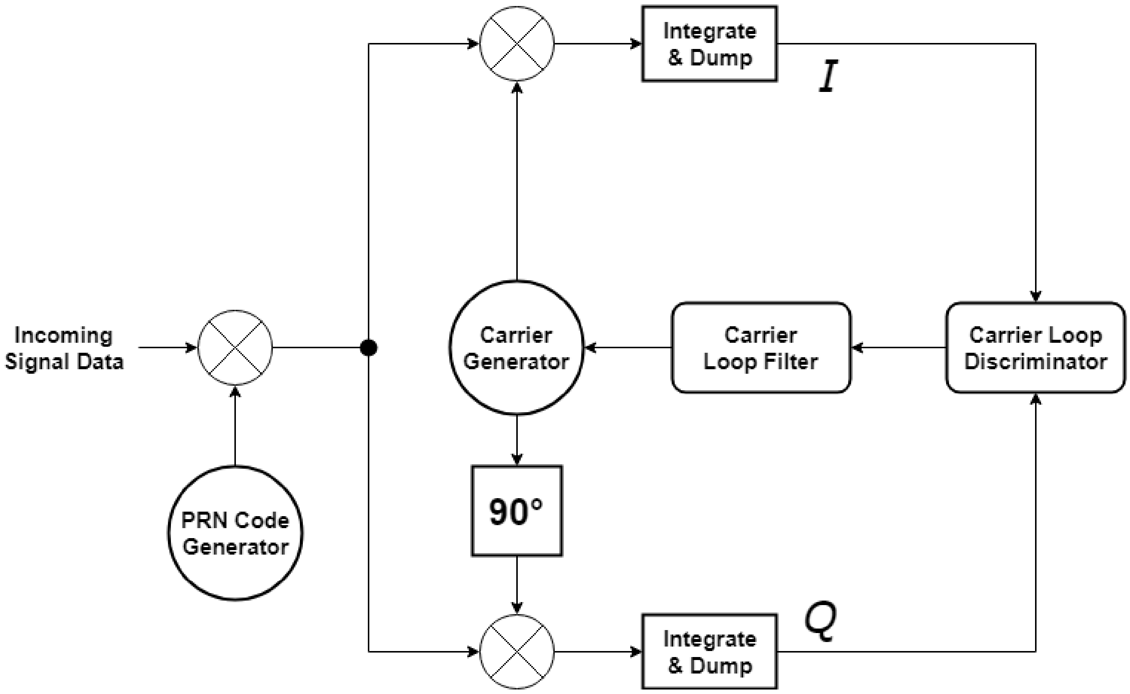

Figure 3.6: Diagram of a Phase Lock Loop used for Carrier Tracking

Since the PLL provides updates to the replica based on carrier phase errors, the PLL cannot lock onto the receiver carrier if the initial frequency error is large. Depending on the system order, the PLL can generally track a signal if the initial carrier frequency error is within 0 to 300 Hz. In cases where there is significant noise on the signal, the phase error can become distorted, which makes carrier tracking with the PLL very challenging.

3.3.2 The Delay Lock Loop

The DLL is a form of the PLL that is used to track recurring patterned codes and clock signals. Codes are generally used for timekeeping purposes, hence the fractional code phase must always be tracked to ensure there is not any time error. Similar to the PLL, the DLL determines a code phase error between the true and replica codes. The code phase error is then low-pass filtered and applied to the replica through a frequency update. In GNSS, the DLL tracks a satellite signal's code frequency and code phase. Figure 3.7 provides a diagram of the DLL. The incoming signal data from the receiver's front-end is multiplied to the receiver's carrier replica, which is assumed to be perfectly matched with the satellite's carrier. The signal data and carrier replica are then multiplied to three different PRN code replicas. These code replicas are the early, prompt, and late PRN's. The early code has a negative chip spacing from the prompt code, and the late code has a positive chip spacing from the prompt code. The chip spacing is generally between half to one chip due to the autocorrelation properties of PRN codes. The combined signal data, carrier replica, and code replica sequences are then integrated (summed) to generate early, prompt, and late correlators. The DLL requires early, prompt, and late correlators to calculate its code phase error.

The correlators are a representation of how well the receiver's replica code matches the satellite's PRN code within the data. When the DLL is correctly tracking the code, the prompt correlator will be high, and the early and late correlators will nearly be equivalent. When the DLL's code replica is ahead of the signal's code, the early correlator will be higher than the late correlator. When the DLL's code replica is behind the signal's code, the late correlator will be higher than the early correlator. The three correlators are then used to calculate a code discriminator that gives a representation of the code phase alignment error between the replica and the signal data. The code discriminator goes through a loop filter to remove noise. The filtered code discriminator is then used to update the code replica's frequency and phase. The DLL then restarts by developing the early, prompt, and late correlators for the next set of signal data.

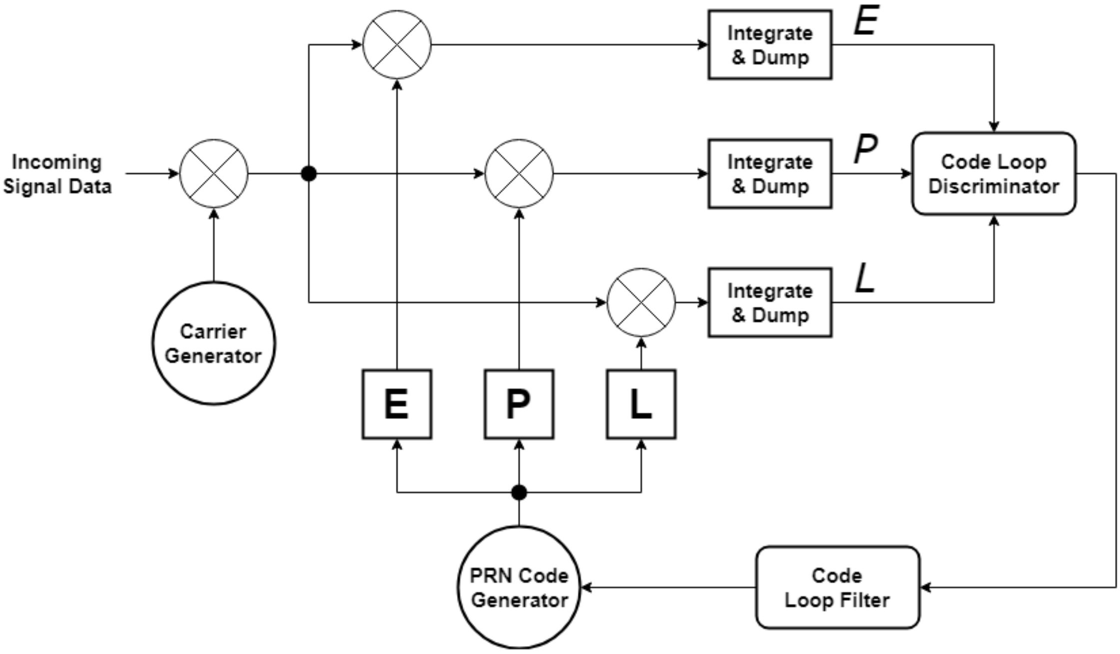

Figure 3.7: Diagram of a Delay Lock Loop used for Code Tracking

Similar to the PLL, since the DLL uses a code phase error for replica updates, the DLL is limited in tracking capability based on code frequency error. However, this is usually not an issue in GNSS since code frequencies (and their Doppler effects) are small. The received code frequency will be near the satellite transmitted code frequency. Based on the PRN autocorrelation function $R(\epsilon)$ from Equation (2.1), the DLL can accurately track a satellite signal's PRN code if the code phase alignment error is within 1 chip. Once the code phase alignment error is greater than a chip, the DLL cannot track the signal, and acquisition is required to re-track the satellite. Although the autocorrelation properties causes this issue, the autocorrelation function allows the PRN code to be tracked in the prescence of significant noise. The autocorrelation function allows the DLL to track at noise levels the PLL could not because the PRN code has a much longer period than the the carrier sinusoid.

3.3.3 GNSS Scalar Tracking

The discussions on the PLL and DLL in the previous subsections provided a simplified representation where the code replica was assumed perfect for the PLL, and the carrier replica was assumed perfect for the DLL. In a real GNSS receiver, neither the code nor carrier replica will be perfect. For the PLL and DLL to effectively track a satellite signal, the tracking loops work together to generate an accurate replica. The PLL relies on the DLL's code replica to develop the carrier replica. The DLL relies on the PLL's carrier replica to develop the code replica. Due to the separation control principle, the loop dynamics of the PLL and DLL can be designed individually and operate together to minimize the carrier and code tracking errors. Figure 3.8 provides a diagram of a combined PLL and DLL tracking loop, this is the typical tracking loop seen in GNSS receivers that is known as scalar tracking.

The PLL used two correlators for carrier tracking, the in-phase and quadrature prompt correlators. The DLL used three correlators for code tracking; the early, prompt, and late correlators. When the PLL and DLL are combined, the signal tracking loop develops six correlators, the in-phase early correlator IE, the in-phase prompt correlator IP, the in-phase late correlator IL, the quadrature late correlator QL, the quadrature prompt correlator QP, and the quadrature early correlator QE. When the receiver is successfully tracking the signal data, the in-phase prompt replica will be an accurate replica of the satellite signal within the data.

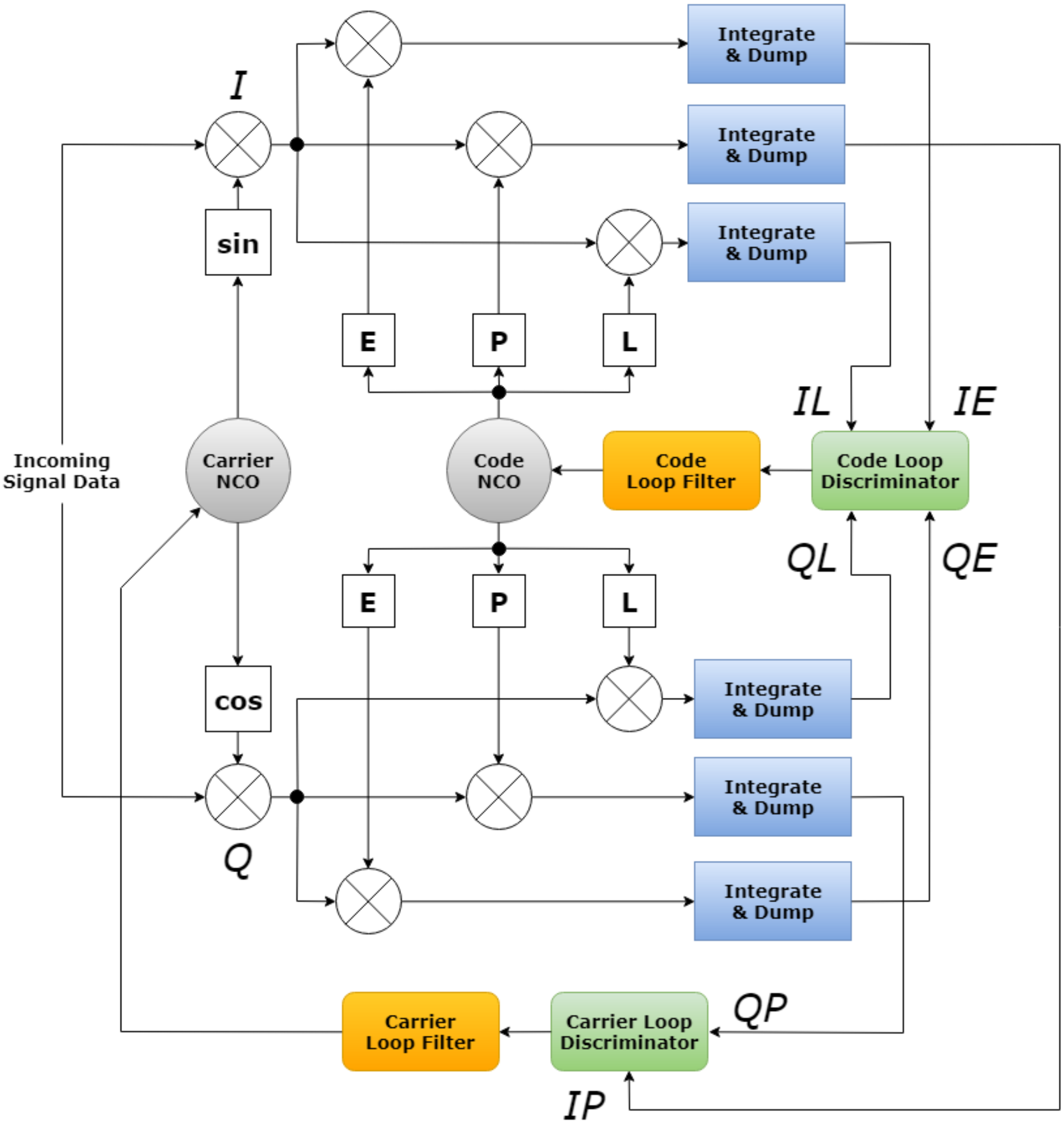

Figure 3.8: Diagram of a Combined PLL and DLL (Scalar Tracking)

Equation (3.3) provides a model of the signal tracking correlators.

$$\begin{aligned}
IE(k) &= AR(\epsilon+\tau)D(k)\cos(\pi f_{err}T+\theta_{err})+\eta_{IE(k)} \\
IP(k) &= AR(\epsilon)D(k)\cos(\pi f_{err}T+\theta_{err})+\eta_{IP(k)} \\
IL(k) &= AR(\epsilon-\tau)D(k)\cos(\pi f_{err}T+\theta_{err})+\eta_{IL(k)} \\
QE(k) &= AR(\epsilon+\tau)D(k)\sin(\pi f_{err}T+\theta_{err})+\eta_{QE(k)} \\
QP(k) &= AR(\epsilon)D(k)\sin(\pi f_{err}T+\theta_{err})+\eta_{QP(k)}
\end{aligned} \tag{3.3}$$

$$\begin{aligned} QL(k) &= AR(\epsilon - \tau)D(k)\sin(\pi f_{err}T + \theta_{err}) + \eta_{QL(k)} \\ \eta &= N(0,1) \end{aligned}$$

In the model, A is the observed correlator amplitude seen in tracking, $D(k)$ is the navigation data bit value for the integration period k, $R(\epsilon)$ is the PRN autocorrelation function with a PRN chip alignment error ϵ between the signal and the replica, f_{err} is the carrier frequency error between the signal and the replica, θ_{err} is the carrier phase error between the signal and the replica, and T is the signal tracking integration period. The variable η is a noise term for the correlators. In this model, the noise is zero-mean with unit variance using a Gaussian distribution. A decrease in the code error ϵ increases the value of all correlators. A decrease in the carrier errors f_{err} and θ_{err} increases the in-phase magnitudes and decreases the quadrature magnitudes. When the signal tracking errors are small, the in-phase prompt correlator $IP(k)$ takes on the value of the navigation data bit $D(k)$ with amplification and noise. From this, the receiver's replica must be code-locked, carrier frequency-locked, and carrier phase-locked to obtain the satellite signal's navigation data message. During accurate signal tracking, the quadrature prompt correlator $QP(k)$ represents the noise on the correlators. The early and late correlators are scaled versions of their prompt counterparts where the scaling is dependent on the PRN chip spacing τ. In this work, the chip spacing is $\tau = 0.5$ chip.

The observed correlator amplitude A is expressed in Equation (3.4) as a function of the satellite signal's carrier-to-noise power density C/N_o ratio, the tracking integration period T, and the carrier frequency error f_{err}.

$$A = \sqrt{2T\frac{C}{N_o}}\frac{\sin(\pi f_{err}T)}{(\pi f_{err}T)} \tag{3.4}$$

The C/$\mathrm{N_O}$ ratio quantifies how much signal to noise power there is on a received signal. A signal with a low C/$\mathrm{N_O}$ will have a low correlator amplitude within the receiver's tracking operation. The C/$\mathrm{N_O}$ ratio is a signal environment and front-end issue that cannot be altered by the software receiver algorithms. The observed correlator amplitude increases with longer integration periods due to more correlation. The observed amplitude also depends on the sinc function with the carrier frequency error. When the carrier frequency error is $f_{err} = 0$, the sinc function evaluates to 1. The observed correlator amplitude is a function of carrier frequency tracking accuracy, which is why A is described as the *observed* amplitude. A high C/N$_0$ satellite signal can have a low amplitude in the receiver if the carrier is poorly estimated.

Figure 3.9 shows correlator values over time during accurate signal tracking with carrier phase lock. Prompt correlators are shown on the left, while early and late correlators are shown on the right. Notice that the in-phase prompt correlator takes on the satellite signal's navigation data message, and the quadrature prompt appears to be zero-mean noise. The in-phase early and late correlators also take on the navigation data message with half the amplitude due to the half chip spacing. The quadrature early and late correlators appear as noise like the quadrature prompt correlator.

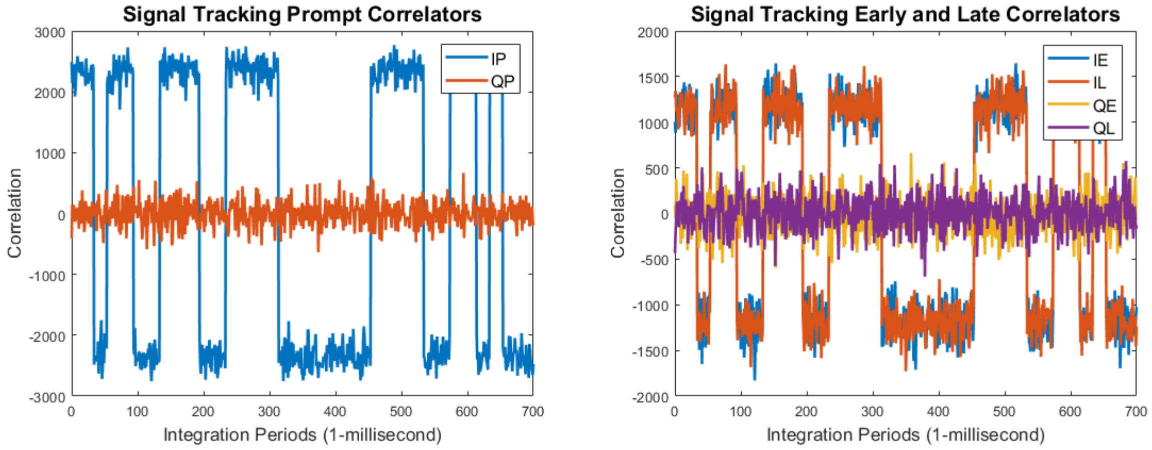

Figure 3.9: Correlators during Accurate Signal Tracking

3.3.4 Tracking Discriminators

The signal tracking correlators are used to calculate representations of the code and carrier errors known as discriminators. There are many types of discriminators that can be used to represent tracking errors, which are derived from the correlator model in Equation (3.3). This work presents two common discriminators used in the PLL and DLL. Equation (3.5) expresses the calculation of the Costas loop PLL discriminator ϕ_{PLL}.

$$\phi_{PLL} = \arctan\left(\frac{QP}{IP}\right)$$

$$
\begin{aligned}
&= \arctan\left(\frac{AR(\epsilon)D\sin(\pi f_{err}T+\theta_{err})+\eta_{QP}}{AR(\epsilon)D\cos(\pi f_{err}T+\theta_{err})+\eta_{IP}}\right) \qquad (3.5)\\
&= \arctan\left(\frac{\sin(\pi f_{err}T+\theta_{err})+\eta_{QP}}{\cos(\pi f_{err}T+\theta_{err})+\eta_{IP}}\right)\\
&\approx \theta_{err}+\eta_{PLL}
\end{aligned}
$$

The PLL discriminator represents the total carrier phase error $(\pi f_{err}T + \theta_{err})$ between the receiver's replica and the signal. However, when the carrier frequency error f_{err} is small and the integration period T is small, the PLL discriminator is a representation of the carrier phase error θ_{err} between the replica and signal. The Costas loop discriminator is a popular feedback error for PLL's because it is insensitive to the navigation data bit value D, as seen in the equation. Since signal data contains noise, the PLL discriminator also contains noise η_{PLL} that distorts the carrier error.

Figure 3.10 shows the Costas loop PLL discriminator output for a given carrier phase error. When the carrier phase error is within 90 degrees, the PLL discriminator output is linearly proportional to the true carrier phase error. When the true error is outside of 90 degrees, the PLL discriminator represents the error with 180 degrees or half a cycle of phase ambiguity. When the discriminator includes phase ambiguity, the PLL can still track the carrier, but inversion of the navigation data bit values occurs. In other words, the phase ambiguity causes the observed navigation data bits to flip sign, which must be accounted for when decoding the navigation data message. The carrier phase discriminator is divided by 2π to convert the error from radians to cycles.

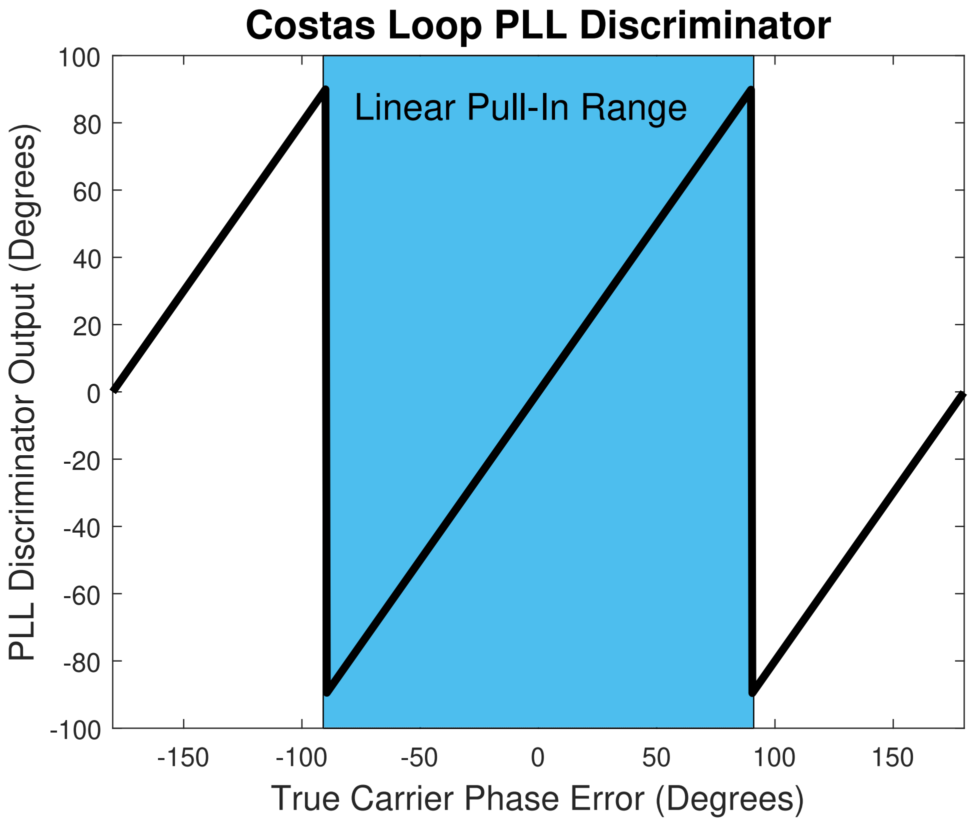

Figure 3.10: Costas Loop Discriminator Output for a Given Carrier Phase Error

The normalized early minus later power discriminator ϕ_{DLL} is used to represent code phase error ϵ and is shown as a function of the early and late correlators in Equation (3.6).

$$\phi_{DLL} = \frac{1}{2}\frac{\sqrt{IE^2 + QE^2} - \sqrt{IL^2 + QL^2}}{\sqrt{IE^2 + QE^2} + \sqrt{IL^2 + QL^2}} \approx \epsilon + \eta_{DLL} \tag{3.6}$$

This is a non-coherent DLL discriminator that can accurately represent code error when the receiver is not carrier phase-locked with the satellite signal. The in-phase and quadrature correlators in the equation ensure the code error calculation is insensitive to carrier phase error. The normalization is required to remove scaling induced by the signal amplitude A and navigation data bit D from the correlator model in Equation (3.3). Like the Costas loop carrier phase error, the DLL discriminator also contains noise η_{DLL}.

Figure 3.11 shows the normalized early minus late power discriminator output for a given code phase error. The figure shows the function for different chip spacings. The linear range of the function occurs for code phase errors within a single chip spacing. The linear range is where the DLL discriminator output is equivalent to the true code phase error. The pull-in range occurs for code phase errors between one and two chip spacings. When the true code error is between the linear and pull-in ranges, the receiver may or may not be able to lock onto the PRN code, depending on the receiver's filter design. The true code error must stay within 1 chip to guarantee tracking lock on the PRN code. The figure shows the linear and pull-in ranges when the chip spacing is $\tau = 0.5$ chip.

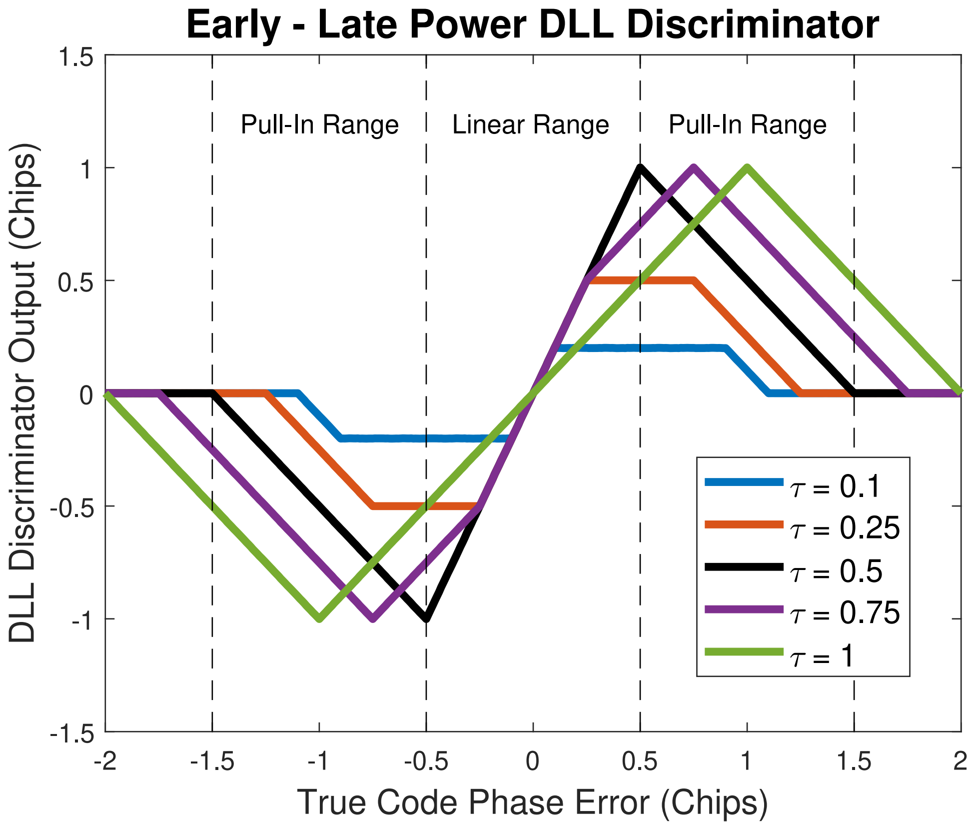

Figure 3.11: DLL Discriminator Output for a Given Code Phase Error

The tracking discriminators are designed to provide accurate representations of carrier and code errors when the errors are small. This allows the PLL and DLL to converge to accurate carrier and code replicas together in separate control loops. It is important to note that if the code phase error becomes too large, the PLL discriminator becomes an inaccurate representation of carrier error. The DLL discriminator also becomes inaccurate if the carrier frequency error becomes too large. GNSS receivers can only accurately track both the carrier and code.

With the tracking loops discussed here, the carrier and code must both be tracked together. One cannot achieve tracking lock without the other.

3.3.5 Loop Filters and NCO Update

Loop filters are compensators that update the receiver's replica carrier frequency using the PLL discriminator and update the receiver's replica code frequency using the DLL discriminator. Loop filters behave as low-pass filters that remove noise from the discriminators to provide better representations of the tracking errors. Even in the case of a strong clean signal, loop filters also compensate for frequency dynamics in the satellite signal. Recall that the satellites always move in orbit; hence there is always signal dynamics, even for a static receiver. The PLL loop filter is generally second or third order, and the DLL loop filter is typically first or second order. Since the PRN code is transmitted at a much lower frequency than the carrier, the DLL loop filter order is usually lower than the PLL loop filter order. Figure 3.12 shows a diagram of a second-order continuous-time loop filter used in PLL's and DLL's. This loop filter is modeled with a natural frequency ω and a damping ratio ζ. The implementation of this loop filter comes from [53].

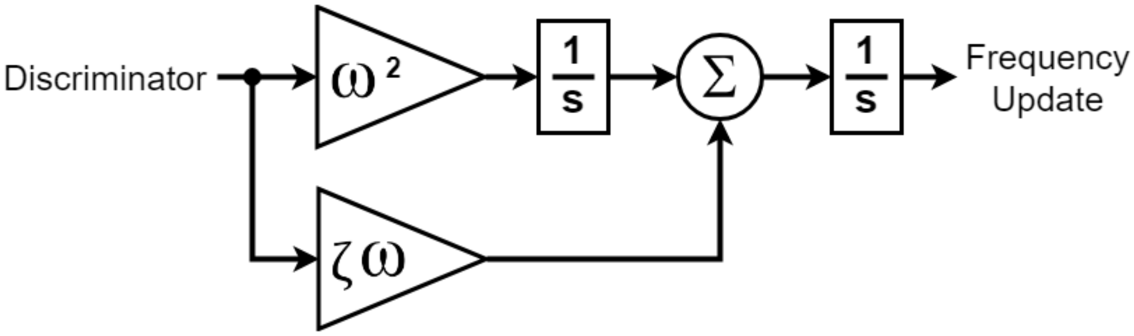

Figure 3.12: Diagram of a Second-Order Continuous-Time Loop Filter

The loop filter implementation from Figure 3.12 is shown as a Numerically Controlled Oscillator (NCO) update for the PLL in Equation (3.7).

$$f_{carr_{k+1}} = f_{carr_k} + K(\omega^2 T\phi_{PLL_k} + \omega\zeta(\phi_{PLL_k} - \phi_{PLL_{k-1}})) \tag{3.7}$$

In the equation, $f_{carr_{k+1}}$ is the replica carrier frequency for the next period, f_{carr_k} is the replica carrier frequency for the current period, ϕ_{PLL_k} is the PLL discriminator for the current period, $\phi_{PLL_{k-1}}$ is the PLL discriminator for the previous period, and T is the signal tracking integration time length. The loop filter parameters are the loop gain K, the natural frequency ω, and the damping ratio ζ. In this implementation, the damping ratio is generally $\zeta = 0.707$ to 1.414. The natural frequency is calculated from a selected noise bandwidth BW_n. The relationship between the natural frequency and the noise bandwidth is shown in Equation (3.8).

$$BW_n = \frac{\omega(1+\zeta^2)}{4\zeta} \tag{3.8}$$

The noise bandwidth determines how much noise the loop filter will allow into the frequency update. The noise bandwidth also allows signal dynamics into the frequency update, which causes a trade-off between noise and signal dynamics. A low noise bandwidth removes noise, but also attenuates signal dynamics, while a high noise bandwidth keeps signal dynamics as well as noise. When the noise bandwidth is too high, the replica frequency estimate becomes too noisy and the receiver fails to track. When the noise bandwidth is too low, the signal dynamics are strongly attenuated and the receiver fails to track. The selection of the noise bandwidth is generally between 5 to 18 Hz. The bandwidth should be determined by receiver dynamics and signal noise levels. The loop gain K is an overall gain for the frequency update. In PLL's, the loop gain is generally $K = 1$ to 4.

Figure 3.13 shows PLL frequency responses for the second-order loop filter from Equation (3.7). Two of the responses are able to converge to the correct frequency. As the noise bandwidth is increased, the system response is faster. The response that did not converge had an initial frequency error that was too large and a noise bandwidth that was too low. This loop filter does not provide enough NCO input for the system to converge to the correct frequency. Since PLL loop filters rely on phase errors instead of frequency errors, they cannot accurately respond if the frequency error is too high. In general, PLL's for GPS and GLONASS carriers can only handle up to 300 Hz of frequency error before failing.

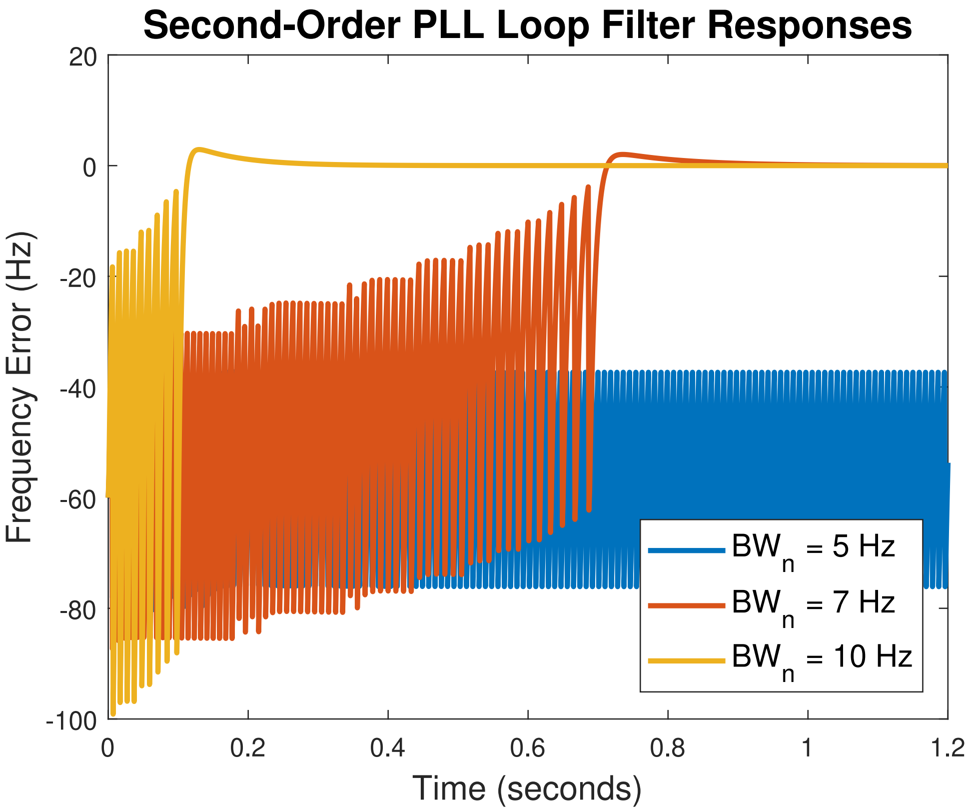

Figure 3.13: PLL Loop Filter Responses for Different Noise Bandwidths

Figure 3.14 shows continuous-time Bode Plots for the second-order loop filter with different noise bandwidths. Increasing the noise bandwidth causes the loop filter to allow (to amplify) additional frequencies into the update. The additional frequencies will include noise and possible signal dynamics. Increasing the noise bandwidth also adds additional phase lag (delay) into the frequency update. If the noise bandwidth is increased too much, the loop filter will become unstable.

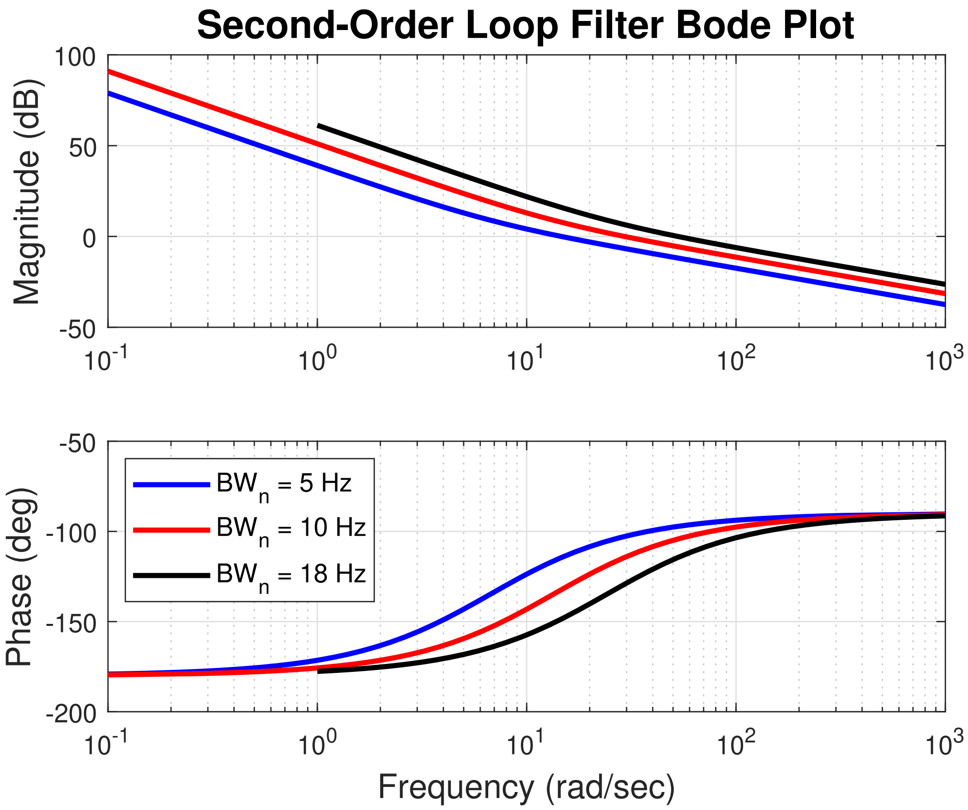

Figure 3.14: PLL Loop Filter Bode Plots for Different Noise Bandwidths

Equation (3.9) shows the second-order loop filter implementation as a DLL code NCO update.

$$f_{code_{k+1}} = f_{code_k} - K(\omega^2 T\phi_{DLL_k} + \omega\zeta(\phi_{DLL_k} - \phi_{DLL_{k-1}})) \tag{3.9}$$

The DLL implementation is very similar to the PLL version in Equation (3.7), however the frequency update from the loop filter is subtracted from the current code frequency instead of added as in the PLL. This is why the implementation is known as a “delay” lock loop, the loop filter update delays the code frequency. The PLL keeps carrier phase lock that may include a 180 degree phase ambiguity, because the sinusoidal wave function is ambiguous. The PRN code is a repeated patterned code for timekeeping, its phase cannot be ambiguous. From this, positive code phase errors require a decrease in code frequency for alignment, and negative code phase errors require an increase in code frequency for alignment. This results in the negative sign in the DLL loop filter implementation. In this work, the following loop filter parameters are used for the DLL; $K = 1$, $\zeta = 0.707$ to 1.414, and $BW_n = 1$ to 5 Hz. The loop gain K and noise bandwidth BW_n are lower for the DLL implementation because the PRN code is transmitted at a much lower frequency than the carrier in GNSS signals.

After the carrier and code replicas are locked by individual PLL and DLL filters, the Doppler frequency from the carrier replica can be used to aid code tracking. This is known as Doppler aided code tracking. The Doppler effect on the PRN code will be proportional to the Doppler effect on the carrier, because both signals take on the same line-of-sight dynamics between the satellite and receiver. Equation (3.10) shows a first-order DLL, Doppler-aided code NCO update where f_{code_t} is the transmitted code frequency at the satellite, f_{carr_t} is the transmitted carrier frequency at the satellite, and f_D is the estimated carrier Doppler frequency from the PLL.

$$f_{code_{k+1}} = f_{code_t} + \frac{f_{code_t}}{f_{carr_t}} f_D - \frac{BW_{n_{DLL}}}{4} \phi_{DLL_k} \tag{3.10}$$

Since GNSS satellites transmit their carriers at much higher rates than their codes, noise on the carrier Doppler frequency estimate will be attenuated in the code NCO update. The code NCO update also includes a first-order update from the DLL discriminator ϕ_{DLL_k} to ensure the code replica stays phase aligned for accurate timekeeping. Note that two codes can be frequency locked without phase lock, which is not acceptable for signal tracking.

3.3.6 The Frequency Lock Loop

The Frequency Lock Loop (FLL) is an alternative control loop to the PLL for carrier tracking. Instead of using a phase error as the PLL does, the FLL uses a frequency error to perform carrier tracking. This allows the FLL to perform tracking in high dynamic and signal degraded conditions the PLL could not. Most commercial receivers start signal tracking with an FLL to lower the frequency error in range for the PLL. If the PLL loses lock on the carrier, the FLL will reinstate until the tracking error is low enough for the PLL to operate again. Like the PLL, the FLL has a limit in frequency error it can withstand before failing. For GPS and GLONASS signals, FLL's can withstand up to 500 Hz of frequency error, which is why signal acquisition uses 500 Hz bins for the Doppler frequency search [49].

Although the FLL can perform in worse signal conditions than the PLL, it cannot phase lock the replica to the satellite's carrier. In other words, a FLL performs coarse carrier tracking, while a PLL performs fine carrier tracking. The satellite signal's navigation data message cannot be accurately decoded without carrier phase lock. Figure 3.15 shows the signal tracking correlators during accurate signal tracking when operating with a FLL. Notice that signal power leaks into the quadrature correlators, and the navigation data bits cannot be made out.

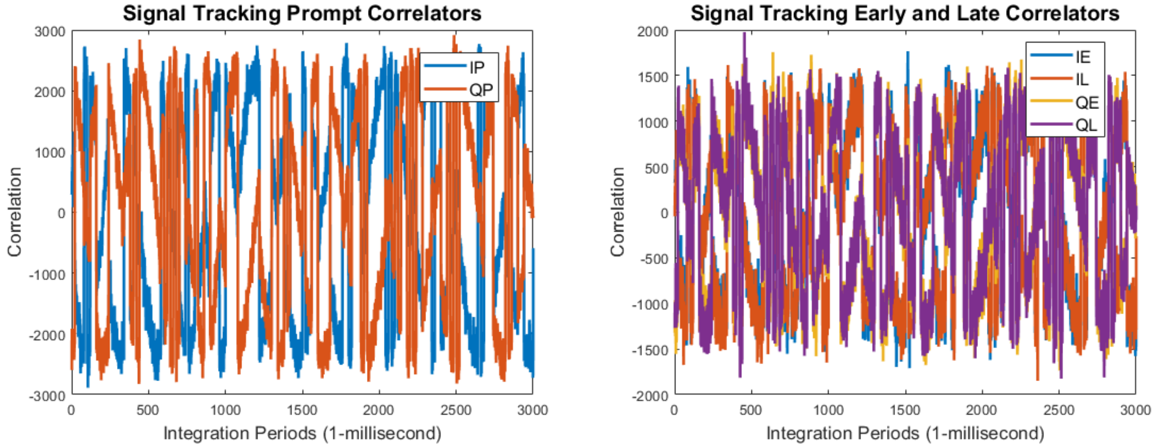

Figure 3.15: Correlators during Accurate Signal Tracking without Carrier Phase Lock

As with the PLL and DLL, there are different types of discriminators that can be used to represent tracking error for the FLL. In this work, the cross-dot arctangent discriminator ϕ_{FLL} shown in Equation (3.11) is used.

$$\phi_{FLL} = \frac{\arctan 2(IP_1QP_2 - IP_2QP_1, IP_1IP_2 + QP_1QP_2)}{\pi T} \approx f_{err} + \eta_{FLL} \tag{3.11}$$

In the equation, IP_1 and QP_1 are the in-phase and quadrature prompt correlators for the first half of the integration period. IP_2 and QP_2 are the in-phase and quadrature prompt correlators for the second half of the integration period. The FLL discriminator is calculated using the four-quadrant arctangent function along with the tracking integration period T. This discriminator represents the carrier frequency error f_{err}, which will contain noise η_{FLL} induced from the receiver's front-end and signal environment. Essentially, the FLL discriminator is calculated by finding a carrier phase error difference over a period of time. This explains the two sets of prompt correlators and the integration period in the calculation. The mathematical constant π is used to convert the error from radians per second to Hz.

Figure 3.16 shows the FLL discriminator output as a function of true carrier frequency error for different integration period lengths. As the integration period length becomes smaller, the frequency bandwidth of the FLL discriminator's linear pull-in range increases. However, this discriminator is sensitive at lower integration periods to noise due to the $(1/T)$ term in Equation (3.11). Figure 3.16 also shows that the frequency discriminator can be zero when the true frequency error is not. This is known as false frequency lock and occurs at different intervals for different integration periods. Depending on loop filter order, if the frequency error is less than 300 Hz, aiding from a PLL can be used to get the carrier tracking loop out of false frequency lock.

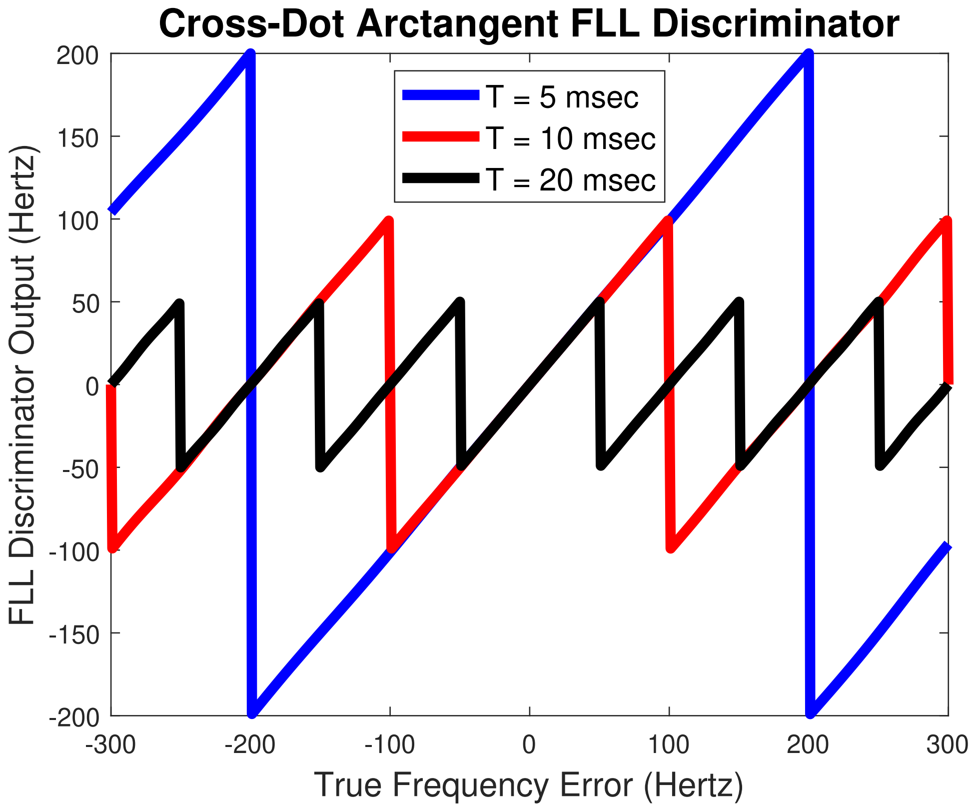

Figure 3.16: FLL Discriminator Output for a Given Carrier Frequency Error

A second-order FLL loop filter implementation is shown in Equation (3.12).

$$f_{carr_{k+1}} = f_{carr_k} + TK(\omega^2 T\phi_{FLL_k} + \omega\zeta(\phi_{FLL_k} - \phi_{FLL_{k-1}})) \tag{3.12}$$

This implementation is very similar to the PLL loop filter in Equation (3.7), the only difference between the filters is the inclusion of the integration period T in the FLL. The integration period is included to convert the frequency unit from the FLL discriminator to a phase unit, as the PLL uses. The typical loop filter parameters for the FLL are $K = 1$ to 4, $\zeta = 0.707$ to 1.414, and $BW_n = 5$ to 18 Hz.

3.3.7 Signal Tracking Integration Period

When tracking initially begins, the integration period is required to be 1 millisecond for the GPS and GLONASS L1 signals. This restriction comes from the signals' modulation designs. As with acquisition, the integration period of correlation can be increased in tracking to amplify the observed signal power. From the correlator model in Equation (3.3), increasing the integration period drowns out noise on the discriminators, which increases tracking performance. However, higher integration periods can only be performed when the beginning of a navigation data bit is known. If the integration period does not start at the beginning of a data bit, a bit transition may occur during the integration period and destroy correlation. For GPS, the start of a data bit can be found by locating the signal's preamble. For GLONASS, the start of a data bit can be found by locating the signal's time mark.

Once the starting location of a data bit is determined, the tracking integration period can be as high as 20 milliseconds for the GPS and GLONASS L1 signals. Both GPS and GLONASS have navigation messages with data bit lengths of 20 milliseconds. For GLONASS, when a time mark is located, the meander sequence becomes a known periodic code that can be wiped-off. In general, a satellite channel's data message is unknown to the receiver, hence the receiver cannot track more than a single data bit without risking loss of correlation.

In some cases, the receiver may fully know the navigation data message. In these situations, the tracking integration period can be longer. However, due to the Doppler effect, the carrier and code frequencies are always dynamic. During an integration period, the carrier and code frequencies are always changing. Signal tracking provides estimates or averages of the dynamic carrier and code frequencies in each period. As the tracking integration period increases, the resolution of the carrier and code frequencies decrease. If the tracking integration becomes too large, the dynamics of the carrier and code frequencies are lost, and the receiver will not be able to obtain accurate measurements for calculating a navigation solution. Signal tracking performance at higher integration periods is discussed in [64].

Longer tracking integration periods also increase the potential of the loop filters to become unstable. The loop filters are designed in continuous-time and discretized for application in the software receiver. In order for a loop filter to be stable, the product of the integration period and the noise bandwidth must be small ($BW_nT << 1$). Figure 3.17 shows Bode plots for discrete second-order loop filters, where each loop filter uses the same design parameters, but different integration periods. As the integration period becomes longer, the overall gain in the filter is enlarged, which increases the probability of unstable tracking. Also, discretizing the loop filter converts the compensator from a low-pass filter to a high-pass filter. However, the correlation performed in signal tracking is not shown in this Bode plot, a low-pass filtering process. In general, the noise bandwidth used for the loop filters should be lowered as the integration period increases to prevent instability.

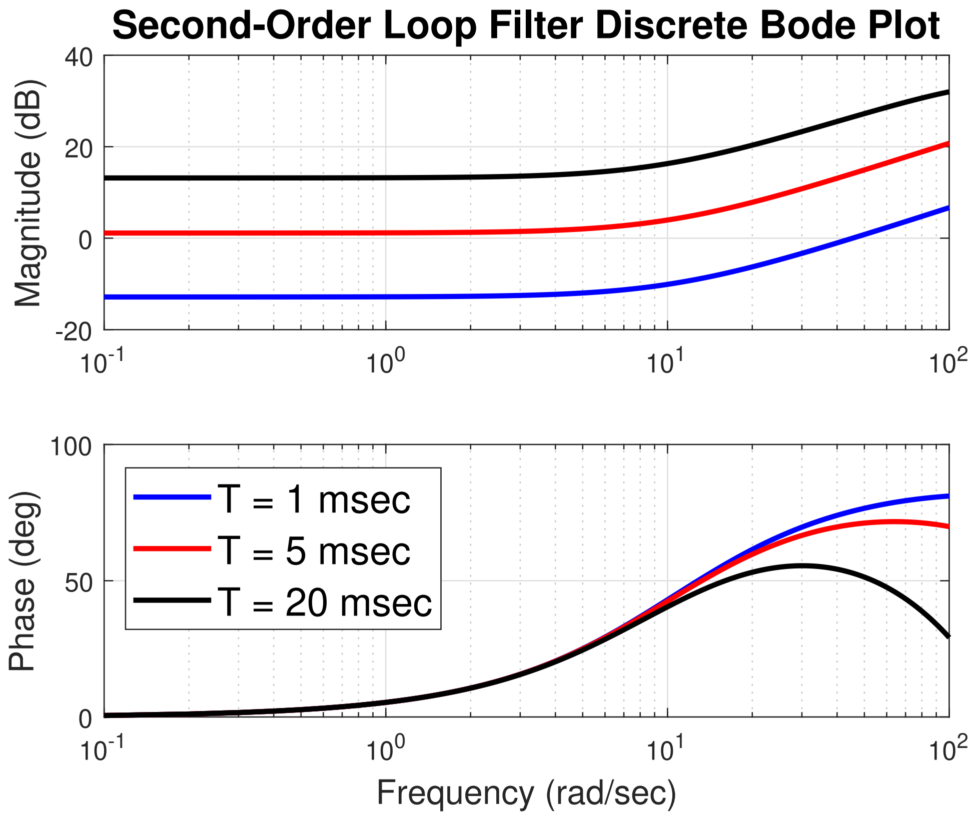

Figure 3.17: Discrete Loop Filter Bode Plots for Different Integration Periods at the Same Noise Bandwidth

3.4 Position, Velocity, and Time Calculation

The end goal of a GNSS receiver is to calculate its PVT solution. Specifically, the receiver estimates its PVT using measurements extracted from signal tracking. Code tracking produces pseudorange measurements that provide distances between the receiver and the satellites. Carrier tracking produces pseudorange rate measurements that provide range velocities between the receiver and satellites. A decoded ephemeris from the satellite signals' navigation data messages is used to calculate satellite positions, satellite velocities, and satellite clock corrections used in the PVT estimation. GPS and GLONASS both provide their satellite positions and velocities in Earth-Centered Earth-Fixed (ECEF) coordinate frames. Because of this, the receiver position and velocity solutions are typically estimated in an ECEF frame. Transitioning the receiver solution from the ECEF frame to another navigation frame (ENU, NED, lla, ECI, etc.) requires a coordinate transformation.

3.4.1 Receiver Position and Clock Bias Calculation

The pseudorange ρ is measured by finding the time difference between when a code chip on the GNSS signal was transmitted by the satellite and when that code chip entered the receiver's antenna. The PRN code sequence is used as the timekeeping mechanism to measure this time difference. The time difference is multipled by the speed of light constant c to convert the pseudorange measurement from a time unit to a distance unit. Electromagnetic (EM) waves travel at the speed of light. In free space, GNSS carriers are EM waves that operate in the radio frequency spectrum.

Equation (3.13) expresses the pseudorange ρ as a function of the true range between the satellite and receiver r along with biases and errors in the measurement.

$$\rho = r + I + T + ct_{sv} + cb_r + \eta_\rho \tag{3.13}$$

The speed of light constant assumes the satellite signal travels through a vacuum, but the signal must travel through the atmosphere to reach Earth's surface. The speed of light is slightly lower in Earth's atmospheric medium compared to empty space, which slows the signal down. Atmospheric errors from the ionosphere and troposphere incur delays on the pseudorange measurement. In the equation, I is the atmospheric delay induced by the ionosphere, and T is the atmospheric delay induced by the troposphere.

Since timekeeping is required to produce pseudorange measurements, clock errors are also observed. Both the transmitting satellite and receiver are required to have clocks to measure the time difference for the pseudorange. All clocks drift, which induce time errors. The pseudorange measurement incurs a satellite clock error t_{sv} and a receiver clock error b_r. Satellite clocks use highly stable atomic oscillators that keep time very well, but still have error. A satellite clock correction term from the signal's navigation message is used to correct the majority of this time error. GNSS receivers typically use inexpensive oscillators that incur a growing time error on the pseudorange measurements, known as the receiver clock bias. This time bias is the same for all pseudorange measurements, which allows the clock term to be solved for. This is why position and *time* are both solved for in a GNSS receiver.

There is also zero-mean Gaussian distributed random noise η_ρ on the pseudorange measurement. The majority of this noise comes from the receiver's front-end through thermal noise. Some noise also comes from the receiver's environment. The amount of noise on the measurement is dependent on the channel's noise power and the effectiveness of the DLL in removing noise from the code replica.

The pseudorange is modeled as a function of the receiver position and clock bias in Equation (3.14).

$$\rho = \sqrt{(x_{sv} - \hat{x}_r)^2 + (y_{sv} - \hat{y}_r)^2 + (z_{sv} - \hat{z}_r)^2} + \hat{cb_r} \tag{3.14}$$

In the equation, x_{sv}, y_{sv}, and z_{sv} are the satellite ECEF position coordinates in the x, y, and z-directions; $\hat{x}_r$, $\hat{y}_r$, and $\hat{z}_r$ are the estimated receiver ECEF position coordinates in the x, y, and z-directions; and $\hat{cb_r}$ is the estimated receiver clock bias scaled by the speed of light. The receiver's position coordinates and clock bias are unknown, thus at least four pseudorange measurements are required to determine them.

3.4.2 Receiver Velocity and Clock Drift Calculation

The pseudorange rate $\dot{\rho}$ is a measurement of the change in the pseudorange with respect to time. The measurement is generated with the Doppler frequency from carrier tracking. Equation (3.15) expresses the pseudorange rate as a function of the line-of-sight velocity between the satellite and receiver $\dot{r}$, along with the receiver clock drift error $\dot{b}_r$ and a noise term $\eta_{\dot{\rho}}$.

$$\dot{\rho} = \dot{r} + \dot{I} + \dot{T} + c\dot{t}_{sv} + c\dot{b}_r + \eta_{\dot{\rho}} \approx \; \dot{r} + c\dot{b}_r + \eta_{\dot{\rho}} \tag{3.15}$$

The atmosphere changes slowly, which results in the ionospheric delay rate $\dot{I}$ and tropospheric delay rate $\dot{T}$ being approximately zero. .

Like the pseudorange, the pseudorange rate measurement also incurs clock errors from the satellite and receiver. Since satellite clocks are highly stable, the satellite clock drift error $\dot{t}_{sv}$ is very small and can be ignored. The receiver's clock has a time bias but also a time drift error that is observed in the pseudorange rate, known as the receiver clock drift $c\dot{b}_r$. The receiver clock drift is modeled as the time derivative of the receiver clock bias, which indicates the same clock drift is observed on all channels' pseudorange rates. The receiver clock drift is solved for in the velocity calculation.

The pseudorange rate also has noise on its measurement $\eta_{\dot{\rho}}$, induced from the receiver's front-end and signal enviornment. The amount of noise on the pseudorange rate measurement depends on the signal's noise levels and the effectiveness of the PLL/FLL in filtering noise from the carrier replica.

The pseudorange rate is expressed as a function of the receiver velocity and clock drift in Equation (3.16).

$$\dot{\rho} = (\dot{x}_{sv} - \hat{\dot{x}}_r)a_x + (\dot{y}_{sv} - \hat{\dot{y}}_r)a_y + (\dot{z}_{sv} - \hat{\dot{z}}_r)a_z + c\hat{\dot{b}}_r \tag{3.16}$$

In the equation, $\dot{x}_{sv}$, $\dot{y}_{sv}$, and $\dot{z}_{sv}$ are the satellite ECEF velocities in the x, y, and z-directions; $\hat{\dot{x}}_r$, $\hat{\dot{y}}_r$, and $\hat{\dot{z}}_r$ are the estimated receiver ECEF velocities in the x, y, and z-directions; a_x, a_y, and a_z are the line-of-sight unit vectors from the satellite to the receiver in the x, y, and z-directions; and $c\hat{\dot{t}}_r$ is the estimated receiver clock drift scaled by the speed of light. The calculation of the line-of-sight unit vectors are shown in Equation (3.17), which are a function of the estimated receiver position and the estimated range $\hat{r}$ between the satellite and receiver, shown in Equation (3.18).

$$a_x = \frac{x_{sv} - \hat{x}_r}{\hat{r}} \qquad a_y = \frac{y_{sv} - \hat{y}_r}{\hat{r}} \qquad a_z = \frac{z_{sv} - \hat{z}_r}{\hat{r}} \tag{3.17}$$

$$\hat{r} = \sqrt{(x_{sv} - \hat{x}_r)^2 + (y_{sv} - \hat{y}_r)^2 + (z_{sv} - \hat{z}_r)^2} \tag{3.18}$$

From this, it is shown that the receiver position must be known to accurately estimate receiver velocity. At least four pseudorange rate measurements are required to calculate the three receiver velocity coordinates and the clock drift.

3.4.3 Newton-Raphson Least Squares Calculation

Each satellite channel provides a pseudorange and pseudorange rate measurement. At least four satellite channels must be accurately tracked to provide enough measurements to estimate receiver PVT. In the case when more than four satellite channels are tracked, a least squares solution can be performed to include all the channels' measurements in the estimation. As shown from Equations (3.13) and (3.15), these measurements have errors that degrade the PVT solution. Including more measurements into the PVT calculation provides more noise filtering and potentially a more accurate solution.

A least squares solution provides the "best fit" for an overdetermined system (more measurements than variables). Least squares uses linear algebra or matrix math to perform the calculation. Equation (3.19) shows a standard least squares solution where $\hat{a}$ is the estimated state vector, b is the measurement vector, and C is the measurement observation matrix that relates the state vector to the measurement vector.

$$\hat{a} = (C^T C)^{-1} C^T b \tag{3.19}$$

The T operation refers to the transpose of a matrix. In the case where there are n state variables to be estimated and m measurements; then $\hat{a}$ is a n x 1 vector, b is a m x 1 vector, and C is a m x n matrix. Note that least squares will correctly estimate only when $m \geq n$, the estimator must be observable to work properly.

The least squares calculation can only be performed for linear systems. Equations (3.14) and (3.16) show that GNSS estimation is nonlinear. To use least squares, the GNSS equations can be linearized at a current PVT prediction using the Newton-Raphson method. Newton-Raphson linearization involves taking partial derivatives of the nonlinear pseudorange and pseudorange rate functions with respect to the state variables being estimated. Equation (3.20) shows the linearized pseudorange-to-state equation where $\tilde{\rho}$ is the measured satellite clock corrected pseudorange, $\hat{\rho}$ is the estimated pseudorange, $\Delta\hat{x}_r$ is the estimated error in the receiver x-position, $\Delta\hat{y}_r$ is the estimated error in the receiver y-position, $\Delta\hat{z}_r$ is the estimated error in the receiver z-position, and $c\Delta\hat{b}_r$ is the estimated error in the receiver clock bias.

$$\tilde{\rho} - \hat{\rho} = -a_x \Delta \hat{x}_r - a_y \Delta \hat{y}_r - a_z \Delta \hat{z}_r + c\Delta \hat{b}_r \tag{3.20}$$

The unit vectors from the satellite to the receiver in Equation (3.17) are included in the linearized equation. The estimated pseudorange $\hat{\rho}$ can be calculated from the current predicted position and clock bias using the nonlinear function in Equation (3.14).

Equation (3.21) shows the linearized pseudorange rate-to-state equation where $\dot{\tilde{\rho}}$ is the measured pseudorange rate, $\dot{\hat{\rho}}$ is the estimated pseudorange rate, $\Delta \dot{\hat{x}}_r$ is the estimated error in the receiver x-velocity, $\Delta \dot{\hat{y}}_r$ is the estimated error in the receiver y-velocity, $\Delta \dot{\hat{z}}_r$ is the estimated error in the receiver z-velocity, and $c\Delta \dot{\hat{b}}_r$ is the estimated error in the receiver clock drift.

$$\dot{\tilde{\rho}} - \dot{\hat{\rho}} = -a_x \Delta \dot{\hat{x}}_r - a_y \Delta \dot{\hat{y}}_r - a_z \Delta \dot{\hat{z}}_r + c\Delta \dot{\hat{b}}_r \tag{3.21}$$

The unit vectors from the satellite to the receiver in Equation (3.17) are included in the linearized equation. The estimated pseudorange rate $\dot{\hat{\rho}}$ can be calculated from the current predicted velocity and clock drift using the nonlinear function in Equation (3.16).

Equations (3.22) through (3.27) describe the Newton-Raphson iterative least squares calculation to estimate the receiver PVT.

$$\underline{\hat{x}} = \begin{bmatrix} \hat{x}_r & \dot{\hat{x}}_r & \hat{y}_r & \dot{\hat{y}}_r & \hat{z}_r & \dot{\hat{z}}_r & c\hat{b}_r & c\dot{\hat{b}}_r \end{bmatrix}^T \tag{3.22}$$

$$\Delta\underline{\hat{x}} = \begin{bmatrix} \Delta\hat{x}_r & \Delta\dot{\hat{x}}_r & \Delta\hat{y}_r & \Delta\dot{\hat{y}}_r & \Delta\hat{z}_r & \Delta\dot{\hat{z}}_r & c\Delta\hat{b}_r & c\Delta\dot{\hat{b}}_r \end{bmatrix}^T \tag{3.23}$$

$$H = \begin{bmatrix} -a_{x_1} & 0 & -a_{y_1} & 0 & -a_{z_1} & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -a_{x_m} & 0 & -a_{y_m} & 0 & -a_{z_m} & 0 & 1 & 0 \\ 0 & -a_{x_1} & 0 & -a_{y_1} & 0 & -a_{z_1} & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -a_{x_m} & 0 & -a_{y_m} & 0 & -a_{z_m} & 0 & 1 \end{bmatrix} \tag{3.24}$$

$$Y = \begin{bmatrix} \tilde{\rho}_1 - \hat{\rho}_1 & \dots & \tilde{\rho}_m - \hat{\rho}_m & \dot{\tilde{\rho}}_1 - \dot{\hat{\rho}}_1 & \dots & \dot{\tilde{\rho}}_m - \dot{\hat{\rho}}_m \end{bmatrix}^T \tag{3.25}$$

$$\Delta\underline{\hat{x}} = (H^T H)^{-1} H^T Y \tag{3.26}$$

$$\underline{\hat{x}}_{k+1} = \underline{\hat{x}}_k + \Delta\underline{\hat{x}} \tag{3.27}$$

Equation (3.22) shows the state vector $\underline{\hat{x}}$ that represents the estimated PVT variables. An initial arbitrary guess is provided (generally, a guess of all zeroes) to the state vector, which is then converged to an accurate estimation over iterations of least squares calculations. Equation (3.23) provides the estimated error-in-state variables $\Delta\underline{\hat{x}}$ that are used to update $\underline{\hat{x}}$. Note that the clock bias and clock drift terms in $\underline{\hat{x}}$ and $\Delta\underline{\hat{x}}$ are scaled by the speed of light c. Equation (3.24) shows the linearized measurement observation H that displays the relationship between the measurement errors and the error-in-state variables. This matrix is shown for when m satellite channels are being accurately tracked in the receiver. The H matrix is equivalent to the C matrix in Equation (3.19) that displayed the general least squares solution. Equation (3.25) shows the linearized measurement vector Y that contains m pseudorange and m pseudorange rate measurement errors. The Y vector is equivalent to the b vector in the general least squares solution. Equation (3.26) shows the calculation of $\Delta\underline{\hat{x}}$ using least squares, and Equation (3.27) shows the update to the state vector $\underline{\hat{x}}$. These set of equations are iterated until $\Delta\underline{\hat{x}}$ becomes small, which indicates the convergence to a PVT solution.

3.4.4 Dilution of Precision

The least squares navigation solution provides an estimate of the receiver position, but the estimation accuracy is unknown unless the user knows its true position from some other source or reference. A receiver can calculate its Dilution of Precision (DOP) to determine its *estimated* position error. The DOP provides an idea of how accurate the position estimation is based on satellite-receiver geometry. In general, the more diverse the satellite geometry in-view of the receiver is, the more accurate the position estimate will be. If satellite channel measurements all come from the same area, the position estimation is less accurate. This concept is also true in other navigation systems, like radar. Figure 3.18 compares satellite geometries that have a high DOP and a low DOP, relative to each other. A high DOP indicates more inaccuacry or more estimation error in the position.

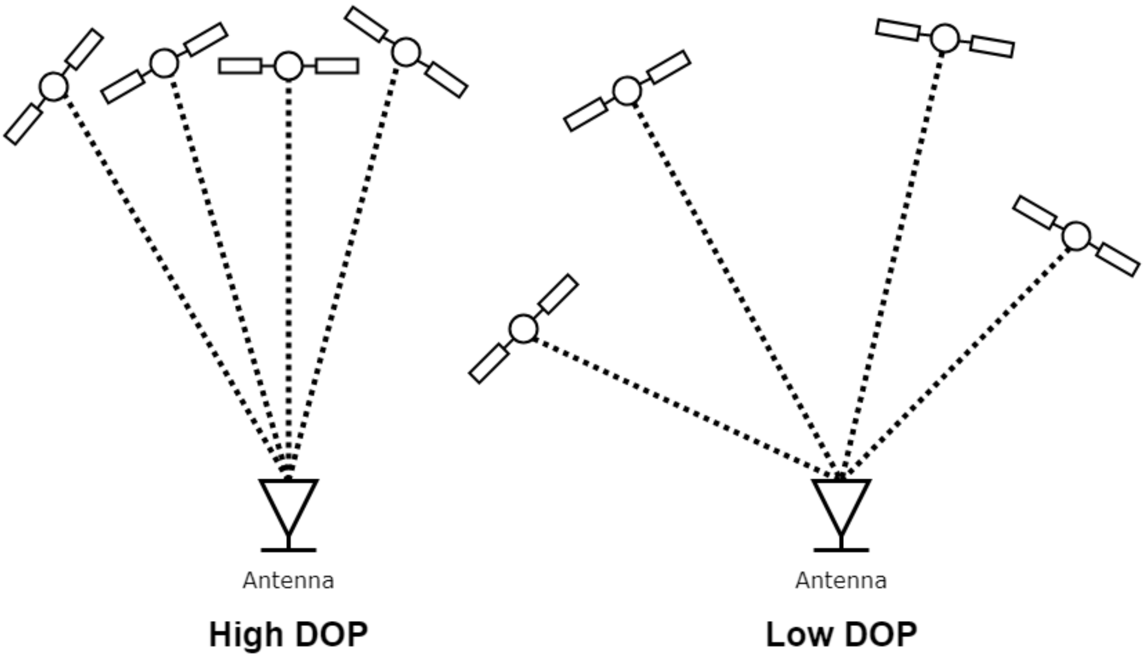

Figure 3.18: Comparison of High DOP and Low DOP Satellite Geometries

Equation (3.28) shows the calculation of the DOP where H is the measurement observation matrix for the receiver position and clock bias calculation.

$$H = \begin{bmatrix} -a_{x_1} & -a_{y_1} & -a_{z_1} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ -a_{x_m} & -a_{y_m} & -a_{z_m} & 1 \end{bmatrix}, \quad DOP = (H^T H)^{-1} \tag{3.28}$$

The DOP matrix is also referred to as the Hessian. For the DOP calculation, the unit vectors are typically converted from the ECEF coordinate frame to a navigation frame, such as East-North-Up (ENU) or North-East-Down (NED). Analyzing the DOP in a navigation frame gives more clarity about uncertainty than in an Earth frame.

Different types of DOP can be analyzed using the diagonal elements of the DOP matrix (DOP_{11}, DOP_{22}, DOP_{33}, and DOP_{44}). Equation (3.29) shows the calculations for Position DOP $PDOP$, Horizontal DOP $HDOP$, Vertical DOP $VDOP$, Time DOP $TDOP$, and Geometrical DOP $GDOP$.

$$
\begin{aligned}
PDOP &= \sqrt{DOP_{11} + DOP_{22} + DOP_{33}} \\
HDOP &= \sqrt{DOP_{11} + DOP_{22}} \\
VDOP &= \sqrt{DOP_{33}} \\
TDOP &= \sqrt{DOP_{44}} \\
GDOP &= \sqrt{DOP_{11} + DOP_{22} + DOP_{33} + DOP_{44}}
\end{aligned}
\tag{3.29}
$$

Since GNSS satellites are overhead of receivers, VDOP typically has the highest value.

3.5 Conclusion

This chapter discussed the basic operations required by any GNSS software receiver. All GNSS receivers require a front-end, signal acquisition block, signal tracking block, and a navigation estimator. The chapter focused on the signal tracking and navigation estimator processes. The effectiveness of these algorithms are directly responsible for providing position and timing services to users. Enhancements to the signal tracking and navigation estimators are discussed later in the thesis. This chapter touched on the pseudorange and pseudorange rate measurements used to calculate a receiver's PVT. The next chapter describes these measurements further in detail, as well as how they are extracted from signal tracking for use in the PVT estimator.

Chapter 4

GNSS Measurements

Once a GNSS receiver has performed accurate signal tracking, navigational measurements can be extracted to calculate a receiver's PVT. Code frequencies from signal tracking are used to generate pseudorange measurements for receiver position estimation. Carrier Doppler frequencies from signal tracking are used to generate pseudorange rate measurements for receiver velocity estimation. The pseudorange and pseudorange rate measurements contain time or clock errors, which must also be estimated. This chapter discusses the pseudorange and pseudorange rate measurements in detail along with other necessities the receiver must determine to calculate a navigation solution.

4.1 The Pseudorange Measurement

Pseudorange measurements are used to calculate a receiver's position. Each satellite in-view of the GNSS receiver provides a pseudorange measurement that quantifies the distance between the satellite and the receiver. The pseudorange is generated by measuring the time the signal travels from the satellite to the receiver. Equation (4.1) describes how the pseudorange $\tilde{\rho}$ is generated using the receive time t_r, the transmit time t_t, and the speed of light in a vacuum c.

$$\tilde{\rho} = c(t_r - t_t) \tag{4.1}$$

The transmit time t_t is associated to the time where a PRN code chip in the signal is transmitted from the satellite. GNSS signals add timestamps in their navigation data messages so the receiver knows the transmit time. For GPS, the transmit time is found from the time of week

(TOW). For GLONASS, the transmit time is found from the time of day (TOD). The receive time t_r is associated to the time where that code chip in the signal is observed by the receiver. Recall that GNSS signals are radio signals that travel at a constant speed of light.

Usually, a code chip where the pseudorange is generated on the satellite signal is associated to the start/end of a PRN code period. To accurately generate the measurement, t_t and t_r must be on the same timescale. Each GNSS constellation uses its own time system in which t_t and t_r are measured. A satellite signal's data message only provides transmit timestamps in intervals; the bookkeeping of the transmit time is shown by Equation (4.2), where T is the integration period used in signal tracking.

$$t_{t_{k+1}} = t_{t_k} + T \tag{4.2}$$

The bookkeeping of a satellite channel's receive time is shown by Equation (4.3).

$$t_{r_{k+1}} = t_{r_k} + \frac{f_{chip}T}{f_{code_k}} \tag{4.3}$$

In the equation, f_{chip} is the transmitted PRN code frequency or chipping rate at the satellite, and f_{code_k} is the code frequency estimate of the channel for the current integration period.

Generally, software receivers keep track of the receive time by using differences in signal data samples as shown by Equation (4.4).

$$t_{r_{k+1}} = t_{r_k} + \frac{s_{end} - s_{start}}{f_s} \tag{4.4}$$

In the equation, s_{start} is the first signal data sample read-in for the integration period, s_{end} is the last signal data sample read-in for the integration period, and f_s is the sampling frequency. The sampling frequency converts the difference from samples to seconds. The satellite channel receive time's resolution becomes better as the front-end's sampling frequency becomes larger. Note that the code frequency estimation from tracking directly determines how many samples of signal data will be contained within an integration period.

The pseudorange is a measurement of the distance between the satellite and the receiver; it is not the exact distance between the satellite and the receiver. The pseudorange measurement contains many errors which were defined by Equation (3.13). The largest errors in the pseudorange are atmospheric errors that cause delays in the measurement. In Equation (4.1), the GNSS signal is assumed to travel at the speed of light in a vacuum, which is an accurate assumption when the signal is traveling in outerspace. However, when the GNSS signal enters the Earth's atmosphere, the signal is traveling through an air medium that alters the speed of light. It is challenging to determine the speed of light constant within Earth's atmosphere because the atmospheric medium is different around the globe and changes with time. Dual frequency is a common method to remove ionospheric delay in the pseudorange measurement. Dual frequency uses pseudorange measurements from two signals at different carrier frequencies (L1/L2 or L1/L5 for GPS) from the same satellite to remove the ionospheric bias. The pseudorange measurement also contains the receiver clock bias, which is discussed in the next section.

4.1.1 Receiver Clock Bias

As stated previously, the pseudorange measurement is generated from the time required for the signal to travel from the satellite to the receiver. This time difference is only as accurate as the clocks used to keep the time at the satellite and the receiver. The satellite contains an atomic clock and time corrections, which make the satellite clock nearly perfect. The issue comes from the receiver clock, which must be small and inexpensive to be suitable. These qualities in the receiver clock also come with time error.

Note that the receiver clock is actually an oscillator that drives the front-end sampling device that discretizes the satellite signals entering the antenna. The same is true for the satellite clock, which is an oscillator that drives the transmission of the signal. The sampling frequency in Equation (4.4) depends on the oscillator or clock used in the receiver's front-end to sample the signal data. When a receiver's front-end states its sampling frequency is $f_s = 20$ MHz, but the actual sampling frequency is $f_s = 20.1$ MHz or $f_s = 19.9$ MHz due to error in the electronic device, this causes error in the receive time bookkeeping and the pseudorange measurement.

Specifically, the sampling frequency error causes a bias in the time difference, which causes a distance bias in the pseudorange measurement; this is known as the receiver clock bias. Since the same clock/oscillator is used to sample all the satellite signals, the pseudorange measurements from all satellite channels contain the same clock bias, which can be solved for along with the receiver's position.

There are different qualities of clocks used in GNSS receivers. The majority of GNSS receivers use a Temperature Controlled Crystal Oscillator (TCXO). The TCXO is inexpensive, but its clock bias grows over time due to clock drift or frequency bias in the oscillator. Lower quality TCXO's may also have acceleration in the clock bias, which the receiver must take into account. Higher-end receivers use an Oven Controlled Crystal Oscillator (OCXO), which has less growth in the clock bias and very little clock bias acceleration. Some receivers may also use Rubidium and Cesium atomic clocks, similar to the ones aboard satellites. These atomic clocks have small clock error with low growth/drift. Atomic clocks may be used with military receivers for important operations or expensive commerical receivers solely used for timekeeping [18]. Figure 4.1 shows estimated receiver clock biases from a static antenna in distance units for the oscillators described.

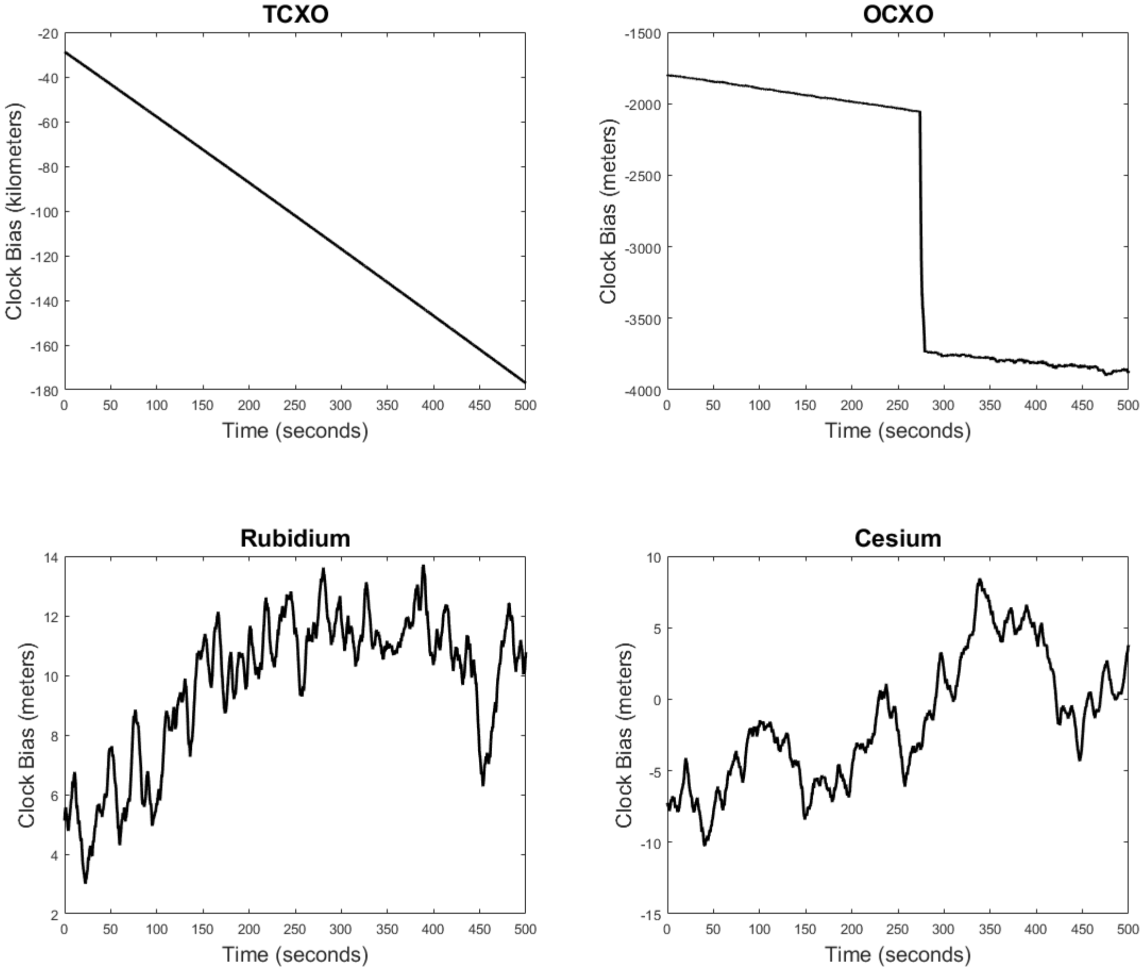

Figure 4.1: Receiver Clock Biases for Different Types of Oscillators

Depending on the receiver, GNSS can be used to obtain nanosecond time accuracy. Since GNSS receivers can estimate an oscillator's clock bias error accurately, many receivers are used solely for time estimation relative to a GNSS constellation's atomic clocks. Timing applications using GNSS have become important in banking, finance, commerce, the stock market, power utilities, and many communication systems including those in hospitals and police stations.

4.1.2 Pseudorange Multipath Bias

GNSS multipath occurs when a receiver's antenna collects delayed versions of a satellite signal. These delayed versions of a given satellite signal occur due to signal reflections off of buildings, tree foliage, and other structures. Figure 4.2 illustrates multipath occurring in a receiver. Multipath signals are always delayed because the line-of-sight signal travels the shortest distance between the satellite and receiver, while reflected signals travel along alternate paths to the receiver. Multipath signals also have smaller signal powers compared to the line-of-sight signal due to propagating through the environment longer. The increased use of GNSS receivers in urban canyons (suburban and downtown city areas), as well as forested areas, has made multipath is a common error in GNSS positioning.

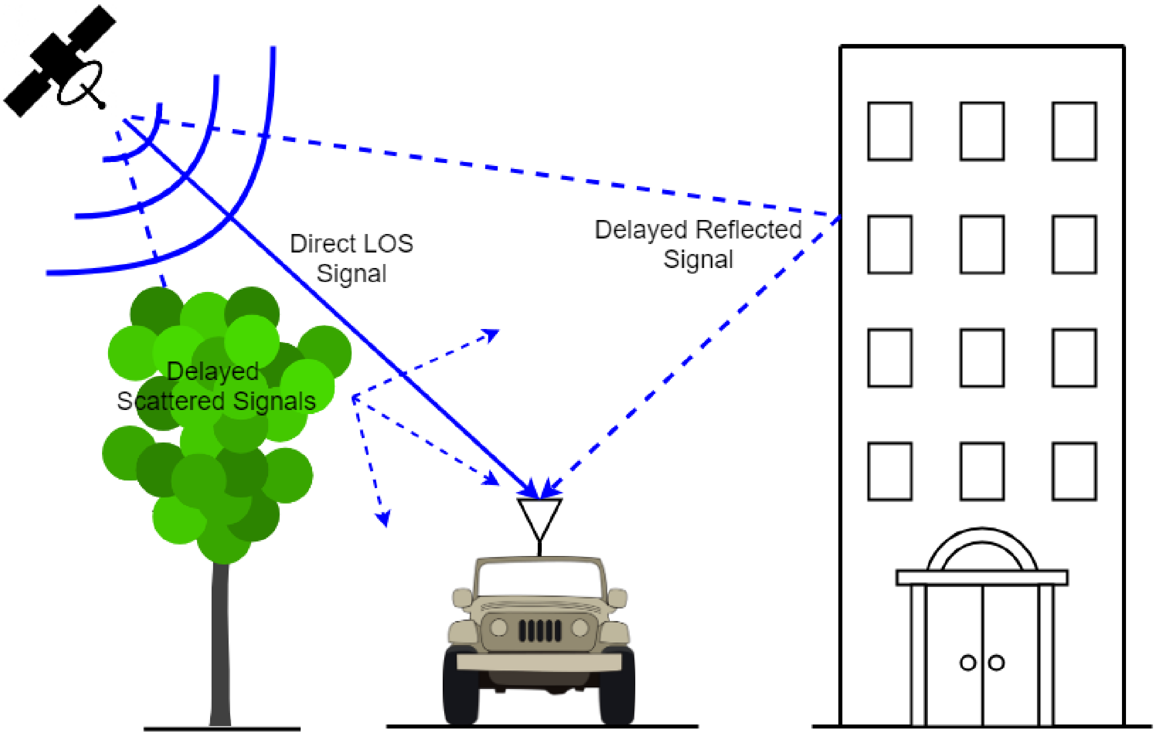

Figure 4.2: Illustration of GNSS Multipath

There are many factors to consider as to how multipath will effect the receiver PVT estimation. A few of these factors include the number of delayed multipath reflections, the signal power of these reflections, and the amount of delay that occurs with respect to the line-of-sight signal. In general, the receiver's measured pseudoranges can be biased due to multipath that can cause a biased or very noisy position estimation. The multipath bias in a channel's pseudorange relates back to PRN code tracking where the pseudorange measurement is generated. Consider a receiver replica tracking a line-of-sight satellite signal with a single delayed multipath signal introduced into the receiver. If the multipath signal is delayed less than one PRN code chip with respect to the line-of-sight signal, there will be correlation between the receiver replica code and the multipath code as described by the PRN autocorrelation function $R(\epsilon)$ in

Equation (2.1). This concept is illustrated in Figure 4.3 where the receiver code replica fights between lock on the GPS signal and the multipath reflection.

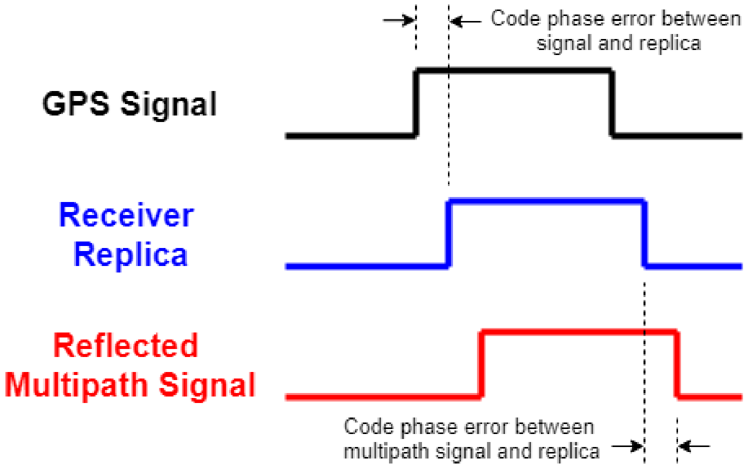

Figure 4.3: Code Correlation with the LOS Satellite PRN and a Delayed Multipath PRN

The PRN code correlation of the receiver replica with the satellite signal and the multipath signal creates a biased DLL code phase discriminator that ultimately biases the receiver channel's pseudorange measurement. The pseudorange bias from the presence of a multipath signal can be modeled using the signal tracking correlator model from Equation (3.3), where correlators are generated between the receiver replica and the line-of-sight signal, *and* correlators are generated between the receiver replica and the delayed multipath signal. Detailed information about this multipath model can be found in [5].

Figure 4.4 shows the pseudorange measurement bias as a function of the PRN code chip delay between the line-of-sight satellite signal and the multipath reflection. The pseudorange bias is shown when using the normalized early minus late power code discriminator (from Equation (3.6)) and a chip spacing of $\tau = 0.5$ chip. Altering the type of DLL discriminator and chip spacing used in PRN code tracking alters the shape of the pseudorange bias [48]. In the figure, different colors are associated to different multipath signal amplitudes. The figure shows biases for multipath signal amplitudes that are 25%, 35%, and 50% of the line-of-sight signal amplitude. The bias on the pseudorange measurement increases as the strength of the multipath reflection increases, relative to the line-of-sight signal. The plots also contain upper and lower limits on the pseudorange bias that come from the carrier phase error between the line-of-sight and multipath signals. The lower limits occur when the carrier phases between the satellite signal and the mulitpath reflection are in phase (no phase error). The upper limits occur when the carrier phases are completely out of phase (180° phase error). The pseudorange bias is most significant at a mulitpath chip delay of 0.5 chip, which comes from the 0.5 chip spacing used for the early and late correlators. When the mulitpath chip delay is greater than 1.5 chips, the pseudorange bias is zero because there is no longer any code correlation between the receiver replica and the mulitpath reflection.

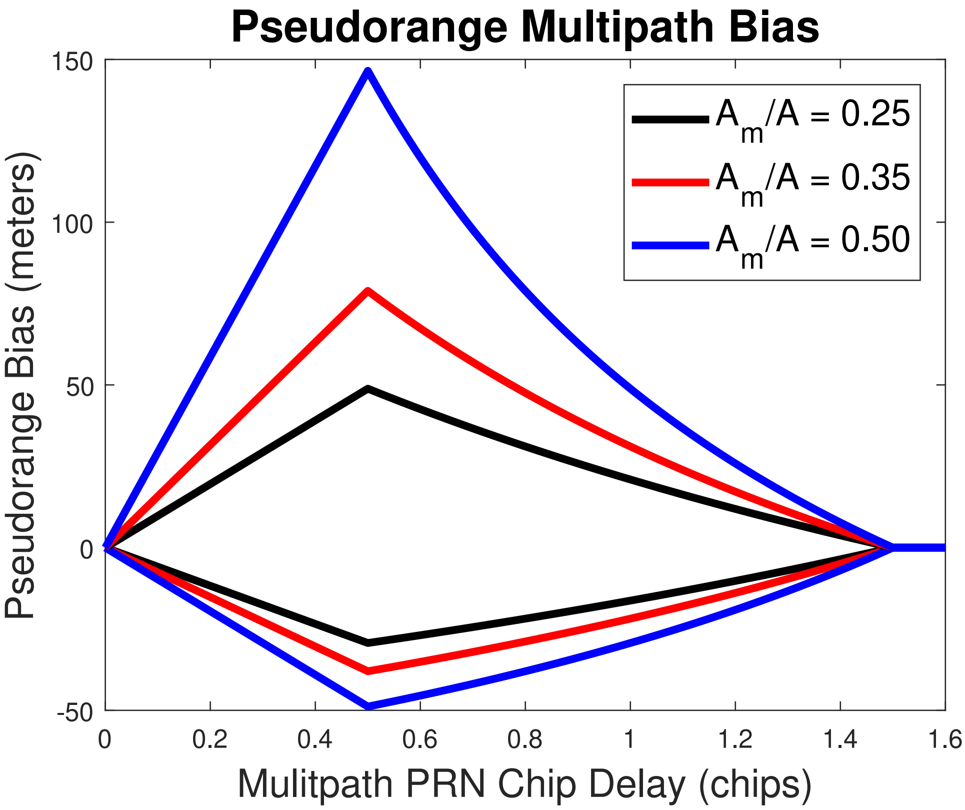

Figure 4.4: Multipath Pseudorange Bias as a Function of Multipath Delay

4.1.3 GNSS Time Synchronization and Relative Positioning

Although a satellite channel's code frequency estimate from signal tracking is used to propagate the receive time, the code frequency cannot determine an initial receive time. An initial receive time synchronized to the constellation timescale must be calculated before pseudorange measurements can be generated. The receiver performs time synchronization by taking advantage of the receiver clock bias through a process called relative positioning [42]. Relative positioning uses relative pseudoranges between the satellite channels within the same constellation to solve for receiver position instead of using actual pseudorange measurements. The error from using relative pseudoranges instead of true pseudorange measurements will only be seen as a constant error value in the receiver clock bias. In other words, relative positioning guesses an initial receive time for all the satellite channels that can then be corrected by the estimated receiver clock bias.

Figure 4.5 illustrates the process of relative positioning. On the left of the figure, four satellites are shown transmitting the beginning of their navigation data messages. For GPS, the beginning of a navigation data message is known as a preamble. For GLONASS, the beginning of a navigation data message is known as a time mark. Within a GNSS constellation, the satellites transmit their navigation data messages synchronously. This means a constellation's satellites will all transmit their preambles/time marks at the same transmission time. There will be some small error in their transmission times because all physical clocks have errors. However, these clock errors are modeled and uploaded into the navigation data messages by government control offices. The receiver applies these satellite clock corrections to the pseudorange measurements to remove the time errors induced by the satellite clocks.

On the right of the figure, the blue blocks (preambles/time marks) enter the receiver at different times. This occurs because each satellite is at a different distance from the receiver. The first blue block that enters the receiver is the earliest preamble/time mark and is assumed to have a pseudorange transit time of 65 to 85 milliseconds, depending on the receiver's elevation. Essentially, the pseudorange measurement for the closest satellite channel is guessed, which results in an approximated receive time for the closest satellite channel. The other satellite channels' pseudoranges are calculated by using the receive time difference between those satellite signals and the earliest satellite signal. The other satellite channels' pseudoranges are relative to the guessed pseudorange for the earliest channel. The transmit time associated to the blue blocks is used to calculate the satellite positions for each channel. A position solution is then estimated with the relative pseudoranges. The position solution can be accurately calculated because the receiver clock bias takes on the bias induced on the relative measurements from assuming the earliest channel's pseudorange. After the position solution is calculated, the receive times for each satellite channel are corrected with the estimated receiver clock bias. The GNSS receiver is now synchronized to the constellation’s time.

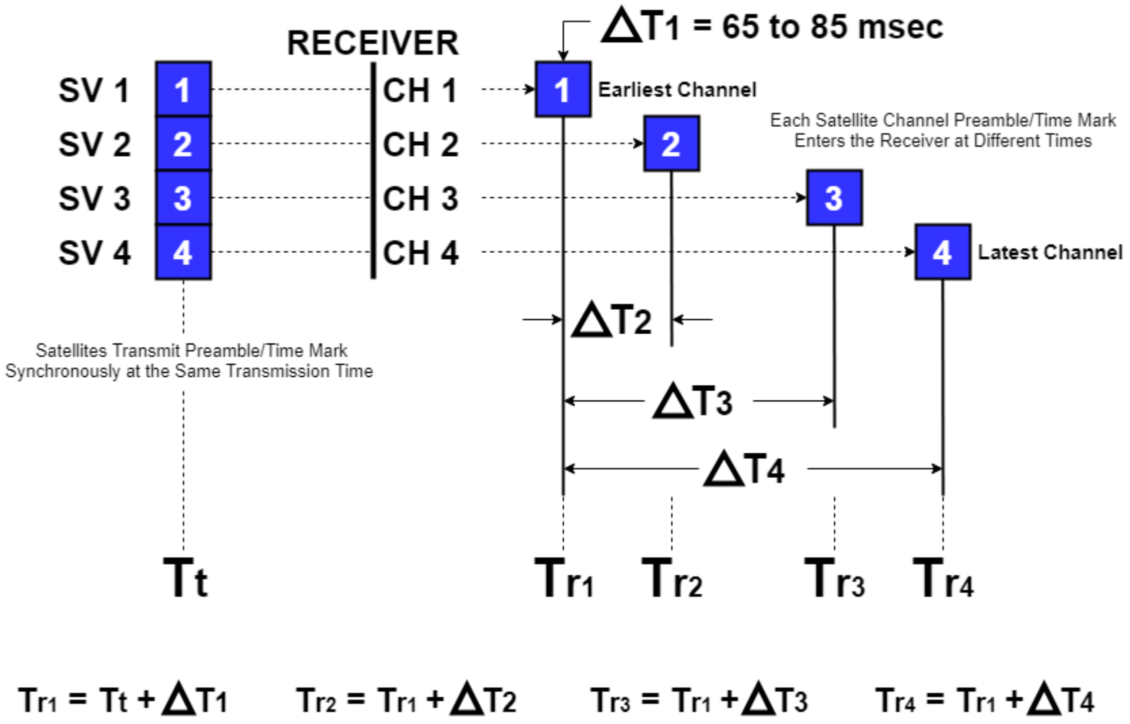

Figure 4.5: Illustration of GNSS Relative Positioning

Relative positioning is a useful tool for GNSS constellation time synchronization. However, it requires at least four satellite channels so the receiver clock bias can be accurately estimated. These four satellite signals must also be in the same constellation because each constellation uses its own time system. After constellation time synchronization, relative positioning can still be used to generate navigation solutions. However, no applicable receivers use relative positioning indefinitely. Relative positioning does not provide true information about the receiver clock bias because it includes bias induced by assuming the earliest channel’s pseudorange. Relative positioning cannot be used with an advanced estimator such as a Kalman filter because the clock bias must be recalculated every period. A Kalmn filter would update the clock bias using a previous bias estimate. Relative positioning also does not provide PVT at any specific receive time, but across a range of receive times. Relative positioning provides an estimated navigation solution at the end of a given set of integration periods that are not associated to any one specific receive time.

Receivers may use the Common Transmission Time (CTT) method for generating pseudoranges and positioning. CTT is nearly identical to relative positioning other than the pseudoranges do not contain the assumed biases after time synchronization. The transmit and receive times are propagated at each tracking period as was shown in Equations (4.2) and (4.3). The channels' pseudoranges are measured at the start/end of the signal tracking integration periods. This results in different receive times, but a common satellite transmission time amongst the measurements. Like relative positioning, CTT cannot specify the PVT estimation at any direct receive time. CTT positioning can be used with multiple GNSS constellations if the signal tracking integration periods are small and equivalent.

Most receivers use the Common Reception Time (CRT) method for generating pseudoranges and positioning. CRT generates pseudorange measurements for all satellite channels at the same signal data sample. This gives all the satellite channels the same receive time, but different transmission times, unlike CTT where the channels' pseudorange measurements are misaligned in the signal data samples. Since CRT pseudoranges do not share a common transmission time, the common signal data sample is associated to different locations in the constellation's synchronous navigation data message for each satellite channel, which makes CRT positioning more complicated. The complication comes from the channels' pseudoranges being measured within a PRN period instead of at the start/end of a PRN period. Figure 4.6 illustrates CRT positoning in the receiver. Since all the pseudoranges are associated with the same signal data sample, there is no receive time ambiguity in the pseudorange measurements. This makes CRT positioning more precise than CTT. Positioning at a common reception time is better suited for real-time, commercial receivers.

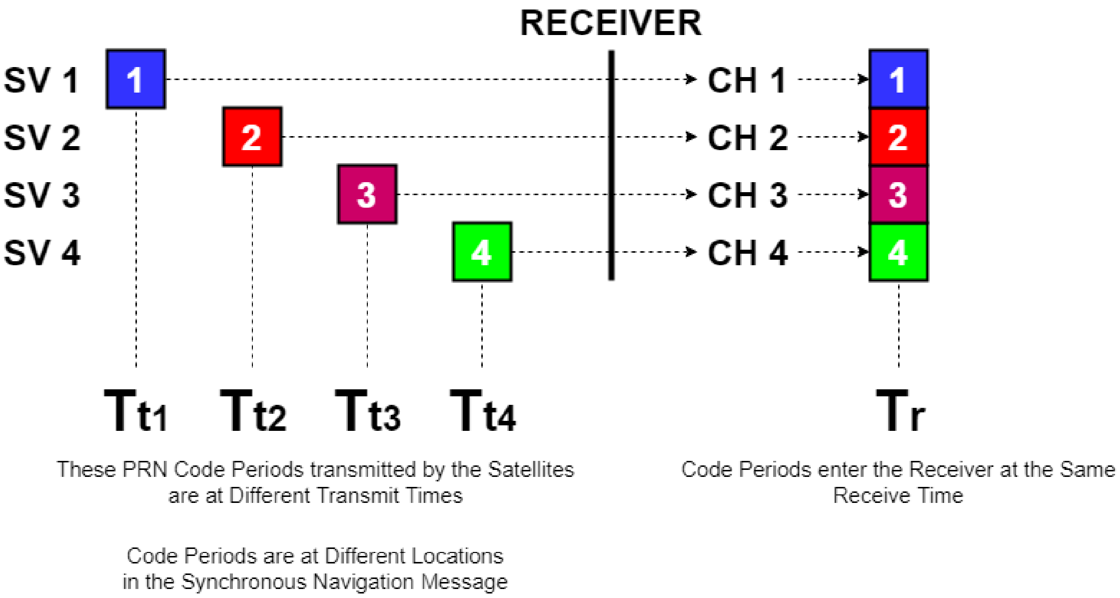

Figure 4.6: Common Reception Time Positioning

Figures 4.7 and 4.8 compare position results when using CTT pseudoranges and CRT pseudoranges. Figure 4.7 shows least squares position results with 9 GPS satellites using the CTT and CRT methods. Figure 4.8 compares the positioning methods using least squares with 6 GLONASS satellites. For both constellations, the position accuracy is similar between CTT and CRT, however, the CRT method is much more precise relative to the CTT method.

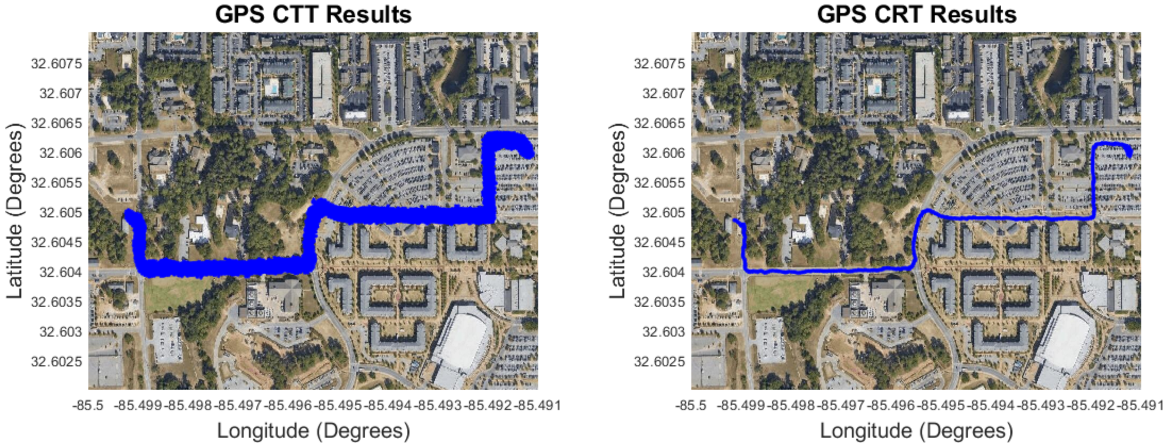

Figure 4.7: CTT and CRT Least Squares Positioning with 9 GPS Satellites

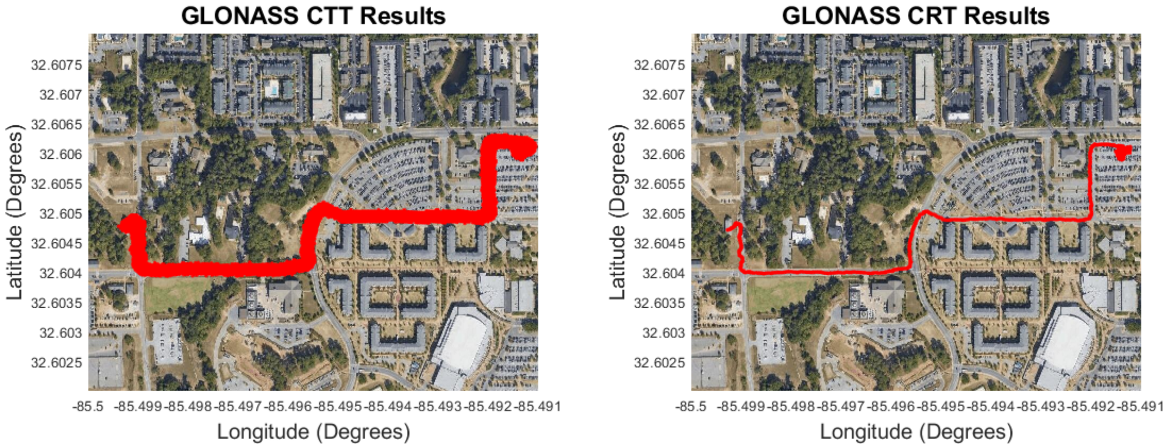

Figure 4.8: CTT and CRT Least Squares Positioning with 6 GLONASS Satellites

4.2 The Pseudorange Rate Measurement

Pseudorange rate measurements are used to calculate the receiver's velocity. The pseudorange rate quanitifies the change in range between the satellite and receiver with respect to time. Each satellite channel provides a pseudorange rate measurement with an estimated carrier Doppler frequency from signal tracking. The Doppler effect is a phenomenon that describes the signal frequency observed by a receiver based on dynamics between the receiver and transmitter. Specifically, the transmitted frequency at the signal source and the line-of-sight velocity between the receiver and transmitter determine the frequency observed by the antenna. Equation (4.5) describes the Doppler effect for electromagnetic signals where f_r is the observed frequency at the receiver, f_t is the transmitted frequency at the transmitter, f_D is the Doppler frequency, $\dot{r}$ is the range rate (line-of-sight velocity) between the receiver and transmitter, and c is the speed of light in a vacuum (the speed of radio signals).

$$f_r = f_t + f_D = f_t \left(1 - \frac{\dot{r}}{c}\right) \tag{4.5}$$

When the receiver and transmitter are moving closer together, the Doppler frequency is positive. When the receiver and transmitter are moving away from each other, the Doppler frequency is negative. When the receiver and transmitter are static relative to each other, the Doppler frequency is zero. Early satellite systems in the 1960's, prior to GPS, used the Doppler effect for ship navigation [45].

Using Equation (4.5), the pseudorange rate measurement can be solved for and is shown in Equation (4.6).

$$\tilde{\dot{\rho}} = -\lambda_{wave} f_D \tag{4.6}$$

In the equation, the pseudorange rate $\tilde{\dot{\rho}}$ is calculated as a product of the satellite's transmitting carrier wavelength λ_{wave} and the estimated Doppler frequency at the receiver f_D. For the GPS L1 C/A signal, the carrier wavelength is approximately $\lambda_{wave} \approx 19$ centimeters. For the GLONASS L1 signal, the carrier wavelength varies due to the FDMA system used in the constellation. Equation (4.6) also requires the negative sign since a positive Doppler frequency is associated with a negative range rate, and a negative Doppler frequency is associated with a positive range rate.

The pseudorange rate measurement for a satellite channel fully depends on the Doppler frequency estimate from carrier tracking, which contains estimation error. The pseudorange rate contains zero-mean noise errors due to thermal noise in the receiver's front-end hardware and signal noise from the environment. The variation on this noise depends on the PLL or FLL noise bandwidth used in signal tracking [53]. High noise bandwidths will cause more noise to be seen on the Doppler frequency estimate and the pseudorange rate. Low noise bandwidths will cause less noise to be seen on the Doppler frequency and the pseudorange rate. However, if the noise bandwidth becomes too low, the Doppler frequency estimate may become inaccurate, which also means the psuedorange rate estimate will be inaccurate. The pseudorange rate measurement does not suffer as significanlty from atmospheric errors as the pseudorange measurement does. This is due to the ionospheric and tropospheric delays' variation with respect to time being small, staying nearly constant. The pseudorange rate does contain a clock drift bias error due to receiver's oscillator, similar to the clock bias error seen on the pseudorange measurement. These pseudorange rate measurement errors were expressed in Equation (3.15). The next section discusses the receiver clock drift in more detail.

4.2.1 Receiver Clock Drift

The pseudorange rate does not require timing to generate the measurement, but still incurs a clock error; this is due to the nature of oscillators. The receiver's oscillator (clock) sampling frequency error is the receiver clock drift. The sampling frequency error biases the observed Doppler frequency in carrier tracking, which then biases the pseudorange rate measurement. Equation (4.7) describes the relationship between the receiver clock's sampling frequency error and the GNSS estimated receiver clock drift.

$$f_{s_{error}} = \frac{c\dot{b}}{\lambda_{wave}} = f_{L1}\dot{b} \tag{4.7}$$

In the equation, $f_{s_{error}}$ is the clock's sampling frequency error in Hz, c is the speed of light in meters per second, $\dot{b}$ is the GNSS estimated receiver clock bias in seconds per seconds (unitless), and λ_{wave} is the transmitted carrier wavelength for the satellite signal in meters per cycle. The sampling frequency error can also be expressed with the constellation's L1 transmitted carrier frequency f_{L1} in Hz.

Since the receiver's oscillator samples all the satellite signals with the same sampling frequency error, the receiver clock drift bias is the same for all the pseudorange rate measurements. This allows the receiver clock drift along with velocity to be solved for when there are at least four satellite signals being accurately tracked. The receiver clock drift is also the velocity of the receiver clock bias, which determines how the receiver clock bias grows overtime. An error in frequency will drive phase error growth. Phase error is directly proportional to time error, which induces the receiver clock bias observed on pseudorange measurements.

The receiver clock drift dynamics depend on the quality of the receiver's oscillator. A TCXO generally has a large receiver clock drift that induces the fast growth of the receiver clock bias. The TCXO clock drift may grow as well, indicating receiver clock acceleration or growth in the oscillator's sampling frequency error. An OCXO has a small receiver clock drift and little receiver clock acceleration. Rubidium and Cesium atomic clocks have zero-mean receiver clock drifts and very little or no receiver clock acceleration. From these descriptions, the TCXO is considered "inaccurate and unstable", the OCXO is considered "inaccurate and stable", and the atomic clocks are considered "accurate and stable". Figure 4.9 shows receiver clock drifts in velocity units for the oscillators described above.

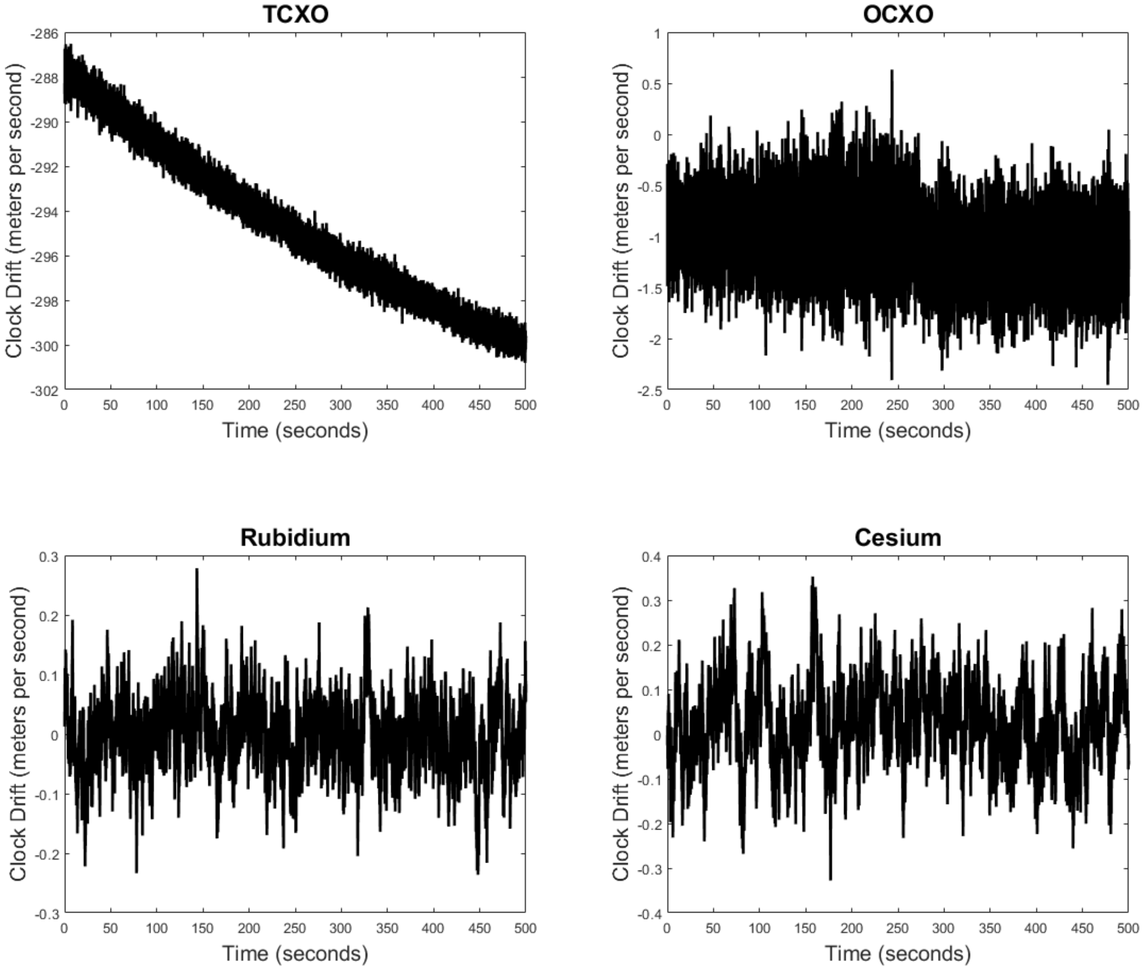

Figure 4.9: Receiver Clock Drifts for Different Types of Oscillators

Along with accurate time estimation using the receiver clock bias, GNSS receivers are heavily demanded in timing services with respect the receiver clock drift. Clock drift estimates can be used to stabilize voltage controlled oscillators for time or other applications. All oscillators used to generate signals contain frequency error. Applications such as radar, communications, controllers, and sensors that rely on signal transmission may not work effectively in the presence of frequency error. GNSS receivers are used in many situations to estimate this oscillator drift so it can be corrected [18].

4.2.2 Pseudorange Rate Multipath Bias

Like the pseudorange measurement, the pseudorange rate measurment can become biased from multipath. Multipath signals do not only contain code reflections, but carrier reflections that can distort the receiver's carrier tracking operations. When entering the receiver antenna, the multipath carrier phase will be delayed with respect to the line-of-sight signal, since multipath reflections travel along longer paths. Reflections can also alter the carrier frequency of a multipath signal from the line-of-sight signal.

For the pseudorange measurement, the range bias was induced in PRN code tracking when the receiver replica code correlated with the delayed multipath code. Correlation between the replica carrier and the multipath carrier induces biases on the carrier discriminators that leak into the pseudorange rate measurement. Figure 4.10 illustrates how the pseudorange rate measurement becomes biased in carrier tracking. The replica carrier contains correlation with both the true carrier and the multipath carrier. In this figure, there is a carrier phase error between the replica and GPS signal, and there is a carrier frequency error between the replica and multipath signal. These carrier tracking errors will alter the receiver replica, which will alter the pseudorange rate measurement. The multipath bias on the pseudorange rate measurement can bias the receiver velocity estimation and possibly the receiver position estimation.

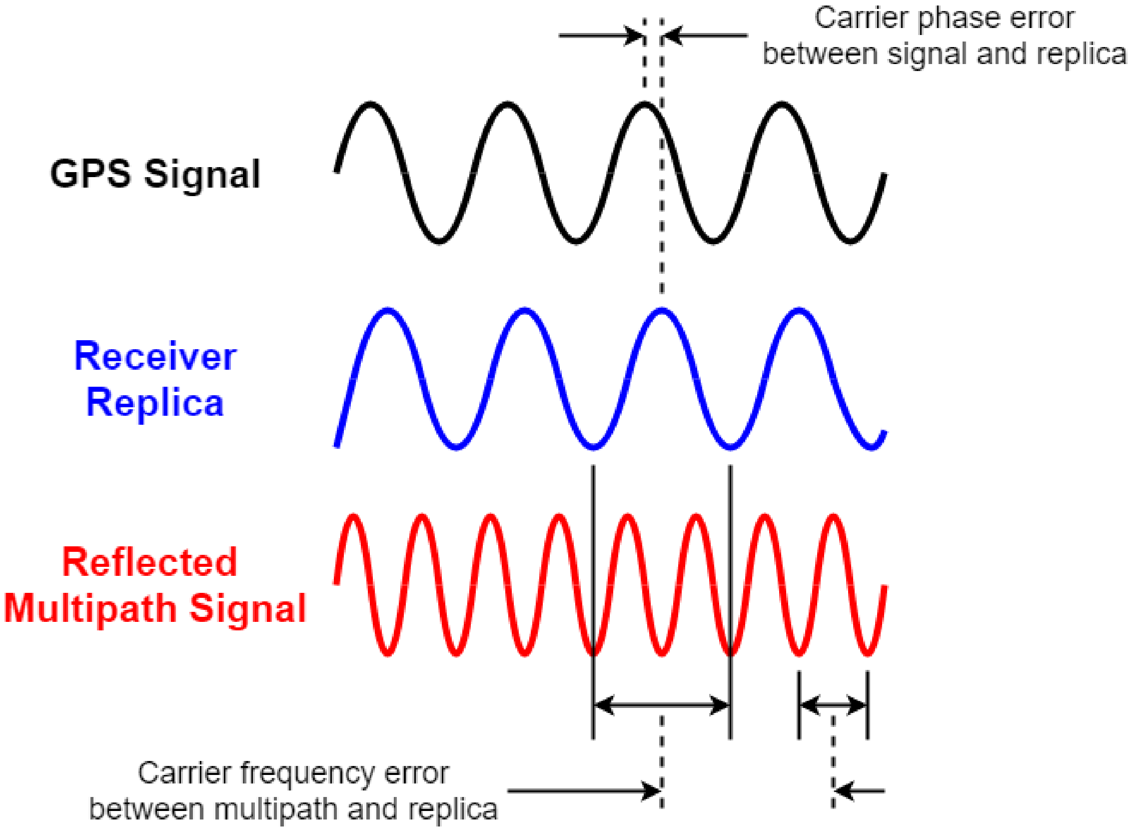

Figure 4.10: Correlation with the LOS Satellite Carrier and a Reflected Multipath Carrier

The pseudorange rate bias induced by multipath can be mathematically modeled using the signal tracking correlator model from Equation (3.3). To determine the range-rate bias, correlators between the replica and the GPS signal are calculated and added to correlators between the replica and the multipath signal [5]. These tracking correlators that contain the mulitpath signal are then used to calculate the carrier frequency discriminator from Equation (3.11). This frequency discriminator represents the mulitpath bias observed on the pseudorange rate measurement when the replica is accurately tracking the true satellite carrier.

Figure 4.11 shows the pseudorange rate multipath bias as a function of carrier phase error and carrier frequency error between the multipath signal and the line-of-sight signal. The GPS signal is at a carrier-to-noise ratio of 42 dB-Hz, and the multipath signal is at a carrier-to-noise ratio of 39 dB-Hz. In other words, the GPS signal has twice the power of the multipath reflection. The representation of the pseudorange rate bias is the frequency discriminator, which was calculated with a signal tracking integration period of $T = 0.02$ seconds.

From the figure, it is shown that the pseudorange rate bias is at its greatest when the carrier phase error is half a cycle (180 degrees). Increasing the carrier frequency error also magnifies the bias. However, if the carrier frequency is increased further (approximately more than 10 Hz), the bias drops because the correlation between the replica and multipath signal will be low. When the carrier errors are low, the multipath bias is nearly zero because the multipath carrier is very similar to the line-of-sight carrier. As seen with the pseudorange and PRN code, multipath tracking errors of the carrier must be optimally placed to maximize the pseudorange rate bias. If the tracking errors are too small or too big, the multipath bias will be approximately zero.

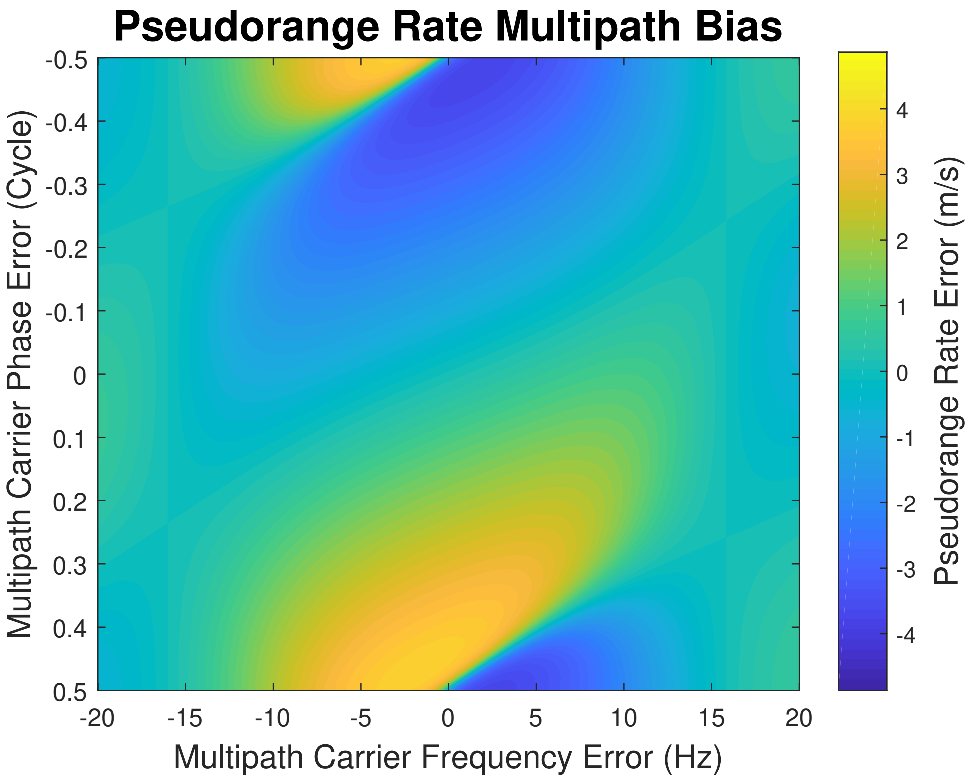

Figure 4.11: Multipath Pseudorange Rate Bias when Signal C/N$_O$ $=$ 42 dB-Hz and Multipath C/N$_O$ $=$ 39 dB-Hz

Figure 4.12 shows the pseudorange rate multipath bias when the line-of-sight and multipath signals both have carrier-to-noise ratios of 42 dB-Hz. The overall shape of the function is the same, but with greater magnitude. As the multipath signal power increases, the correlation between the replica and multipath carriers increases for a given set of tracking errors, thus inducing a greater bias on the pseudorange rate measurement. Note that the increase in pseudorange rate bias with respect to mulitpath signal power is not as significant as it was with the pseudorange in Figure 4.4.

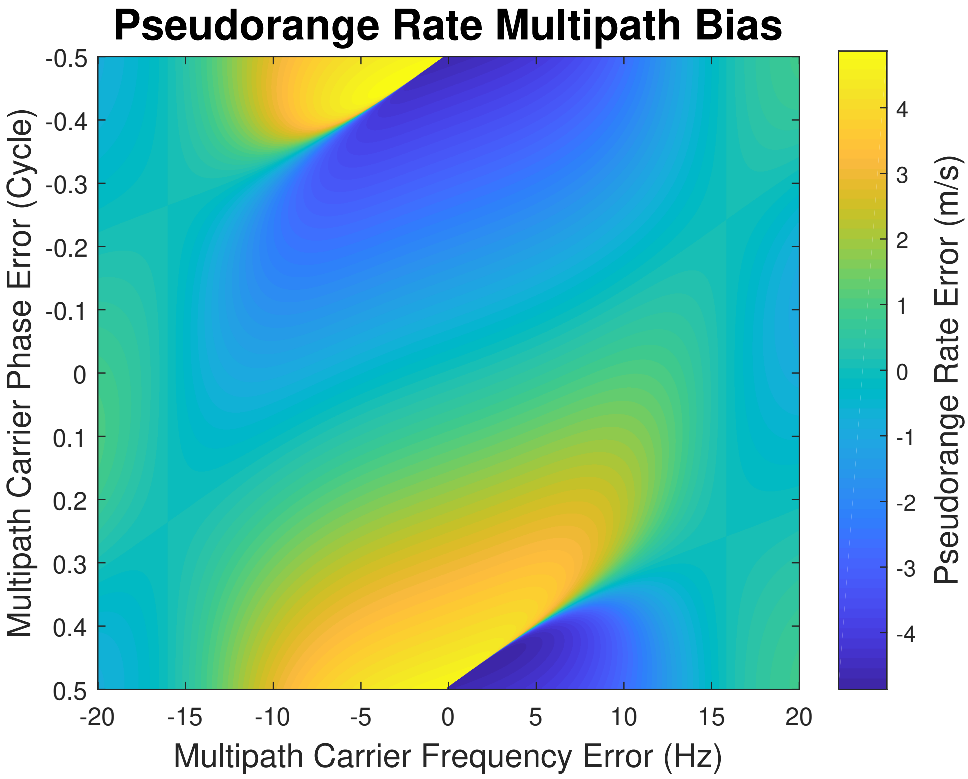

Figure 4.12: Multipath Pseudorange Rate Bias when Signal C/N$_\text{O}$ $= 42$ dB-Hz and Multipath C/N$_\text{O}$ $= 42$ dB-Hz

4.3 Satellite Position, Velocity, and Clock Correction

Satellite position and velocity are necessary to estimate the receiver PVT. Pseudorange measurements determine receiver position by reference of the satellite positions, and pseudorange rate measurements determine receiver velocity by reference of the satellite velocities. Satellite position and velocity are determined by an ephemeris that is decoded from a satellite signal's navigation data message. The position and velocity are determined in an ECEF coordinate frame. GPS uses the WGS84 frame, and GLONASS uses the PZ-90.11 frame. Along with satellite position and velocity, the ephemeris also provides parameters to generate a satellite clock correction. Although highly stable, satellite atomic clocks do drift over time, which induces a transmission time error that must be accounted for in the pseudorange measurement. Equation (4.8) describes the correction where $\tilde{\rho}$ is the measured pseudorange in meters, τ_{sv} is the satellite clock correction in seconds, c is the speed of light in meters per second, and $\tilde{\rho}_{corr}$ is the time corrected pseudorange measurement.

$$\tilde{\rho}_{corr} = \tilde{\rho} + c\tau_{sv} \tag{4.8}$$

The next sections discuss the calculation of satellite position, velocity, and clock correction using the ephemeris parameters from the GPS and GLONASS L1 data messages.

4.3.1 GPS Satellite Calculation

The GPS L1 C/A navigation data ephemeris provides a set of orbital and time parameters to determine the satellite position, velocity, and clock correction. These ephemeris parameters are shown in Table 4.1. The orbital parameters are used to model an orbit that describes the satellite motion for a duration of time. Typically, the orbital parameters are accurate over a two hour period, and the GPS navigation data message updates the ephemeris every two hours. The time parameters are used to determine where the satellite is located on the orbit model along with the satellite clock correction. A series of calculations are required to determine GPS satellite position and velocity in the WGS84 ECEF coordinate frame.

Table 4.1: Satellite Ephemeris Parameters from GPS Navigation Data Message [44]

Variable	**Description**	**Units**
M_0	Mean Anomaly at Reference Time	semi-circles
Δn	Mean Motion Difference from Computed Value	semi-circles/second
e	Eccentricity	dimensionless
$\sqrt{A}$	Square-Root of Semi-Major Axis	$\sqrt{\text{meters}}$
Ω_0	Longitude of Ascending Node	semi-circles
i_0	Inclination Angle	semi-circles
ω	Argument of Perigee	semi-circles
$\dot{\Omega}$	Right Ascension Rate	semi-circles/second
$IDOT$	Inclination Angle Rate	semi-circles/second
C_{uc}	Cosine Harmonic Correction Term to the Arg. of Latitude	radians
C_{us}	Sine Harmonic Correction Term to the Arg. of Latitude	radians
C_{rc}	Cosine Harmonic Correction Term to the Orbit Radius	meters
C_{rs}	Sine Harmonic Correction Term to the Orbit Radius	meters
C_{ic}	Cosine Harmonic Correction Term to the Inclination Angle	radians
C_{is}	Sine Harmonic Correction Term to the Inclination Angle	radians
t_{oe}	Reference Ephemeris Time	seconds
t_{oc}	Clock Correction Reference Time	seconds
a_{f0}	Time Clock Correction	seconds
a_{f1}	Time Drift Clock Correction	dimensionless
a_{f2}	Time Acceleration Clock Correction	seconds^{-1}
T_{GD}	Group Delay Differential	seconds

Prior to performing satellite calculations, the correct satellite time t_k must be determined from the satellite transmission time t_t. The calculations to find t_k are shown in Equations (4.9) through (4.11).

$$dt = t_t - t_{oc} \tag{4.9}$$

$$corr = a_{f2}dt^2 + a_{f1}dt + a_{f0} - T_{GD} \tag{4.10}$$

$$t_k = t_t - corr - t_{oe} \tag{4.11}$$

Note that dt must be checked for week crossovers before using it to calculate $corr$. If dt is greater than 302,400 seconds then 604,800 seconds must be subtracted from dt. If dt is less than -302,400 seconds then 604,800 seconds must be added to dt. Week crossover must also be checked for t_k.

Equations (4.12) through (4.18) provide corrections to the orbital parameters to calculate the GPS satellite position.

$$n = \sqrt{\frac{\mu}{\sqrt{A}^6}} + \Delta n \tag{4.12}$$

$$M_k = M_0 + nt_k = E_k - e \sin E_k \tag{4.13}$$

$$\Phi_k = \arctan\left(\frac{\sqrt{1-e^2}\sin E_k/(1-e\cos E_k)}{(\cos E_k - e)/(1-e\cos E_k)}\right) + \omega \tag{4.14}$$

$$u_k = \Phi_k + C_{us}\sin(2\Phi_k) + C_{uc}\cos(2\Phi_k) \tag{4.15}$$

$$r_k = \sqrt{A}^2(1-e\cos E_k) + C_{rs}\sin(2\Phi_k) + C_{rc}\cos(2\Phi_k) \tag{4.16}$$

$$i_k = i_0 + C_{is}\sin(2\Phi_k) + C_{ic}\cos(2\Phi_k) + (IDOT)t_k \tag{4.17}$$

$$\Omega_k = \Omega_0 + (\dot{\Omega} - \dot{\Omega}_e)t_k - \Omega_e t_{oe} - \dot{\Omega}_e\frac{\tilde{\rho}}{c} \tag{4.18}$$

Equation (4.12) uses the Earth gravitational constant $\mu = 3.986005 \text{ x } 10^{14}\ \frac{\text{m}^3}{\text{s}^2}$. Note that the eccentric anomaly E_k can be solved iteratively using Equation (4.13). Equation (4.18) uses the Earth's rotation rate $\dot{\Omega}_e = 7.2921151467 \text{ x } 10^{-5}\ \frac{\text{rad}}{\text{s}}$ and the pseudorange transit time measurement $\tilde{\rho}$ / c. Note that some implementations do not use the transit time when determining the longitude of ascending node Ω_k.

Equations (4.19) through (4.23) calculate the ECEF x, y, and z-direction GPS satellite positions (x_{sv}, y_{sv}, and z_{sv}) using the corrected orbital parameters.

$$x_{orb} = r_k\cos(u_k) \tag{4.19}$$

$$y_{orb} = r_k \sin(u_k) \tag{4.20}$$

$$x_{sv} = x_{orb} \cos(\Omega_k) - y_{orb} \cos(i_k) \sin(\Omega_k) \tag{4.21}$$

$$y_{sv} = x_{orb} \sin(\Omega_k) + y_{orb} \cos(i_k) \cos(\Omega_k) \tag{4.22}$$

$$z_{sv} = y_{orb} \sin(i_k) \tag{4.23}$$

This satellite position algorithm can also be found in the GPS L1 C/A ICD [44]. The GPS ICD does not provide any information about satellite velocity calculations, however, the equations used to determine satellite position can be differentiated to determine satellite velocity. Equations (4.24) through (4.29) show the corrected orbital parameter rates required to calculate GPS satellite velocity.

$$\dot{E}_k = \frac{n}{1 - e\cos(E_k)} \tag{4.24}$$

$$\dot{\Phi}_k = \frac{\sqrt{1-e^2}}{1 - e\cos(E_k)} \dot{E}_k \tag{4.25}$$

$$\dot{u}_k = (1 + 2C_{us} \cos(2\Phi_k) - 2C_{uc} \sin(2\Phi_k))\dot{\Phi}_k \tag{4.26}$$

$$\dot{r}_k = 2(C_{rs} \cos(2\Phi_k) - C_{rc} \sin(2\Phi_k))\dot{\Phi}_k + \sqrt{A}^2 e \sin(E_k)\dot{E}_k \tag{4.27}$$

$$IDOT_{corr} = 2(C_{is} \cos(2\Phi_k) - C_{ic} \sin(2\Phi_k))\dot{\Phi}_k + IDOT \tag{4.28}$$

$$\dot{\Omega}_{corr} = \dot{\Omega} - \dot{\Omega}_e \tag{4.29}$$

Equations (4.30) through (4.34) calculate the ECEF x, y, and z-direction GPS satellite velocities ($\dot{x}_{sv}$, $\dot{y}_{sv}$, and $\dot{z}_{sv}$) using the corrected orbital parameter rates and the parameters for determining satellite position.

$$\dot{x}_k = \dot{r}_k \cos(u_k) - r_k \sin(u_k)\dot{u}_k \tag{4.30}$$

$$\dot{y}_k = \dot{r}_k \sin(u_k) + r_k \cos(u_k)\dot{u}_k \tag{4.31}$$

$$\dot{x}_{sv} = \dot{x}_k \cos(\Omega_k) - \dot{y}_k \cos(i_k) \sin(\Omega_k) + y_{orb} \sin(i_k) \sin(\Omega_k) IDOT_{corr} - y_{sv}\dot{\Omega}_{corr} \tag{4.32}$$

$$\dot{y}_{sv} = \dot{x}_k \sin{(\Omega_k)} + \dot{y}_k \cos{(i_k)} \cos{(\Omega_k)} - y_{orb} \sin{(i_k)} \cos{(\Omega_k)} IDOT_{corr} + x_{sv} \dot{\Omega}_{corr} \tag{4.33}$$

$$\dot{z}_{sv} = \dot{y}_k \sin{(i_k)} + y_{orb} \cos{(i_k)} IDOT_{corr} \tag{4.34}$$

The GPS satellite clock correction calculation is shown by Equation (4.35).

$$\tau_{sv} = a_{f2} dt^2 + a_{f1} dt + a_{f0} - T_{GD} + Fe\sqrt{A} \sin{(E_k)} \tag{4.35}$$

The last term in the equation accounts for relativity effects, due to the satellite's far distance from Earth's gravity and the satellite's high velocity. The relativistic constant is $F = -4.442807633 \text{ x } 10^{-10} \frac{\text{s}}{\sqrt{\text{m}}}$.

4.3.2 GLONASS Satellite Calculation

Instead of providing orbital parameters like GPS, the GLONASS L1 navigation data ephemeris provides satellite position and velocity at time t_b. Using this information, the satellite position and velocity can be integrated from time t_b to the current transmission time t_t. The ephemeris also provides time parameters to determine the satellite clock correction. Table 4.2 lists the parameters given by the GLONASS ephemeris. The ephemeris also provides lunar and solar perturbations that affect the satellite's acceleration.

Table 4.2: Satellite Ephemeris Parameters from GLONASS Navigation Data Message [55] and [56]

Variable	**Description**	**Units**
$x_{sv}(t_b)$	Satellite x-Position at time t_b	km
$y_{sv}(t_b)$	Satellite y-Position at time t_b	km
$z_{sv}(t_b)$	Satellite z-Position at time t_b	km
$\dot{x}_{sv}(t_b)$	Satellite x-Velocity at time t_b	km/s
$\dot{y}_{sv}(t_b)$	Satellite y-Velocity at time t_b	km/s
$\dot{z}_{sv}(t_b)$	Satellite z-Velocity at time t_b	km/s
$\ddot{x}_{ls}$	Satellite x-Acceleration due to Lunar and Solar Perturbations	km/s^2
$\ddot{y}_{ls}$	Satellite y-Acceleration due to Lunar and Solar Perturbations	km/s^2
$\ddot{z}_{ls}$	Satellite z-Accleration due to Lunar and Solar Perturbations	km/s^2
t_b	Satellite State Reference Time	minutes
γ_n	Carrier Frequency Deviation	dimensionless
τ_n	Time Clock Correction	seconds

The GLONASS ephemeris may provide the satellite information and reference time before or ahead of the satellite transmission time. The GLONASS ICD states the initial satellite position and velocity can be integrated for a duration of 15 minutes before or ahead of the satellite state reference time t_b to obtain an accurate satellite position and velocity. After 15 minutes of integration, the satellite solution will degrade. Using these standards, the GLONASS ephemeris can be used for a total of 30 minutes. Generally, the GLONASS navigation data message updates the ephemeris parameters every 30 minutes. With this update rate, the GLONASS satellite position and velocity can be determined at any transmission time. This concept is illustrated by Figure 4.13 where three satellite state reference times are shown as GLONASS transmission time propagates. The shaded areas indiciate the 15 minutes of integration before or after t_b that can be used to obtain accurate satellite information.

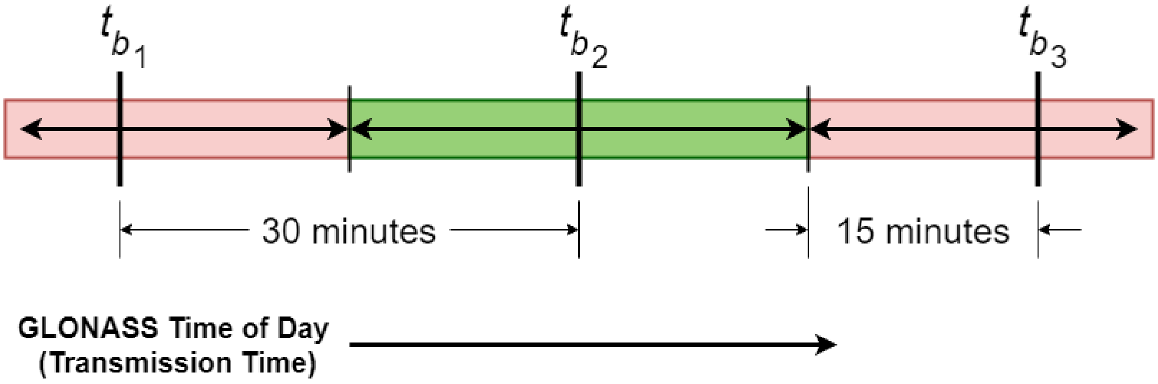

Figure 4.13: GLONASS Time of Day with Respect to Satellite State Reference Times

To determine satellite position and velocity, the time difference between the reference time t_b and the transmission time t_t must be determined and checked for a day crossover. This time difference dt is shown by Equation (4.36).

$$dt = t_t - t_b \tag{4.36}$$

If dt is greater than 85,500 seconds then 86,400 seconds must be subtracted from dt. If dt is less than -85,500 seconds then 86,400 seconds must be added to dt.

GLONASS satellite position (x_{sv}, y_{sv}, and z_{sv}) and velocity ($\dot{x}_{sv}$, $\dot{y}_{sv}$, and $\dot{z}_{sv}$) in the x, y, and z directions are determined in the PZ-90.11 ECEF coordinate frame by integrating satellite acceleration over time. Equations (4.37) through (4.40) are used to determime satellite acceleration ($\ddot{x}_{sv}$, $\ddot{y}_{sv}$, and $\ddot{z}_{sv}$), which come from [8] and [62].

$$\ddot{x}_{sv} = -\frac{\mu}{r^3}x_{sv} + \frac{3}{2}J_0^2\frac{\mu a_e^2}{r^5}x_{sv}\left(1 - \frac{5z_{sv}^2}{r^2}\right) + \omega_e^2 x_{sv} + 2\omega_e\dot{y}_{sv} + \ddot{x}_{ls} \tag{4.37}$$

$$\ddot{y}_{sv} = -\frac{\mu}{r^3}y_{sv} + \frac{3}{2}J_0^2\frac{\mu a_e^2}{r^5}y_{sv}\left(1 - \frac{5z_{sv}^2}{r^2}\right) + \omega_e^2 y_{sv} - 2\omega_e\dot{x}_{sv} + \ddot{y}_{ls} \tag{4.38}$$

$$\ddot{z}_{sv} = -\frac{\mu}{r^3}z_{sv} + \frac{3}{2}J_0^2\frac{\mu a_e^2}{r^5}z_{sv}\left(3 - \frac{5z_{sv}^2}{r^2}\right) + \ddot{z}_{ls} \tag{4.39}$$

$$r = \sqrt{x_{sv}^2 + y_{sv}^2 + z_{sv}^2} \tag{4.40}$$

Note that satellite acceleration is a function of satellite position and velocity. The GLONASS ICD recommends performing this integration with the Runge-Kutta 4th-order method. Euler's integration does not provide enough accuracy over time due to rounding errors. The GLONASS satellite accelerations in Equations (4.37) through (4.39) are also a function of three PZ-90.11 Earth parameter constants, which are listed in Table 4.3.

Table 4.3: PZ-90.11 Earth Parameter Constants

Variable	**Description**	**Value**
μ	Earth Gravitational Constant	$398600.44\ \frac{\mathrm{km}^3}{\mathrm{s}^2}$
J_0^2	Second Zonal Harmonic of the Geopotential	1082625.7 x 10^{-9}
ω_e	Earth Rotation Rate	0.7292115 x $10^{-4}\frac{\mathrm{rad}}{\mathrm{s}}$

The GLONASS satellite clock correction τ_{sv} is calculated using Equation (4.41).

$$\tau_{sv} = -\tau_n + \gamma_n dt \tag{4.41}$$

The clock correction is a function of the ephemeris time parameters and the crossover checked time difference dt.

4.4 Carrier-to-Noise Power Density Ratio

The carrier-to-noise power density (C/N$_\mathrm{O}$) ratio is a representation of received signal strength or a representation of how noisy the signal is. The C/N$_\mathrm{O}$ is often confused with the signal-to-noise power ratio (SNR). A signal's SNR is defined as the ratio of signal variance to noise variance. Note that signal variance and noise variance are equivalent to signal power and noise power. A signal's C/N$_\mathrm{O}$ is defined as the ratio of signal power to noise density variance. GNSS receivers use C/N$_\mathrm{O}$ over SNR because receivers cannot estimate the noise power, but they can estimate the noise power density. Another way to compare C/N$_\mathrm{O}$ and SNR is as follows: the SNR is the ratio of satellite signal power to the noise power, while the C/N$_\mathrm{O}$ is the the ratio of satellite signal power to the noise power for a given single Hertz of noise bandwidth. In GNSS signal analysis, the noise bandwidth is assumed infinite. The mathematical relationship between the C/N$_0$ and the SNR is shown in Equation (4.42) where the C/N_o is in dBW-Hz, the SNR is in dBW, and $B_{n_{rcvr}}$ is the receiver equivalent noise bandwidth in dB.

$$C/N_o = SNR + B_{n_{rcvr}} \tag{4.42}$$

The receiver equivalent noise bandwidth depends on the receiver's hardware front-end. Typically, the SNR is 60 to 70 dB lower than the C/N$_0$ [1].

Most GNSS receivers estimate the satellite channels' C/N$_0$ ratios. A channel's C/N$_0$ ratio can be used as a measure to determine if the receiver is accurately tracking the signal. More importantly, C/N$_0$ ratios can be used to weigh the satellite channels' navigation measurements when estimating receiver PVT. Weighing the measurements places more trust in strong signals (high C/N$_0$ ratios) and less trust in noisy signals (low C/N$_0$ ratios) within the estimator. Using today's antenna and front-end technology, satellite signals have C/N$_0$ ratios between 40 to 50 dBW-Hz. Signals with C/N$_0$ ratios below 35 dBW-Hz are considered very noisy.

4.4.1 C/N$_0$ Estimation

A satellite channel's C/N$_0$ ratio is typically estimated during signal tracking. There have been many proposals to C/N$_0$ ratio estimation; the one provided in this work comes from [31]. The estimation approach is derived from the correlator model in Equation (3.3). To determine a channel's C/N$_0$ ratio, measurements of the channel's signal power $\tilde{A}^2$ and noise variance $\tilde{\eta}^2$ are required. The signal power is measured by using the early and late signal tracking correlators. Equation (4.43) provides the calculation of the satellite channel's measured signal power $\tilde{A}^2$ for a given integration period.

$$\tilde{A}^2 = (IE + IL)^2 + (QE + QL)^2 \tag{4.43}$$

In the equation, IE is the in-phase early correlator, IL is the in-phase late correlator, QE is the quadrature early correlator, and QL is the quadrature late correlator. The squares in the equation ensure the carrier power (not the carrier amplitude) is estimated.

The satellite channel's noise variance $\tilde{\eta}^2$ is measured using a series of early and late noise correlators during each signal tracking integration period. These noise correlators are generated using PRN replicas with very large chip spacings ($>>$ 1 chip). From the PRN autocorrelation function in Equation (2.1), correlation between the replica and signal is zero when the chip offset is greater than 1 chip, leaving only noise in the correlation. Using more noise correlators with different chip spacings provides a more accurate measurement of the channel's noise variance. Note that for each chip spacing used, an in-phase early, in-phase late, quadrature early, and quadrature late correlator is generated and used in the noise variance calculation. The carrier replicas must be included when generating the noise correlators to peel off the signal's carrier so only noise is represented in the correlation. Generating correlators is computationally intensive. Noise variance estimation places a significant expense on the receiver's processor to perform the calculation.

In this work, noise variance is estimated using 12 PRN spacings, resulting in 48 noise correlators per satellite channel. The GPS and GLONASS PRN codes have differing chip lengths, which gives their noise correlators different chip spacings. Table 4.4 shows the 12 chip spacings used for GPS and GLONASS channels.

Table 4.4: PRN Chip Spacings for GPS and GLONASS Noise Correlators

GPS	50	110	150	225	310	400	495	600	720	805	900	980
GLONASS	35	80	110	140	200	250	305	365	420	445	470	510

With measurements of the signal power $\tilde{A}^2$ and noise variance $\tilde{\eta}^2$, the C/N_o ratio is calculated by Equation (4.44).

$$C/N_o = 10\log_{10}\left(\frac{\tilde{A}^2 - 4\tilde{\eta}^2}{2T\tilde{\eta}^2}\right) \tag{4.44}$$

Since the signal power is calculated with four signal tracking correlators that contain noise, this noise is removed with the noise variance estimation in the numerator scaled by four. In the equation's denominator, the signal tracking integration period T is required for appropriate scaling. Larger integration periods will increase the magnitude of $\tilde{A}^2$ and $\tilde{\eta}^2$. The ratio is then converted to decibels with the $\log_{10}$ operation and the multiplication of 10.

4.4.2 Navigation Measurement Variance

As stated earlier, the satellite channels' C/N_O ratios can be used to weigh their pseudorange and pseudorange rate measurements, which provides better PVT estimation accuracy. Weighing the measurements is performed by estimating the measurements' variances. There are different implementations of measurement variance estimation, all which operate similarly, but vary in complexity. This work uses a simple method for measurement variance estimation that comes from [61]. Equation (4.45) shows the estimated variance for the pseudorange residual $\sigma^2_{\delta\rho}$, and Equation (4.46) shows the estimated variance for the pseudorange rate residual $\sigma^2_{\delta\dot{\rho}}$.

$$\sigma^2_{\delta\rho} = \frac{\lambda^2_{chip}}{2T^2(C/N_o)^2} + \frac{\lambda^2_{chip}}{4T(C/N_o)} \tag{4.45}$$

$$\sigma^2_{\delta\dot{\rho}} = K\left(\frac{\lambda_{wave}}{\pi T}\right)\left(\frac{2}{T^2(C/N_o)^2} + \frac{2}{T(C/N_o)}\right) \tag{4.46}$$

Both equations are functions of the channel's C/N_o ratio and the signal tracking integration period T. The pseudorange residual $\delta\rho$ variance uses the signal's PRN chip width λ_{chip}, and the pseudorange rate residual $\delta\dot{\rho}$ uses the signal's carrier wavelength λ_{wave}. The pseudorange rate residual variance can be very small, so a scaling constant K can be applied to the variances to increase their magnitudes, which increases the pseudorange rate variances when using a Kalman filter implementation.

The least squares estimator is improved by weighing the navigation measurements with the variances described above. This implementation is known as weighted least squares, and its calculation is shown in Equation (4.47) where $\Delta\underline{\hat{x}}$ is the estimated error in receiver PVT, H is the measurement model from Equation (3.24), and Y is the measurement array from Equation (3.25).

$$\Delta\underline{\hat{x}} = (H^T R^{-1} H)^{-1} H^T R^{-1} Y \tag{4.47}$$

The variable R is the measurement covariance matrix that applies the weighting to the satellite channel measurements. Equation (4.48) shows the measurement covariance matrix for m satellite channels.

$$R = \begin{bmatrix} \sigma^2_{\delta\rho_1} & 0 & 0 & \cdots & 0 \\ 0 & \sigma^2_{\delta\rho_m} & 0 & \cdots & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & \cdots & \sigma^2_{\delta\dot{\rho}_1} & 0 \\ 0 & 0 & \cdots & 0 & \sigma^2_{\delta\dot{\rho}_m} \end{bmatrix} \tag{4.48}$$

The pseudorange and pseudorange rate variances are along the diagonal indicating that all covariances between the measurements are zero. The measurement errors/noise of one satellite channel are not correlated to those of another satellite channel.

4.5 Conclusion

This chapter discussed the processes required to extract information from signal tracking for purposes of navigation in a GNSS receiver. The relationship between PRN code tracking and the pseudorange measurement was explained. The relationship between carrier tracking and the pseudorange rate measurement was also explained. The errors that can occur in the measurements and their effects on navigation were summarized. The chapter briefly touches on determining satellite position and velocity from GPS and GLONASS ephemerides, a necessity to navigation. Variance weighting of a satellite channel's measurements based on its estimated C/N$_O$ ratio was provided for the purpose of enhancing a receiver's navigation processor. The next chapter describes how the relationship between signal tracking and navigation can be exploited into an algorithm that can enhance a receiver's performance. This algorithm is known as vector tracking.

Chapter 5

Vector Tracking Loops

As stated throughout the thesis, signal tracking is the central component to GNSS receivers. Tracking satellite signals generates navigation measurements and extracts the channels' data messages, so the receiver can perform positioning and timing. Most receivers today operate with scalar tracking, where individual DLL's are used for code tracking and individual PLL's/FLL's are used for carrier tracking on each satellite channel. Specifically, each channel in the receiver uses loop filters to track the satellite signals' dynamics. Code tracking generates pseudorange measurements for positioning, and carrier tracking generates pseudorange rate measurements for velocity determination. The clock bias must also be accounted for in the position estimation and the clock drift in the velocity estimation. GNSS scalar tracking was discussed in Chapter 3. Figure 5.1 provides a diagram of a GNSS scalar receiver.

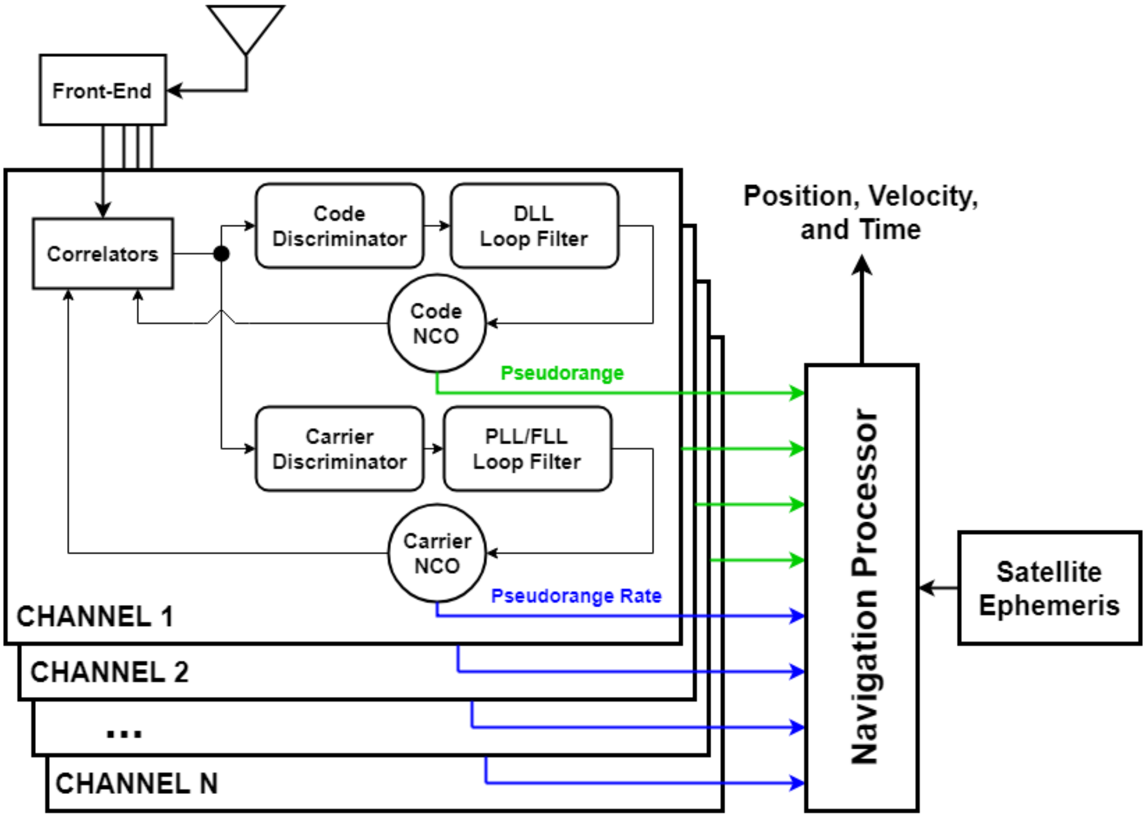

Figure 5.1: Diagram of a GNSS Scalar Tracking Receiver

In cases where a received signal C/N_O ratio is low, that channel's loop filters may not be sufficient to maintain tracking lock. Loop filters operate with feedback from the signal tracking discriminators. When the discriminators carry significant noise due to poor signal quality, the tracking loops can become unstable. They can also fail when the discriminator feedback is biased from events such as multipath. Tracking instability can be prevented or mitigated in harsh signal environments through the use of vector tracking loops. Vector tracking couples the signal tracking and navigation operations of the receiver together, where the PVT dynamics are tracked instead of the signal dynamics on each satellite channel. In other words, the receiver's navigation estimator is used to couple the signal tracking operations of all the satellite channels together. This allows high C/N_O channels to aid the tracking of low C/N_O channels to prevent loss of lock.

Figure 5.2 provides a diagram of a GNSS vector receiver. In the diagram, all the satellite channels' code phase and carrier frequency discriminators are directly inputted into the centralized navigation processor instead of local loop filters. The navigation processor updates the receiver PVT estimation, which then provides frequency updates to the channels' code and carrier replicas. Most vector implementations use an Extended Kalman Filter (EKF) to estimate PVT, although other estimators can be used [73]. Vector tracking requires an initial PVT solution and a decoded ephemeris, which requires scalar tracking initially. Since the navigation solution is necessary for vector tracking, the receiver must be tracking at least four satellite channels.

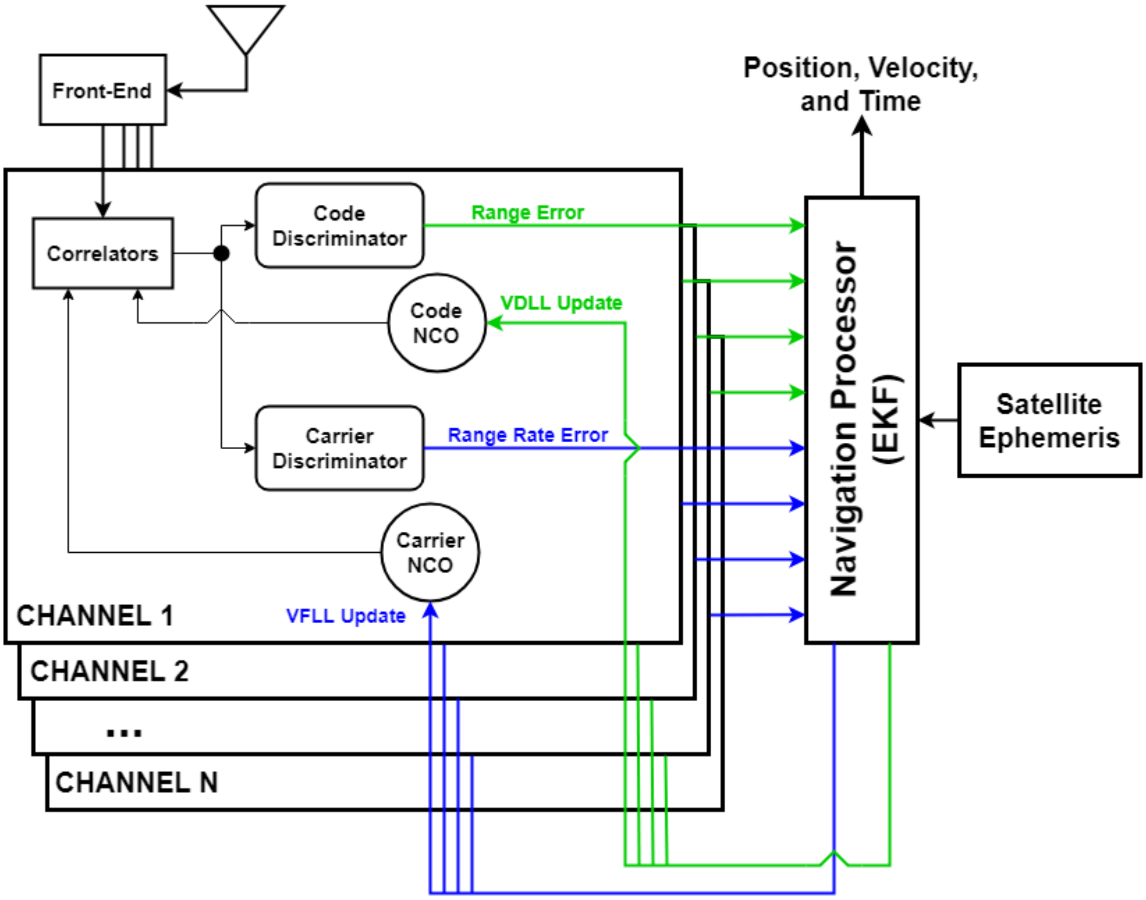

Figure 5.2: Diagram of a GNSS Vector Tracking Receiver

Vector tracking architecture uses two types of loops, the Vector Delay Lock Loop (VDLL) and the Vector Frequency Lock Loop (VFLL). The VDLL couples the satellite channels' code tracking operations together through the receiver position and clock bias estimation. The VFLL couples the satellite channels' carrier tracking operations together through the receiver velocity and clock drift estimation. One or both of these loops can be implemented into a receiver. For example, a receiver may perform vector processing on the channels' code tracking operations through a VDLL and scalar processing on the channels' carrier tracking operations through individual PLL's on each satellite. More discussion is provided on the VDLL and VFLL in the next sections.

5.1 The Vector Delay Lock Loop

GNSS receivers use PRN code tracking to generate pseudorange measurements for the receiver's position and clock bias calculation. The VDLL, originally proposed by James Spilker [21], couples all the satellite channels' code tracking operations together through the receiver position and clock bias estimation. Instead of tracking the PRN code dynamics on each satellite channel individually, the position and clock bias dynamics are tracked and reverted back to each channel's PRN code replica. Figure 5.3 shows a diagram of the VDLL. The code discriminators from each satellite channel are used to correct the estimate of the receiver's position and clock bias. The code discriminators from signal tracking are used in vector processing as pseudorange error measurements instead of code phase feedback errors. The receiver position and clock bias are then predicted one integration period in the future using the current estimation. Pseudorange estimates are formed from the predicted receiver position and clock bias as well as the satellite positions and clock corrections. The pseudorange predictions are then reverted back to PRN code frequency updates for each satellite channel. Next, the vector code loop restarts with the frequency commands performing correlation on the next integration period of signal data.

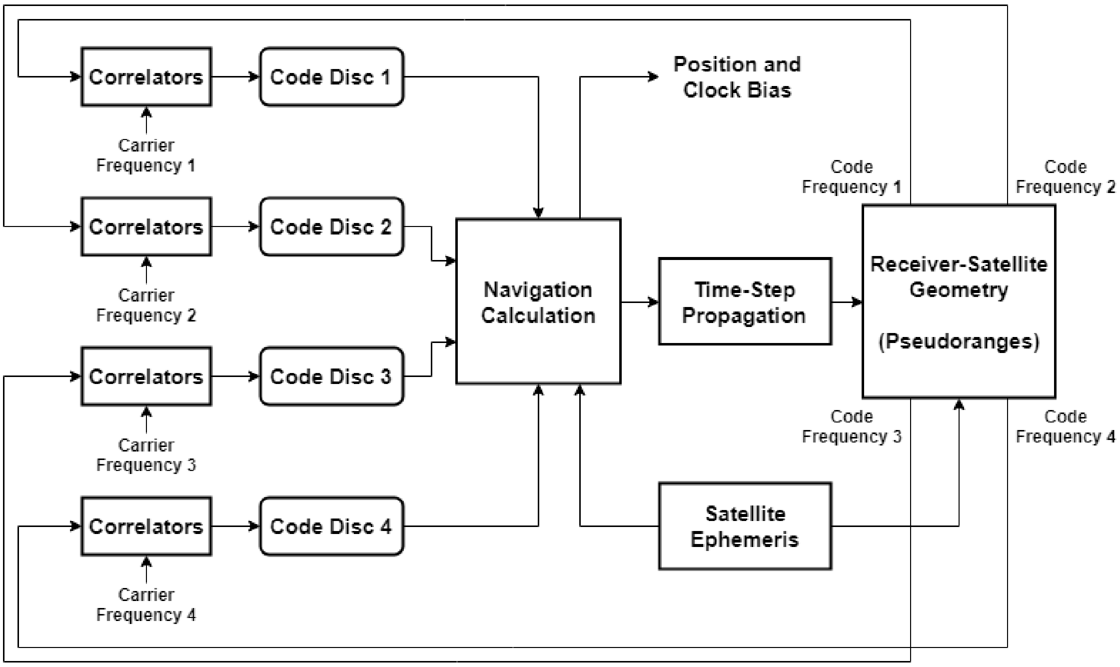

Figure 5.3: Diagram of the Vector Delay Lock Loop

5.1.1 Vector Position and Clock Bias Estimation

Estimating receiver position and clock bias using the VDLL is very similar to the least squares estimation. The least squares model for estimating VDLL position and clock bias using m satellite channels is shown by Equation (5.1) where the measurement vector Y is shown on the left, the measurement observation matrix H is shown in the middle, and the state vector $\underline{x}$ is shown on the right.

$$\begin{bmatrix} \delta\rho_1 \\ \vdots \\ \delta\rho_m \end{bmatrix} = \begin{bmatrix} -a_{x_1} & -a_{y_1} & -a_{z_1} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ -a_{x_m} & -a_{y_m} & -a_{z_m} & 1 \end{bmatrix} \begin{bmatrix} \Delta\hat{x} \\ \Delta\hat{y} \\ \Delta\hat{z} \\ c\Delta\hat{b} \end{bmatrix} \tag{5.1}$$

The measurement vector contains pseudorange errors $\Delta\rho$ for each satellite channel, which describe the difference between measured and receiver estimated pseudoranges. The measurement model contains the ECEF x, y, and z unit vectors from the satellite to the receiver (a_x, a_y, and a_z) for each satellite channel. The unit vector calculations were shown by Equation (3.17).

The state vector contains the estimated error in the receiver's ECEF position ($\Delta\hat{x}$, $\Delta\hat{y}$, $\Delta\hat{z}$) and clock bias $c\Delta\hat{b}$. In other words, the satellite channels' pseudorange errors are used to update the current estimation of receiver position and clock bias, hence why the VDLL requires an initial receiver position and clock bias estimate.

In traditional scalar navigation, the pseudorange residual is the difference between the measured pseudorange $\tilde{\rho}$ and the receiver estimated pseudorange $\hat{\rho}$, which describes the range error difference between the signal measurement and the receiver's current position and time estimation. The pseudorange residual in the VDLL is calculated by scaling the code discriminator ϕ_{DLL} with the PRN code chip width λ_{chip}, shown in Equation (5.2).

$$\delta\rho = \lambda_{chip}\phi_{DLL} \tag{5.2}$$

The discriminator, when using the VDLL, also describes the range error between the true pseudorange from the signal and the receiver estimated pseudorange. The chip width transforms the error residual from a code phase measurement in chips to a range measurement in meters. For the GPS L1 C/A signal, the width is $\lambda_{chip} = 293.05$ meters per chip. For the GLONASS L1 signal, the width is $\lambda_{chip} = 586.1$ meters per chip. Similar to the DLL, the VDLL will become unstable if the channels' PRN code errors become greater than 1 chip. In terms of navigation, if the net receiver position and clock bias estimation error is greater than 1 chip, the VDLL will become unstable.

Most VDLL implementations use the pseudorange error residual from Equation (5.2), however, some researchers whose work is described in [13] and [19], include the traditional scalar pseudorange error along with the code discriminator in the residual as shown by Equation (5.3).

$$\delta\rho = \lambda_{chip}\phi_{DLL} + \tilde{\rho} - \hat{\rho} \tag{5.3}$$

Theoretically, the measured pseudorange $\tilde{\rho}$ and the receiver estimated pseudorange $\hat{\rho}$ should be the same in the VDLL, because the vector algorithm defines the code frequency based upon the estimated pseudorange. This is not the case in GNSS receivers due to sampling frequency quantization that occurs in the digital signal data. The code discriminator must be included in the VDLL residual to provide feedback. Without the discriminator, the VDLL does not have any representation of PRN code tracking error. The scalar pseudorange error does not provide navigation, it simply corrects the code discriminator for quantization effects in the signal data's resolution. The scalar pseudorange error describes the range error between the receiver estimated pseudorange and what that pseudorange was quantized to when the code replica was generated based off of the signal data's sampling frequency.

Figure 5.4 compares positioning results using the VDLL residuals described above with 10 GPS satellites. The signal data was recorded from a front-end in a live sky environment. The blue trajectories are associated to the VDLL measurement residual described by Equation (5.2), and the red trajectories are associated to the VDLL measurement residual described by Equation (5.3). Inclusion of the pseudorange error in the VDLL measurement gives better position accuracy as seen in Figure 5.4. Performance was also more stable under the interstate bridge when including the pseudorange error with the code discriminator.

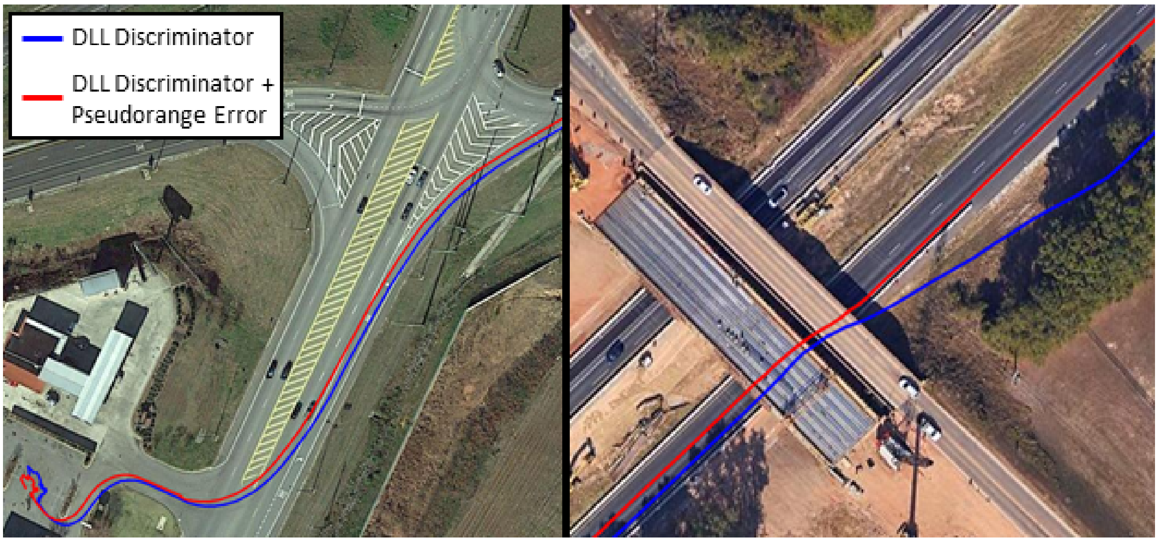

Figure 5.4: Comparisons of VDLL Measurement Residuals using 10 GPS Satellites

Figure 5.5 shows positioning results using the VDLL residuals with 5 GLONASS satellites. Including the pseudorange error with the code discriminator in the VDLL residual appears to help positioning results as it did with GPS. The GLONASS position is also more stable in the presence of multipath from tree foliage when the pseudorange error is included in the VDLL measurement residual as seen in right picture of the Figure 5.5.

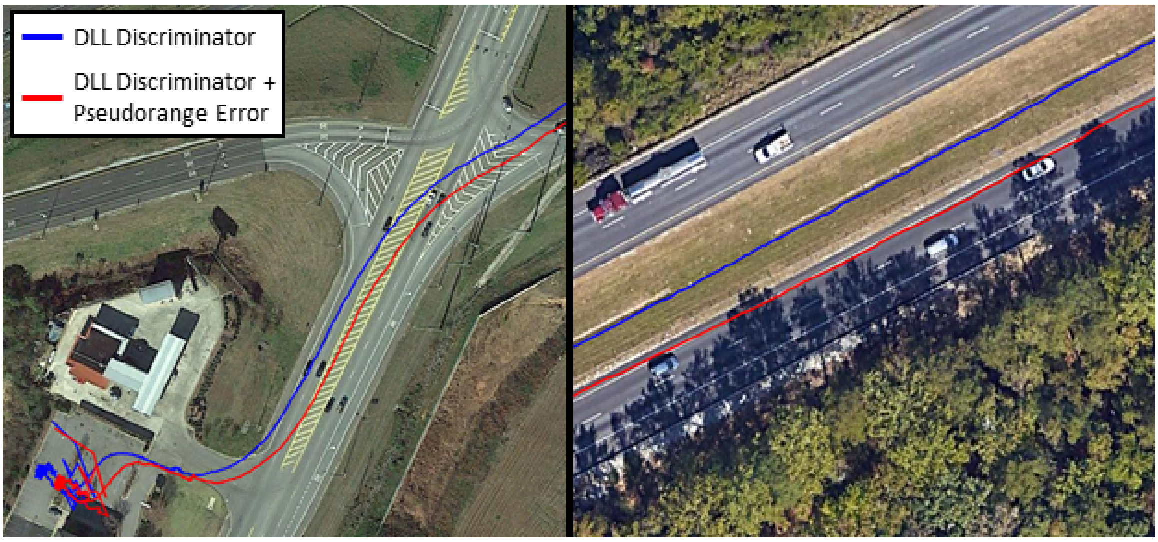

Figure 5.5: Comparisons of VDLL Measurement Residuals using 5 GLONASS Satellites

5.1.2 Vector Code Frequency NCO Update

In the scalar DLL, the code discriminator is applied through a loop filter and integrated to update the PRN code frequency for each satellite channel. The VDLL uses the code discriminators to update the receiver position and clock bias. The receiver position and clock bias estimation is predicted one integration period in time; the prediction is then used to generate estimated pseudorange measurements, which are fed back to the PRN code frequency for each satellite channel [40]. Figure 5.6 illustrates the vector code frequency update. The update uses a form of carrier aiding because the receiver velocity and clock drift estimation is used to predict the receiver position and clock bias at the next integration period. The receiver velocity and clock drift have a proportional relationship to the satellite channels' carrier Doppler frequencies.

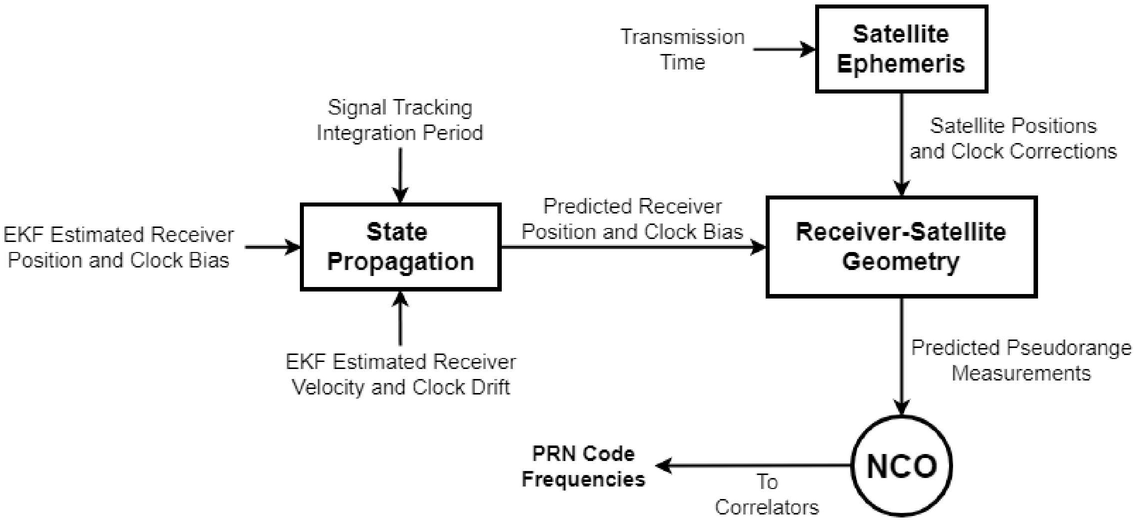

Figure 5.6: VDLL Code Frequency NCO Update Diagram

There are different implementations of the VDLL code frequency NCO update, however, all implementations include the estimated pseudorange measurement. Equation (5.4) shows the receiver estimated pseudorange measurement for a given satellite channel.

$$\hat{\tilde{\rho}} = \sqrt{(x_{sv} - \hat{x_r})^2 + (y_{sv} - \hat{y_r})^2 + (z_{sv} - \hat{z_r})^2} + c\hat{b} - c\tau_{sv} \tag{5.4}$$

The pseudorange is generated from the normalized distance between the satellite and estimated receiver position, along with the estimated receiver clock bias and the satellite clock correction τ_{sv}. A received signal's PRN code frequency contains a raw pseudorange measurement that does not contain the satellite clock correction, and an accurate navigation solution requires corrected satellite clocks. Hence, to relate the measurement back to the code frequency, the satellite clock correction must be removed in the receiver estimated pseudorange. Note that the satellite position and clock correction can be determined at any point in time so long as an accurate ephemeris is associated with that transmission time.

Equation (5.5) shows the pseudorange calculated NCO update for the code frequency [27].

$$f_{code_{k+1}} = f_{chip} - \frac{\hat{\tilde{\rho}}_{k+1} - \hat{\tilde{\rho}}_k}{\lambda_{chip} T} \tag{5.5}$$

This NCO update relies on the difference between the current estimated pseudorange measurement $\hat{\tilde{\rho}}_k$ and the estimated pseudorange measurement for the next integration period $\hat{\tilde{\rho}}_{k+1}$. The PRN chip width λ_{chip} and the signal tracking integration period T are used to convert the pseudorange difference from meters to chips per second. The pseudorange difference provides a Doppler frequency offset and code phase adjustment from the PRN chipping rate transmitted at the satellite f_{chip}. Note that negative sign in the equation is due to the relationship between the pseudorange and code frequency. An increase in the pseudorange requires an increase in receive time, which requires a decrease in the code frequency. When the code frequency is operating at a lower rate, it takes more time to move through an entire integration period. Another way to think of this NCO update is, that the code frequency for the next integration period $f_{code_{k+1}}$ is defined to generate the receiver estimated pseudorange $\hat{\tilde{\rho}}_{k+1}$ as the actual pseudorange measurement observed in the receiver at the next integration period.

The code frequency NCO update can also be determined from estimated receive times [61], as shown in Equation (5.6).

$$f_{code_{k+1}} = \frac{f_{chip} T}{\hat{t}_{r_{k+1}} - \hat{t}_{r_k}} \tag{5.6}$$

The estimated receive time $\hat{t}_r$ comes from the estimated pseudorange measurement $\hat{\tilde{\rho}}$. Equation (5.7) shows the relationship between the pseudorange and receive time, where t_t is the satellite transmission time.

$$\hat{t}_r = \frac{\hat{\tilde{\rho}}}{c} + t_t \tag{5.7}$$

This update determines what the code frequency must be to succesfully generate the necessary amount of PRN code chips in the time period that is the difference between the current and next receive times.

The previous vector code updates were direct calculations of the code frequency instead of integrated NCO updates that are used in traditional scalar loop filters. The third vector code update uses an integrated NCO approach shown by Equation (5.8), which is used in [6] and [72].

$$f_{code_{k+1}} = f_{code_k} - \frac{\hat{\tilde{\rho}}_{k+1} - \hat{\tilde{\rho}}_{code_{k+1}}}{\lambda_{chip} T} \tag{5.8}$$

In this approach, the current code frequency f_{code_k} is updated by using the difference between the receiver estimated pseudorange $\hat{\tilde{\rho}}_{k+1}$ and the code estimated pseudorange $\hat{\tilde{\rho}}_{code_{k+1}}$. Both of these pseudorange estimations are for the next integration period $k+1$. The receiver generated pseudorange uses the estimated receiver position and clock bias for range determination. The code generated pseudorange uses the satellite channel's current code frequency command for range determination as shown by Equation (5.9).

$$\hat{\tilde{\rho}}_{code_{k+1}} = c(\hat{t}_{r_k} + \frac{f_{chip} T}{f_{code_k}} - t_{t_{k+1}}) \tag{5.9}$$

In the equation, the current estimated receive time $\hat{t}_{r_k}$ is updated by the current code frequency and subtracted by the transmission time for the next integration period $t_{t_{k+1}}$ to obtain a transit time pseudorange measurement.

Figure 5.7 show code Doppler responses for GPS and GLONASS satellite channels using the different vector NCO updates discussed. The code Dopplers shown come from VDLL's tracking real signal data recorded in a live sky environment. The GPS channel shows code responses when the channel is in the process of locking onto the PRN frequency. The vector code NCO update responses all behave nearly identical. The GLONASS channel shows code responses when the channel was likely effected by multipath as tree foliage was in the channel's signal path. Prior to the multipath, the GLONASS code responses are nearly identical. When multipath is introduced into the signal, the GLONASS responses have small offsets from each other. The selection of the vector code NCO update can have small effects on positioning performance. In this work, the pseudorange code frequency update from Equation (5.5) is used in the VDLL.

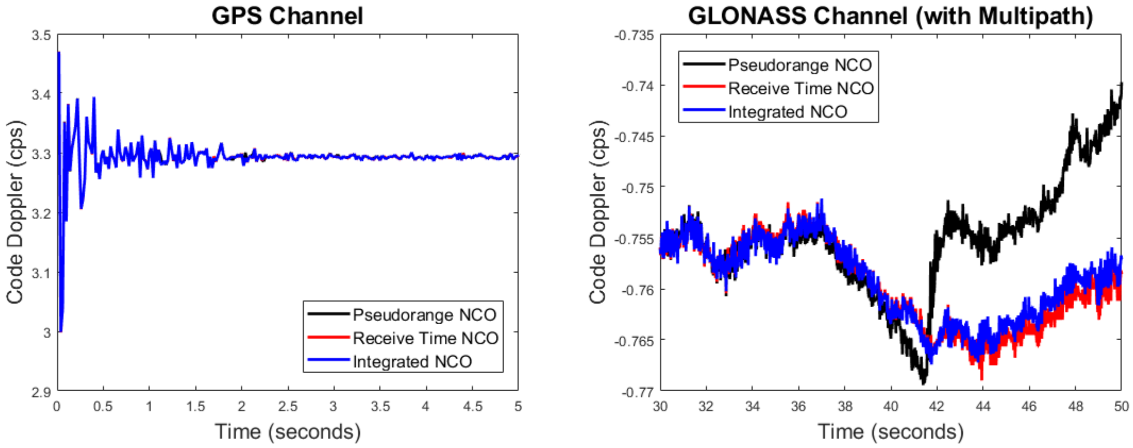

Figure 5.7: GPS and GLONASS Code Frequencies using Different Vector NCO Updates

5.2 The Vector Frequency Lock Loop

GNSS receivers use carrier tracking to generate pseudorange rate measurements for the receiver's velocity and clock drift estimation. The VFLL couples all the satellite channels' carrier tracking operations together through the velocity and clock drift calculation. Instead of tracking the carrier Doppler dynamics on each satellite channel individually, the velocity and clock drift are directly tracked and reverted back to each channel's carrier replica. Figure 5.8 shows a diagram of the VFLL. The carrier frequency discriminators from each satellite channel are used to generate an estimate of the receiver's velocity and clock drift. The carrier frequency discriminators from signal tracking are still used in vector processing, but as pseudorange rate error measurements instead of carrier frequency feedback error. The velocity and clock drift are then predicted one integration period in the future using the current estimation and satellite ephemeris. Pseudorange rate estimates are formed from the prediction, which are then reverted back to carrier frequency updates for each satellite channel. The cycle restarts with the carrier frequency commands generating correlators with the next integration period of signal data. Since the VFLL only relies on the carrier frequency discriminators, the channels' replicas will not be carrier phase locked. The receiver velocity and clock drift solution also requires an accurate receiver position, generally the position is estimated when operating the VFLL.

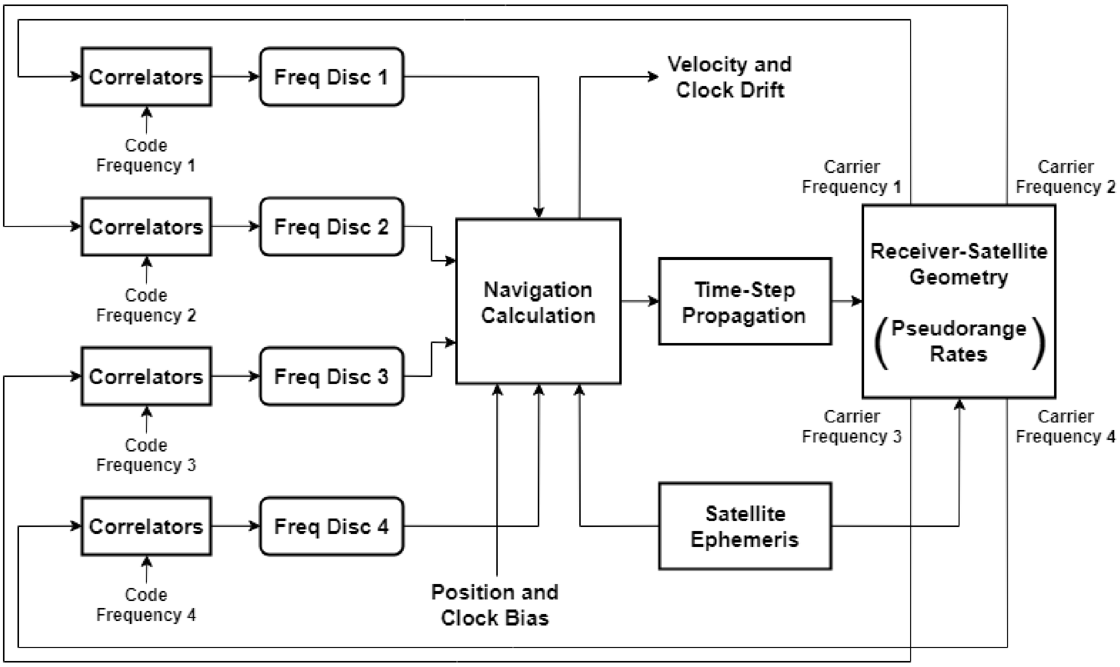

Figure 5.8: Diagram of the Vector Frequency Lock Loop

5.2.1 Vector Velocity and Clock Drift Estimation

Like the least squares solution, the VFLL estimates the receiver's ECEF velocity and clock drift using pseudorange rate error residuals, as shown by Equation (5.10).

$$\begin{bmatrix} \delta\dot{\rho}_1 \\ \vdots \\ \delta\dot{\rho}_m \end{bmatrix} = \begin{bmatrix} -a_{x_1} & -a_{y_1} & -a_{z_1} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ -a_{x_m} & -a_{y_m} & -a_{z_m} & 1 \end{bmatrix} \begin{bmatrix} \Delta\hat{\dot{x}} \\ \Delta\hat{\dot{y}} \\ \Delta\hat{\dot{z}} \\ c\Delta\hat{\dot{b}} \end{bmatrix} \tag{5.10}$$

The VFLL's measurement observation matrix relates the estimated error-in-state variables to the pseudorange rate error measurements using the receiver-satellite geometry, which contains the ECEF unit vectors in the x, y, and z-directions.

Like the VDLL, the main difference between the traditional least squares velocity solution and the VFLL is the measurement error residual used. The traditional least squares solution in scalar tracking receivers use the difference between the measured pseudorange rate $\tilde{\dot{\rho}}$ and the receiver estimated pseudorange rate $\hat{\dot{\rho}}$. The VFLL residual measurement error for a satellite channel is its carrier frequency discriminator ϕ_{FLL} scaled by the transmitted carrier wavelength λ_{wave}, as shown by Equation (5.11).

$$\delta\dot{\rho} = -\lambda_{wave}\phi_{FLL} \tag{5.11}$$

The carrier wavelength converts the frequency discriminator from a frequency error (Hz) to a range-rate error (m/s). Like the pseudorange rate measurement in Equation (4.6), the negative sign in the VFLL discriminator residual is required due to the Doppler effect. The carrier frequency discriminators are filtered through the velocity and clock drift estimator in the VFLL, instead of local loop filters from individual FLL's.

Some VFLL implementations also include the difference between the measured pseudorange rate and the receiver estimated pseudorange rate into the measurement residual as seen in Equation (5.12).

$$\delta\dot{\rho} = -\lambda_{wave}\phi_{FLL} + \tilde{\dot{\rho}} - \hat{\dot{\rho}} \tag{5.12}$$

This is generally not required for optimal performance in the VFLL because the pseudorange rate difference will be zero. The VFLL defines the pseudorange rate measurement based on a predicted carrier Doppler frequency, which comes from the receiver estimated pseudorange rate. Most receivers have enough resolution that quantization error is not an issue. The relationship between the carrier Doppler frequency and pseudorange rate measurement is a direct linear proportionality, which makes defining the pseudorange rate measurement easy.

When a VFLL is aided with additional carrier tracking mechanisms, such as a PLL, the measurement residual must include the pseudorange rate difference from Equation (5.12). When a PLL is used to aid a VFLL, the Doppler frequency is no longer solely specified by the estimated pseudorange rate, hence the difference between the measured and estimated pseudorange rates will not be zero. Figure 5.9 justifies this statement by showing the pseudorange rate differences for GPS and GLONASS satellite channels using the VFLL and the VFLL aided with a PLL (VPLL). The results in the figure were processed with live sky signal data.

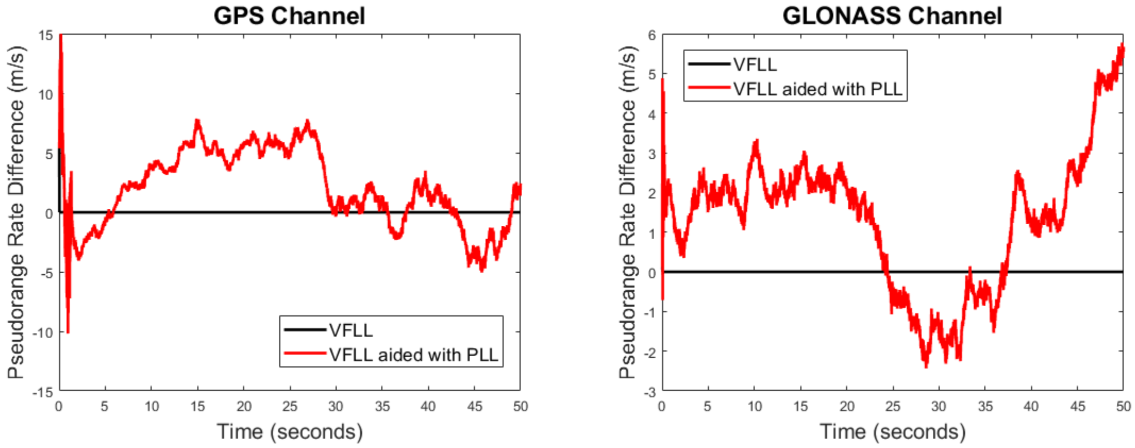

Figure 5.9: VFLL and VPLL Pseudorange Rate Differences for GPS and GLONASS

5.2.2 Vector Carrier Frequency NCO Update

In the scalar PLL or FLL, the carrier discriminator is passed through a loop filter and integrated to update the carrier frequency for each satellite channel. The VFLL uses the carrier frequency discriminators to update the receiver velocity and clock drift solution. The solution is predicted one integration period in time; the prediction is then used to generate estimated pseudorange rate measurements, which are fed back to drive the carrier frequency for each satellite channel. The estimated pseudorange rate measurement for a given satellite channel is shown by Equation (5.13).

$$\hat{\dot{\rho}} = a_x(\dot{x}_{sv} - \hat{\dot{x}}_r) + a_y(\dot{y}_{sv} - \hat{\dot{y}}_r) + a_z(\dot{z}_{sv} - \hat{\dot{z}}_r) + c\hat{\dot{b}}_r \tag{5.13}$$

The pseudorange rate is calculated using the satellite velocity ($\dot{x}_{sv}$, $\dot{y}_{sv}$, and $\dot{z}_{sv}$), the estimated receiver velocity ($\hat{\dot{x}}_r$, $\hat{\dot{y}}_r$, and $\hat{\dot{z}}_r$), the estimated receiver clock drift $c\hat{\dot{b}}_r$, and the estimated unit vectors from the satellite to the receiver (a_x, a_y, and a_z) from Equation (3.17).

The vector carrier frequency update is shown in Equation (5.14) where $f_{carr_{k+1}}$ is the carrier frequency for the next integration period, f_{IF} is the receiver's front-end intermediate frequency, $\hat{\dot{\rho}}_{k+1}$ is the receiver estimated pseudorange rate measurement at the next integration period, and λ_{wave} is the transmitted carrier wavelength.

$$f_{carr_{k+1}} = f_{IF} - \frac{\hat{\tilde{\dot{\rho}}}_{k+1}}{\lambda_{wave}} \tag{5.14}$$

For GLONASS channels, the satellite's FDMA frequency offset must also be included with the intermediate frequency. It is also noted that each GLONASS satellite has its own unqiue carrier wavelength.

5.3 The VDFLL Extended Kalman Filter

When the VDLL and VFLL architectures are used together, the implementation is known as a Vector Delay Frequency Lock Loop (VDFLL) [32]. The VDFLL uses an Extended Kalman Filter (EKF) to update receiver PVT with the signal tracking discriminators as measurement inputs. The EKF uses a system model to filter the receiver measurements to obtain a more accurate PVT estimation. Specifically, the EKF adaptively calculates an estimation gain to determine whether the estimator should put more trust into the system model or the measurements. Two VDFLL implementations are formulated, the position-state VDFLL and the pseudorange-state VDFLL. The Kalman filter estimators discussed here originated from [31].

5.3.1 Position-State VDFLL

The position-state VDFLL tracks the receiver PVT, similar to the VDLL and VFLL discussions in the previous sections. Equations (5.15) through (5.18) provide the position-state VDFLL Kalman filter matrices.

$$\underline{\hat{x}} = \begin{bmatrix} \hat{x}_r & \hat{\dot{x}}_r & \hat{y}_r & \hat{\dot{y}}_r & \hat{z}_r & \hat{\dot{z}}_r & c\hat{b}_r & c\hat{\dot{b}}_r \end{bmatrix}^T \tag{5.15}$$

$$A = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.16}$$

$$H = \begin{bmatrix} -a_{x_1} & 0 & -a_{y_1} & 0 & -a_{z_1} & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -a_{x_m} & 0 & -a_{y_m} & 0 & -a_{z_m} & 0 & 1 & 0 \\ 0 & -a_{x_1} & 0 & -a_{y_1} & 0 & -a_{z_1} & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -a_{x_m} & 0 & -a_{y_m} & 0 & -a_{z_m} & 0 & 1 \end{bmatrix} \tag{5.17}$$

$$Y = \begin{bmatrix} \delta\rho_1 & \ldots & \delta\rho_m & \delta\dot{\rho}_1 & \ldots & \delta\dot{\rho}_m \end{bmatrix}^T \tag{5.18}$$

Equation (5.15) shows the state vector, which includes the estimated receiver position, velocity, and clock errors. The discrete state transition model A is shown by Equation (5.16), where a kinematic velocity model is used. The measurement observation matrix H and measurement array Y for m satellite channels are shown by Equations (5.17) and (5.18), respectively. The measurement covariance matrix R used to weigh the satellite channel measurements was shown in Equation (4.48).

The discrete-time process noise covariance matrix Q describes the uncertainty of unmodeled dynamics not expressed in the state transition matrix used by the filter and is shown by Equation (5.19).

$$Q = \begin{bmatrix} Q_x & 0 & 0 & 0 \\ 0 & Q_y & 0 & 0 \\ 0 & 0 & Q_z & 0 \\ 0 & 0 & 0 & Q_c \end{bmatrix} \tag{5.19}$$

$$Q_x = \begin{bmatrix} \sigma_x^2 \frac{T^3}{3} & \sigma_x^2 \frac{T^2}{2} \\ \sigma_x^2 \frac{T^2}{2} & \sigma_x^2 T \end{bmatrix} \qquad Q_y = \begin{bmatrix} \sigma_y^2 \frac{T^3}{3} & \sigma_y^2 \frac{T^2}{2} \\ \sigma_y^2 \frac{T^2}{2} & \sigma_y^2 T \end{bmatrix}$$

$$Q_z = \begin{bmatrix} \sigma_z^2 \frac{T^3}{3} & \sigma_z^2 \frac{T^2}{2} \\ \sigma_z^2 \frac{T^2}{2} & \sigma_z^2 T \end{bmatrix} \qquad Q_c = \begin{bmatrix} \sigma_b^2 T + \sigma_r^2 \frac{T^3}{3} & \sigma_r^2 \frac{T^2}{2} \\ \sigma_r^2 \frac{T^2}{2} & \sigma_r^2 T \end{bmatrix}$$

The tuning parameters σ_x^2, σ_y^2, and σ_z^2 describe the variances of the noise processes that drive the ECEF x, y, and z velocities with units of m^2/s^3. The tuning parameters σ_b^2 and σ_r^2 describe the variances of the noise processes that drive the clock phase and clock frequency errors with units of m^2/s and m^3/s^2. The signal tracking integration period T is included to scale the noise process variances by the appropriate time step in the Kalman filter. The clock tuning parameters σ_b^2 and σ_r^2 are calculated using Equations (5.20) and (5.21), respectively.

$$\sigma_b^2 = c^2 \frac{h_0}{2} \tag{5.20}$$

$$\sigma_d^2 = c^2 2\pi^2 h_{-2} \tag{5.21}$$

The parameters are scaled by the speed of light c and use the oscillator spectral coefficients h_0 and h_{-2}. The spectral coefficients for different clock qualities can be found in [54].

5.3.2 Pseudorange-State VDFLL

Instead of tracking the receiver PVT dynamics like the position-state VDFLL, the pseudorange-state VDFLL tracks the measurement dynamics. Specifically, this filter tracks each satellite channel pseudorange and pseudorange rate measurement. Theoretically, the position-state and pseudorange-state filters are the same [34]. However, in implementation, the position-state filter performs more robustly [37]. The position-state VDFLL uses an overdetermined system in the measurement observation while the pseudorange-state VDFLL does not, which makes the pseudorange-state filter's vector tracking capabilities limited. In other words, the channel coupling effect of the pseudorange-state filter is weaker relative to the position-state filter.

Equations (5.22) through (5.25) provide the pseudorange-state VDFLL Kalman filter matrices.

$$\underline{\hat{x}} = \begin{bmatrix} \tilde{\rho}_1 & \tilde{\dot{\rho}}_1 & \dots & \tilde{\rho}_m & \tilde{\dot{\rho}}_m \end{bmatrix}^T \tag{5.22}$$

$$A = \begin{bmatrix} 1 & T & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.23}$$

$$C = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.24}$$

$$Y = \begin{bmatrix} \delta\rho_1 & \dots & \delta\rho_m & \delta\dot{\rho}_1 & \dots & \delta\dot{\rho}_m \end{bmatrix}^T \tag{5.25}$$

Equation (5.22) shows the state vector, which consists of the satellite channels' pseudorange and pseudorange rate measurements. Equation (5.23) shows the system model used by the pseudorange-state EKF. The pseudorange rates are treated as the derivatives of the pseudoranges. The estimator's measurement observation matrix and measurement vector are shown by Equations (5.24) and (5.25), respectively. The measurement observation C is an identity matrix, and the measurement array Y is the same as the position-state VDFLL. Note that the measurement array only contains the tracking discriminators, the scalar pseudorange and pseudorange rate differences cannot be included in the pseudorange-state measurement array because they are used in the estimated error-in-state vector. The discriminator measurements are weighed with the same covariance matrix used in the position-state VDFLL.

Since the pseudorange-state VDFLL tracks the satellite measurements instead of the receiver PVT, a recursive least squares calculation is used to update the receiver PVT from the Kalman filtered pseudorange and pseudorange rate residuals. The recursive least squares calculation from Equation (4.47) is used in this work. Note that a Kalman filter cannot be used to update the receiver PVT since an EKF is used to filter the pseudorange and pseudorange-rate measurements in this VDFLL. Cross correlation between the VDFLL and PVT Kalman filters would cause divergence and instability in the receiver estimation. In other words, the statistical information used for estimation was performed in the VDFLL and cannot be used again in a separate PVT Kalman filter.

Figure 5.10 helps describe the pseudorange-state VDFLL algorithm. The signal tracking discriminators are inputted as measurements into the VDFLL Kalman filter. The code discriminators are used to correct the differences between the measured pseudoranges $\tilde{\rho}$ and the receiver estimated pseudoranges $\hat{\rho}$. The carrier frequency discriminators are used to filter the differences between the measured pseudorange rates $\tilde{\dot{\rho}}$ and the receiver estimated pseudorange rates $\hat{\dot{\rho}}$. The filtered pseudorange and pseudorange rate residuals are then inputted as measurements into a recursive least squares estimator to correct the receiver PVT. The updated receiver PVT is used to generate code and carrier frequency commands that are sent to the tracking loops to generate discriminators for the next integration period. Some implementations may use the filtered pseudorange and pseudorange rate measurements directly to update the code and carrier frequency commands, while keeping the navigation processor separate from performing NCO updates. However, using the least squares updated receiver PVT solution in the NCO commands provides additional filtering to the frequency estimations.

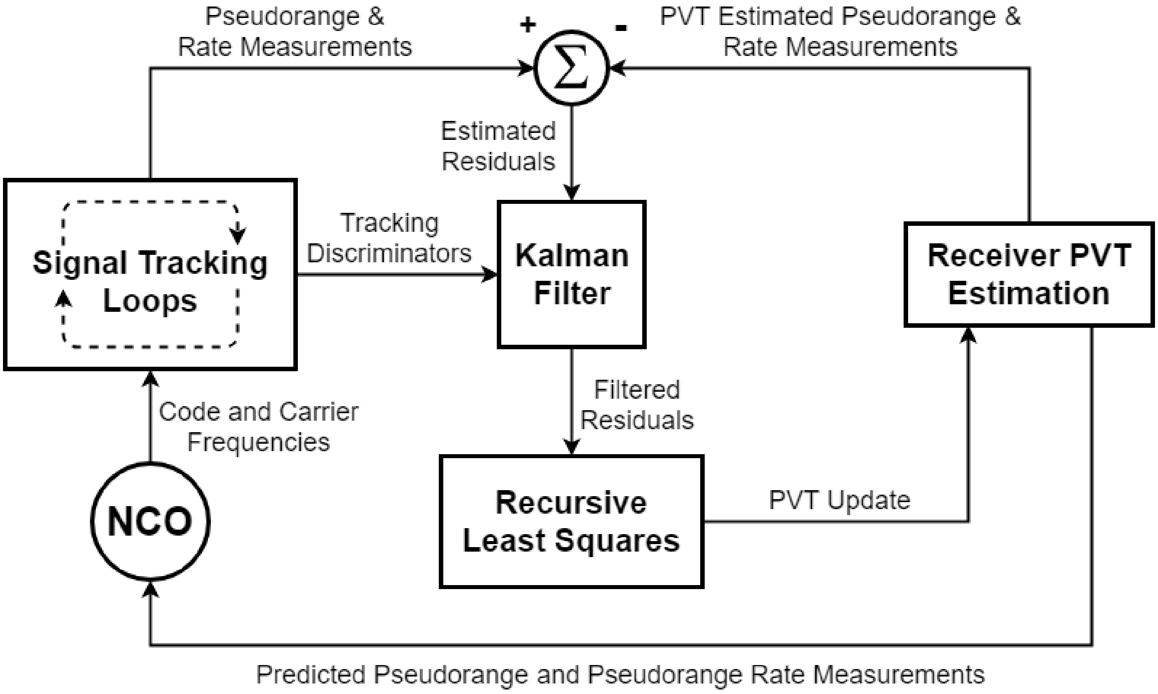

Figure 5.10: Diagram of the Pseudorange-State VDFLL Algorithm

From the measurement observation C in Equation (5.24), it is seen that each satellite channel has a measurement for its pseudorange and pseudorange rate. This measurement matrix is a stack of individual filters for each satellite channel. The channels are uncoupled, which implies the receiver is not performing vector processing. The position-state VDFLL couples the satellite channels through the measurement observation, the pseudorange-state VDFLL couples the satellite channels through the process noise covariance matrix Q, shown by Equation (5.26).

$$\begin{aligned}
Q &= \begin{bmatrix} Q_{1,1}+Q_c & Q_{1,2}+Q_c & \dots & Q_{1,m}+Q_c \\ Q_{2,1}+Q_c & Q_{2,2}+Q_c & & \vdots \\ \vdots & & \ddots & \vdots \\ Q_{m,1}+Q_c & \dots & \dots & Q_{m,m}+Q_c \end{bmatrix} \\
Q_{i,j} &= \begin{bmatrix} \beta_{i,j}\frac{T^3}{3} & \beta_{i,j}\frac{T^2}{2} \\ \beta_{i,j}\frac{T^2}{2} & \beta_{i,j}T \end{bmatrix} \\
\beta_{i,j} &= \sigma_x^2 a_{x_i} a_{x_j} + \sigma_y^2 a_{y_i} a_{y_j} + \sigma_z^2 a_{z_i} a_{z_j}
\end{aligned} \tag{5.26}$$

The clock covariance Q_c from Equation (5.19) is applied to all covariance elements since the receiver clock biases all the pseudorange and pseudorange rate measurements equivalently. Covariance between channel i and channel j is determined by $\beta_{i,j}$, which is a function of the channels' unit vectors (a_x, a_y, and a_z) from the receiver to the satellite. The tuning parameters σ_x^2, σ_y^2, and σ_z^2 are the same as those used in the position-state VDFLL.

Note that if the covariance elements in Q are zeroed and only the diagonal variance elements are used, the pseudorange-state VDFLL operates as a scalar tracking receiver with Kalman loop filters. This is also known as a vector receiver in scalar formulation, which has been used to compare the performance of vector and scalar tracking implementations in [36] and [37]. The same vector code and carrier frequency NCO updates used for the position-state VDFLL are applied to the pseudorange-state VDFLL. However, when applying the code frequency NCO update, from Equation (5.5), the measured pseudorange $\tilde{\rho}_k$ must replace the receiver estimated pseudorange $\hat{\tilde{\rho}}_k$ at the current integration period. This occurs due to the state formulation of the VDFLL implementations. The position-state VDFLL uses the estimated receiver position and clock bias in its state, so its code NCO update requires a pseudorange that is formulated from the estimated position and clock bias. The pseudorange-state VDFLL uses the pseudorange measurement for a given satellite channel in its state, hence that pseudorange measurement must be applied to the code NCO update.

Although the uncoupled pseudorange-state VDFLL performs scalar processing, the implementation is significantly different from a typical scalar receiver that operates with loop filters. This is due to the code and carrier NCO frequencies being directly calculated from the receiver navigation solution. This also induces a severe instability that the uncoupled pseudorange-state filter possesses. In theory, the VDFLL uses strong channels to aid the tracking of weak channels. In the case of the uncoupled pseudorange-state VDFLL, the filter can lead to weak channels degrading the tracking of all satellite channels. This instability is shown in [5].

Figure 5.11 shows positioning results using the pseudorange-state VDFLL in uncoupled (scalar) and coupled (vector) modes with 10 GPS satellites. This is the same GPS signal data used in Figure 5.4 where position-state VDFLL results are shown. The same process noise covariance tuning parameters (σ_x^2, σ_y^2, σ_z^2, σ_b^2, and σ_r^2) were used in all the VDFLL filters. The pseudorange-state results are similar in accuracy to the position-state results, but are much less precise. Although the pseudorange and pseudorange rate measurements are filtered with the Kalman filter, the position is calculated using recursive least squares, which results in noisy estimates. The positioning results between the uncoupled and coupled filters are similar, with the coupled VDFLL performing slightly better under the interstate bridge.

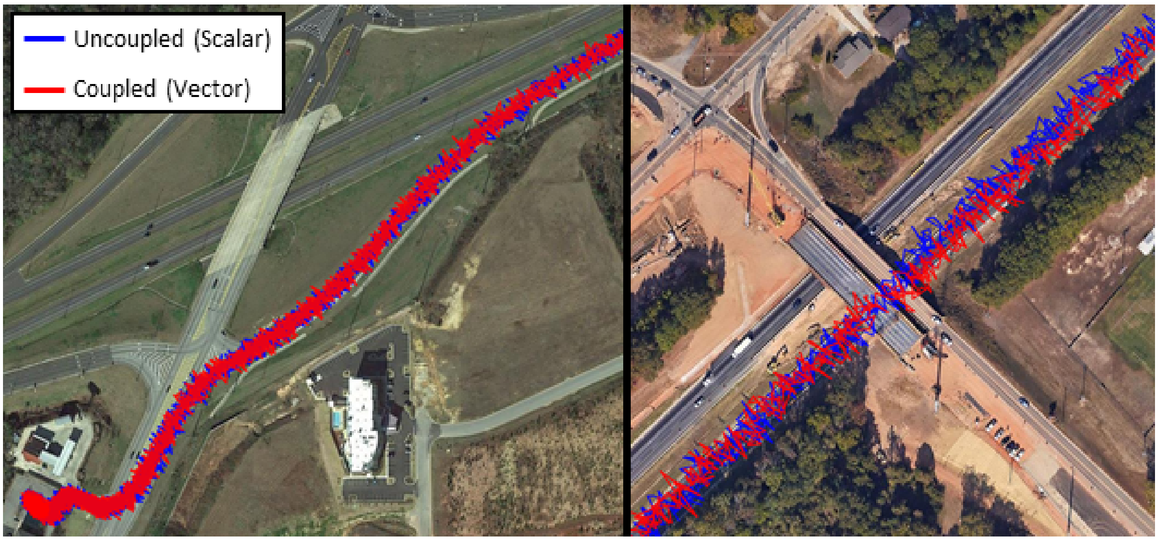

Figure 5.11: Pseudorange-State VDFLL Positioning Results using 10 GPS Satellites

Figure 5.12 shows positioning results using the pseudorange-state VDFLL in uncoupled (scalar) and coupled (vector) modes with 5 GLONASS satellites. This is the same GLONASS signal data used in Figure 5.5 where position-state VDFLL results are shown. The same process noise covariance tuning parameters (σ_x^2, σ_y^2, σ_z^2, σ_b^2, and σ_r^2) were used in all the VDFLL filters. Initially, the GLONASS results are similar to the GPS results shown in Figure 5.11. Later, the uncoupled GLONASS pseudorange-state filter becomes biased in its position estimations. At the interstate bridge, both the uncoupled and coupled filters have divergence issues. The uncoupled pseudorange-state VDFLL diverges in position and becomes unstable when signal tracking lock is lost. The coupled pseudorange-state VDFLL diverges at the bridge, but stabilizes with a significantly biased position estimate. Signal tracking lock continues in the coupled filter after the interstate bridge. The instability issues with the GLONASS pseudorange-state VDFLL may come from the low number of satellite channels providing measurements to the estimator.

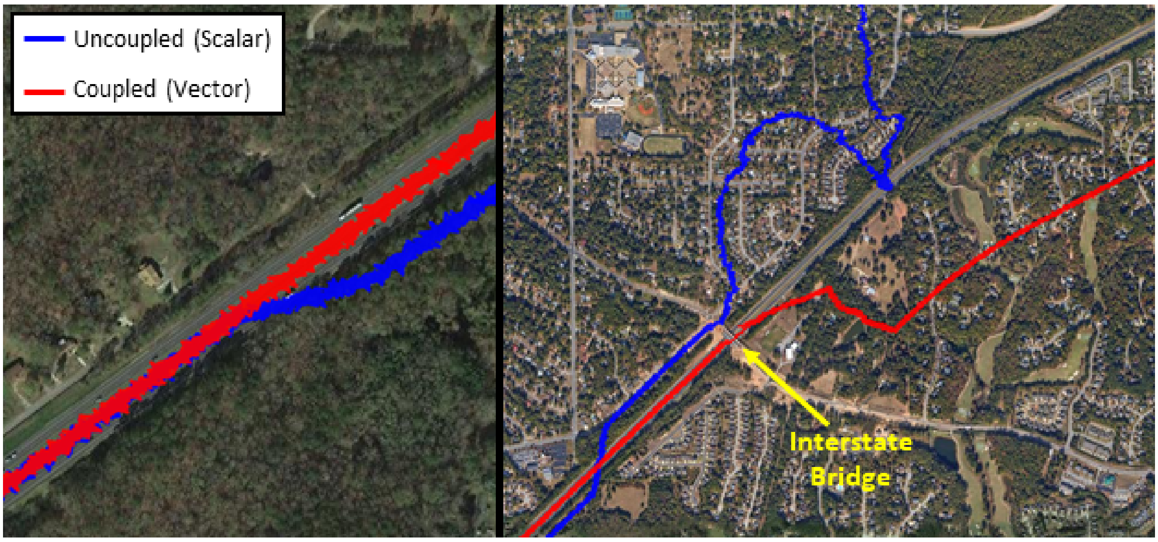

Figure 5.12: Pseudorange-State VDFLL Positioning Results using 5 GLONASS Satellites

Due to the instability issues of the pseudorange-state VDFLL and the overall better performance of the position-state VDFLL, the software-receiver in this work uses the position-state implementation of the VDFLL.

5.4 The Vector Phase Lock Loop

Carrier phase lock is required in GNSS receivers to decode the navigation data messages. Maintaining carrier phase lock is also important to extract precise carrier phase measurements for differential positioning [60]. The VFLL is unable to carrier phase lock since it only uses the carrier frequency discriminators. Noise sharing among satellite channels in vector tracking loops is also a concern for carrier phase lock. The Vector Phase Lock Loop (VPLL) is a cascaded implementation of the VFLL and the PLL in order to obtain carrier phase lock during vector processing. A diagram of the VPLL is shown in Figure 5.13. The VPLL makes use of both the FLL and PLL discriminators to perform carrier tracking. The channels' FLL discriminators are inputted into the centralized VFLL to generate frequency errors, that are applied to local loop filters on each satellite channel. The local PLL discriminators are also included in the channels' loop filters to obtain carrier phase lock. This implementation has been shown to increase carrier phase tracking performance in degraded signal environments [59]. Note that this VPLL is not to be confused with the Real Time Kinematic (RTK) VPLL, where vector processing is applied to the channels' carrier phases for differential positioning. More information on the RTK VPLL, as well as a similar implementation called the Differential VPLL (DVPLL), can be found in [61] and [20].

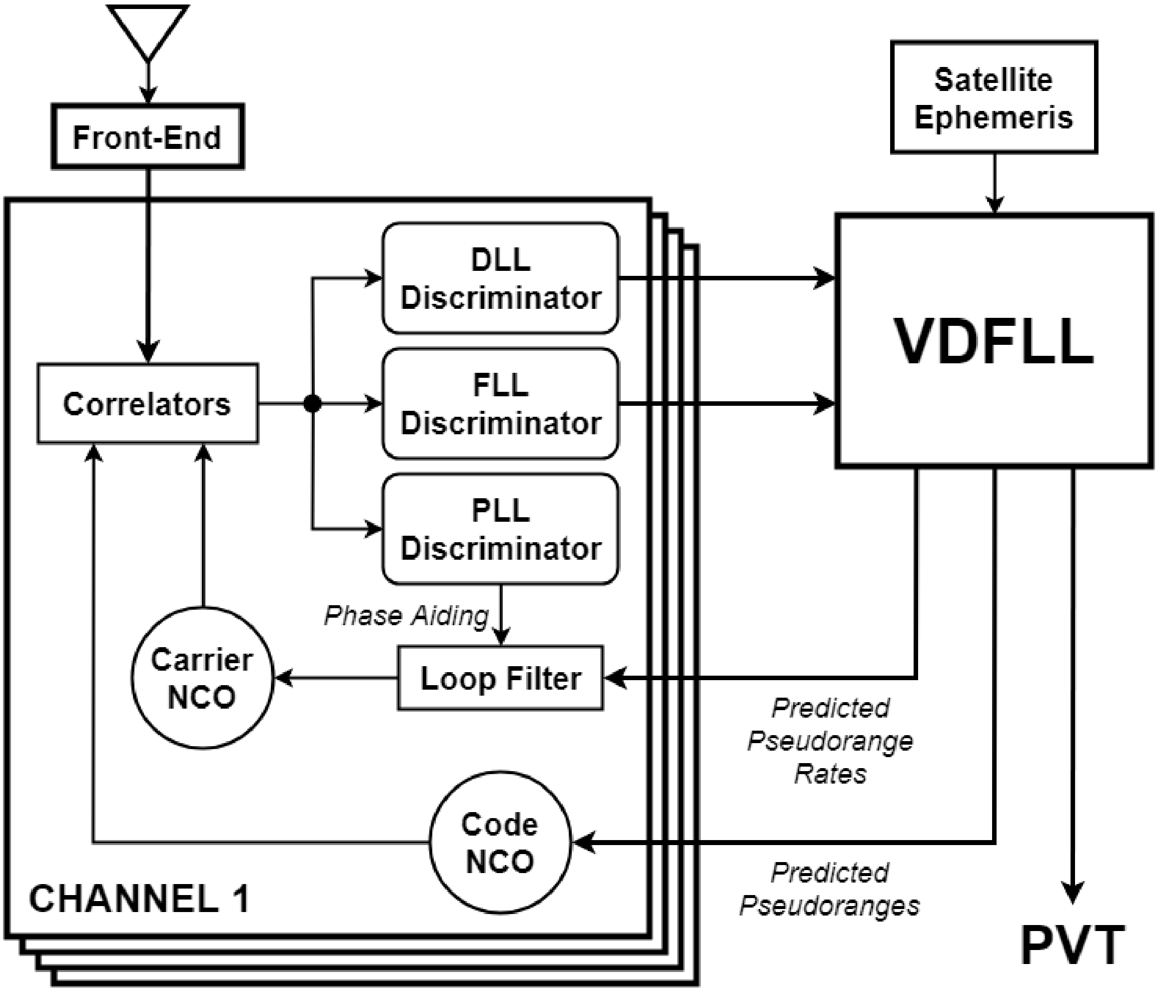

Figure 5.13: Diagram of the Vector Phase Lock Loop

Note that the VPLL uses both the frequency discriminator and the scalar pseudorange rate error in the VFLL measurement residual, from Equation (5.12), since the carrier frequency is updated by both the centralized VFLL and the local PLL. For the VFLL portion of the NCO update, a vector frequency error is calculated to be applied to the local loop filters for each satellite channel. The calculation of the vector frequency error is shown in Equation (5.27), where a difference between the current measured pseudorange rate $\tilde{\dot{\rho}}_k$ and the VFLL predicted pseudorange rate at the next integration period $\hat{\dot{\rho}}_{k+1}$ is taken.

$$\Delta f = \frac{\tilde{\dot{\rho}}_k - \hat{\tilde{\dot{\rho}}}_{k+1}}{\lambda_{wave}} \tag{5.27}$$

The carrier wavelength λ_{wave} is used to convert the error from meters per second to Hertz. A satellite channel's vector frequency error is simply its carrier frequency discriminator with filtering from the navigation estimator.

5.4.1 PLL aided FLL Loop Filter

The VPLL's loop filter design will determine the algorithm's carrier tracking performance. Specifically, the performance depends on how much of the carrier frequency update is dependent on the vector frequency error and the scalar phase discriminator. The most common implementation and the one used in this work is the third-order PLL aided with a second-order FLL (3PLL-2FLL). A continuous-time block diagram of the 3PLL-2FLL is shown in Figure 5.14. This implementation was originally proposed by Phillip Ward to increase carrier phase scalar tracking performance [50]. The implementation was then repurposed for the VPLL by using a vector frequency error input [63]. Additional information on the implementation of this loop filter can be found in [53].

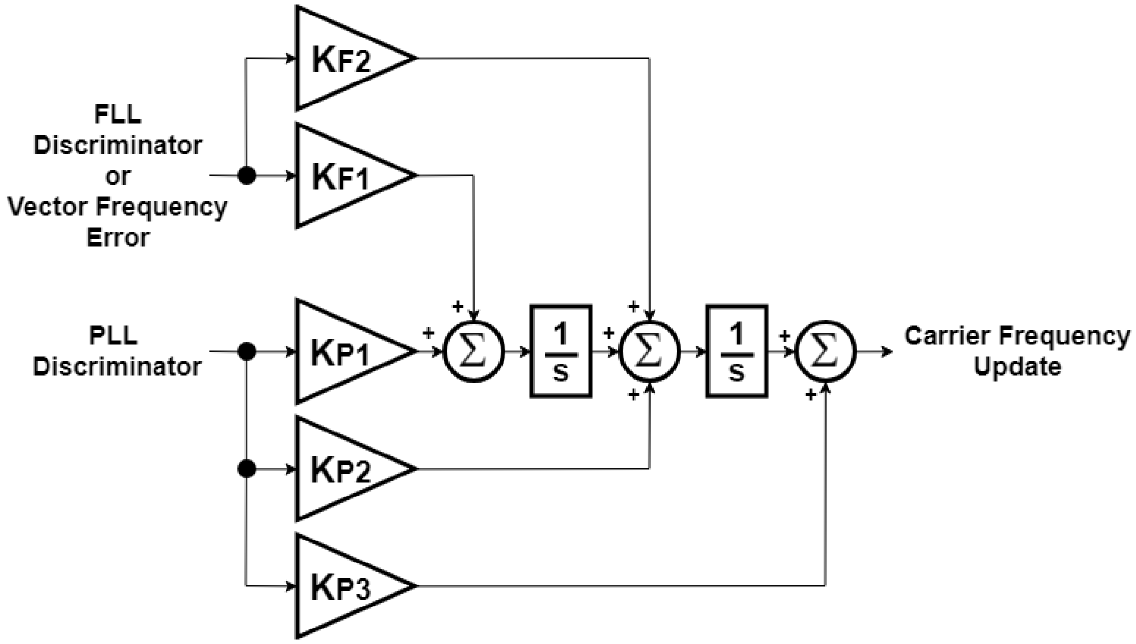

Figure 5.14: Third-Order PLL aided Second-Order FLL Loop Filter

Figure 5.15 shows the frequency lock-on dynamics for a GPS satellite channel using the VFLL, the PLL aided FLL loop filter with the vector frequency error (3PLL-2VFLL or VPLL), and the PLL aided FLL loop filter with the scalar frequency discriminator (3PLL-2FLL). The VFLL immediately locks onto the channel's Doppler frequency because of an accurate receiver velocity and clock drift estimation. However, the VFLL Doppler frequency estimation is not precise due to noise sharing in the vector loop. The VFLL response also is not carrier phase locked. The loop filter implementations are much slower to frequency lock due to low noise bandwidth tuning. However, the frequency responses are precise due to noise rejection by the loop filter. The responses are also carrier phase locked from PLL aiding. The VPLL and 3PLL-2FLL dynamics are very similar because they both use the same filter design. There is a small phase lead in the VPLL response relative to the 3PLL-2FLL response. The VPLL locks onto the Doppler frequency slightly faster because the vector frequency error contains less noise than the carrier frequency discriminator.

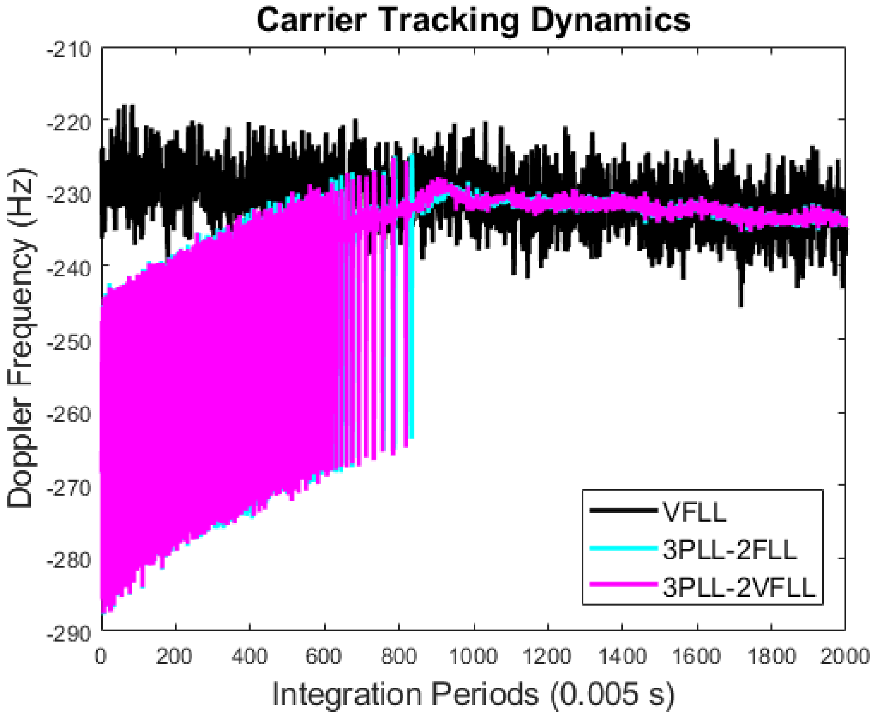

Figure 5.15: Frequency Lock-On Dynamics of VFLL, VPLL, and Scalar 3PLL-2FLL

The software receiver in this thesis uses the VPLL over the VFLL and PLL for tracking the GPS and GLONASS carriers. Including the VPLL into the receiver will enhance velocity estimation performance and also allow the receiver to maintain carrier phase lock.

5.5 Conclusion

This chapter discussed vector tracking loops, with emphasis on the VDFLL. The VDFLL performs code and carrier tracking across all satellite channels through a centralized EKF instead of individual DLL's and PLL's. The chapter also touched on the VPLL, which cascades a centralized VFLL estimator with local PLL loop filters for enhanced carrier tracking. Vector tracking results were shown for GPS and GLONASS data sets, when the constellations operate independently of each other. These results indicate that vector tracking can be performed on both constellations. The next chapter focuses on implementation and analysis of applying both constellations to a vector tracking processor. Combining GPS and GLONASS into a single vector tracking filter should enhance navigation accuracy and estimator stability.

Chapter 6

GPS & GLONASS Receiver Implementation and Results

Single constellation (GPS or GLONASS) vector tracking was discussed in Chapter 5. This chapter discusses the implementation details required to combine GPS and GLONASS in a single vector tracking filter. The chapter also analyzes the performance of vectorizing the constellations together through simulation and experimental results.

6.1 GPS & GLONASS Vector Tracking Implementation

In this work, the software receiver performs navigation through a vector tracking loop that uses both GPS and GLONASS. Figure 6.1 provides a diagram of the GPS & GLONASS vector tracking receiver. The hardware front-end downconverts GPS L1 C/A and GLONASS L1 signals into digital IF data processed by the receiver. A centralized EKF couples the navigation and tracking operations together for both constellations' satellite channels. The vector tracking EKF maintains estimations of receiver position, velocity, and time. The code tracking operations are performed with a VDLL, and the carrier tracking operations are performed with a VPLL. Specifically, the VPLL design uses a third-order PLL aided with a second-order FLL loop filter on each satellite channel. The PLL uses carrier phase discriminator inputs, generated from its own channel. The aiding FLL uses vector frequency error inputs, generated from filtering the channels' carrier frequency discriminators in the centralized navigation processor.

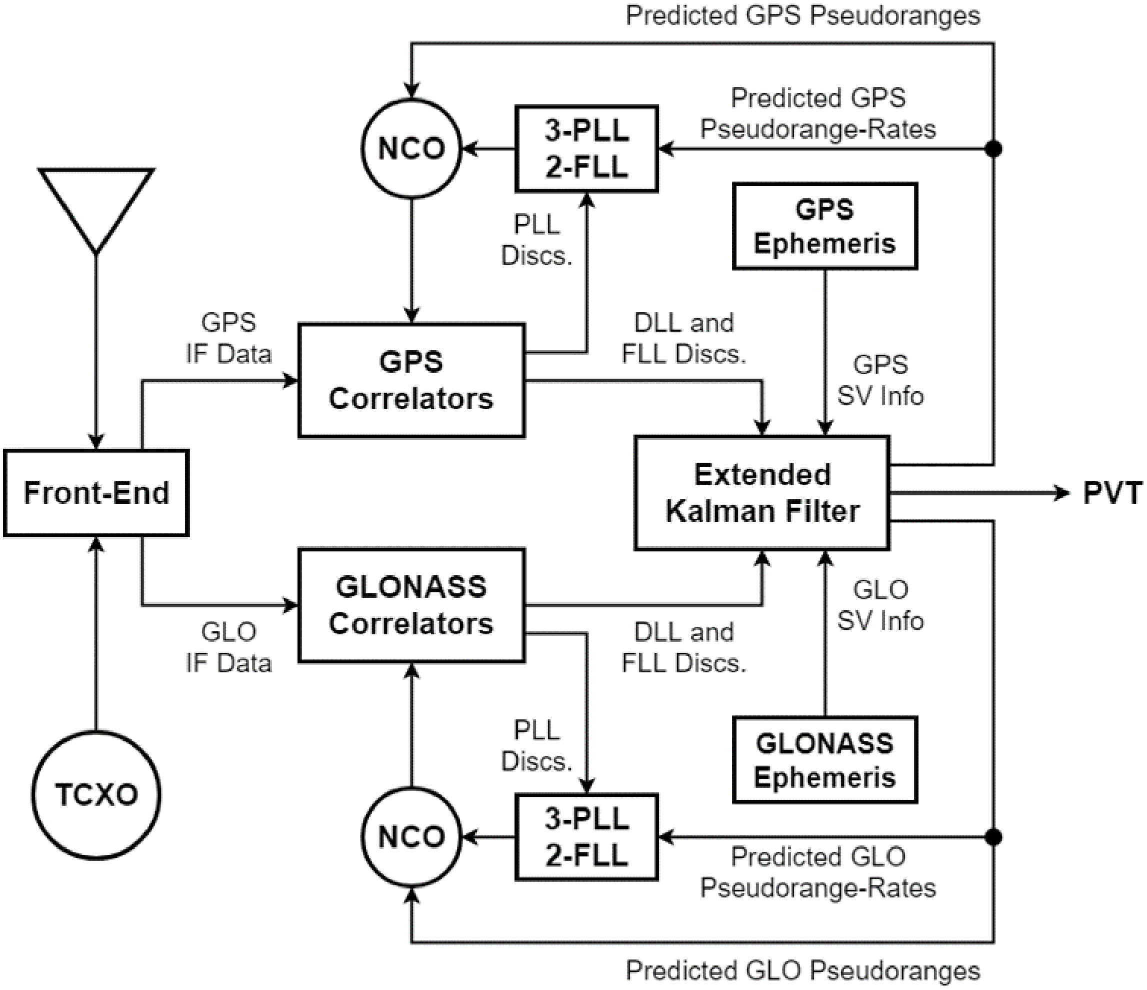

Figure 6.1: Diagram of the GPS & GLONASS Vector Tracking Receiver

6.1.1 Timing and Synchronization Between GPS and GLONASS

To successfully use both GPS and GLONASS signal measurements in the software receiver's navigator, the constellations' broadcasts need to be recorded with accurate timing. In this work, the IFEN SX3 hardware front-end is used to record GPS and GLONASS signal data. A picture of the front-end is shown in Figure 6.2. Since GPS and GLONASS operate at different carrier frequencies in the L-band, the IFEN uses separate front-ends to record the systems' satellite signals. However, both front-ends run on the same clock, a TCXO that operates with a sampling frequency of $f_s = 20$ MHz. The GPS signal data is downmixed to an intermediate frequency of 5000445.88565834 Hz, and the GLONASS signal data is downmixed to an intermediate frequency of 4999635.33878326 Hz.

Figure 6.2: Picture of the IFEN SX3 Hardware Front-End

Since both front-ends are operated by the same clock, the GPS and GLONASS signal data samples are synchronized in receive time. This also means that a single receiver clock drift can be estimated for both constellations. Because GPS and GLONASS generate pseudorange measurements in different time systems, where the time offset between the systems may change, two receiver clock biases (one for each constellation) should be estimated. However, it is assumed that the time difference between the systems is small and constant for the software receiver developed in this work. This assumption is validated in Appendix C for the IFEN SX3 front-end's internal TCXO clock. However, this assumption was not true for higher-quality atomic oscillators. More information on the clock bias difference between GPS and GLONASS for different types of oscillators can be found in Appendix C.

If the front-ends operated with different oscillators, individual clock errors would have be estimated for the GPS channels and the GLONASS channels. Using different clocks for each front-end would make combined vector tracking challenging to perform, due to the issue of synchronizing the GPS and GLONASS signal data samples.

Vector tracking operates with integrate and dump cycles on all the satellite channels. When vector tracking is performed only on GPS or GLONASS, the integrate and dump cycles occur on the channels' synchronous data bits. This means that the integrate and dump cycles are initialized on the channels' preamble locations (GPS) or time mark locations (GLONASS). When both constellations are used in the vector algorithm, the data messages are not synchronous across the satellite systems. In other words, the GPS preambles will not align with the GLONASS time marks in transmission time.

The receiver designed in this work initializes the GPS and GLONASS integrate and dump cycles based on signal data samples. Meaning, the initial integrate and dump cycles for GPS and GLONASS are aligned as closely as possible in data samples, while ensuring the constellations' channels are all associated to the same synchronous data bit within their respective systems. Another way of stating this: the GPS channels are all associated to the same TOW for their integration periods, and the GLONASS channels are all associated to the same TOD for their integration periods; while the GPS and GLONASS periods are aligned to similar signal data samples. An integration period of $T = 0.02$ seconds is used for both constellations, since both GPS and GLONASS navigation messages are transmitted at 50 bps. Once GLONASS time marks are located, the 100 cps meander sequence is deterministic and can be peeled off in signal tracking.

The vector tracking filter initially requires an accurate receiver PVT estimate, hence, a total of 4 satellite channels are required. The satellite channels' receive times must also be synchronized to their respective constellation's timescale. In general, GPS and GLONASS will both have at least 4 satellites, making synchronization simple to perform for each constellation. In the case where one constellation has at least 4 satellites and the other does not, the the initial navigation solution from the sufficient constellation can be used to synchronize the channels for the insufficient constellation.

The challenge comes when neither constellation has 4 satellite channels, but together they have at least 4 channels. In this scenario, transformation between GPS time and GLONASS time is required. Figure 6.3 helps illustrate this concept. In the figure, 2 GPS satellites are transmit synchronous preambles and 2 GLONASS satellites are transmit synchronous time marks. The preambles and time marks do not align in absolute time, and there is a difference of δt between them. The preambles and time marks enter the receiver at similar signal data samples (this will not always be the case), so they may be used to calculate an initial PVT estimate as well as synchronize the channels' receive times.

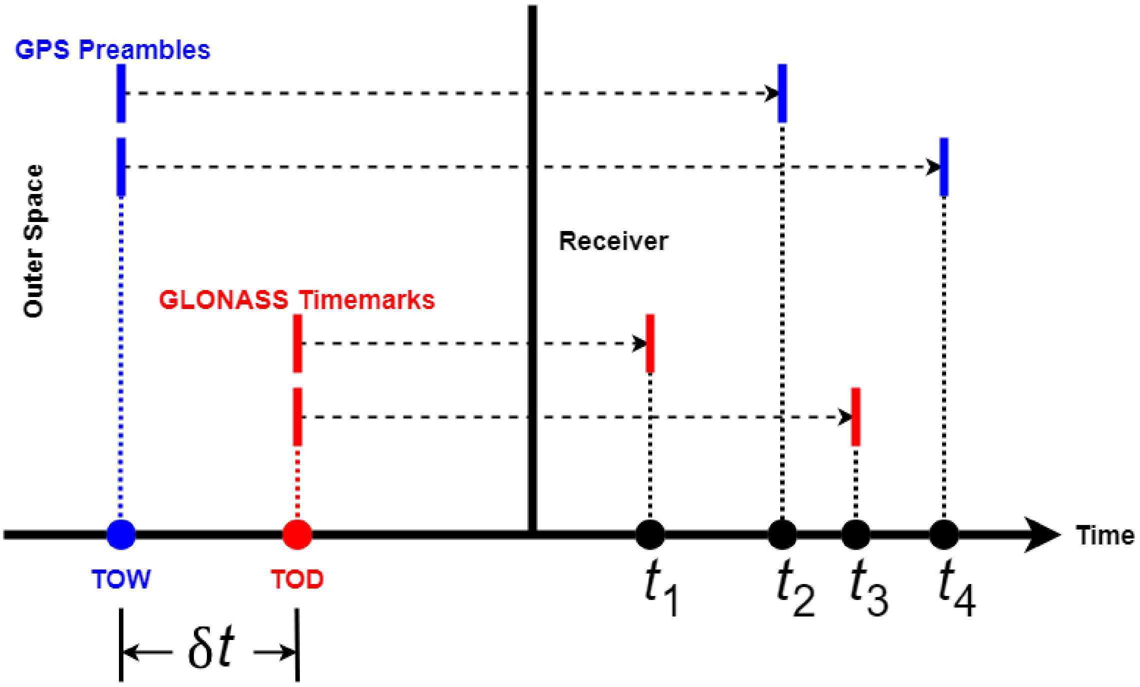

Figure 6.3: Time Synchronization between GPS and GLONASS

The earliest channel comes from a GLONASS satellite that enters the receiver at time t_1, which is assumed to have a transit time of $\Delta t = 65$ to 85 milliseconds. Another GLONASS channel enters the receiver at time t_3. The difference in signal data samples Δs and the front-end sampling frequency f_s are used to calculate the receive time for that GLONASS channel. The same is true for the GPS channels at times t_2 and t_4 when δt is included in the calculation and TOW is converted to GLONASS time $(TOW)_{GLO}$. Equation (6.1) summarizes the channels' receive times in the GLONASS timescale.

$$\begin{aligned}
t_1 &= TOD + \Delta t \\
t_2 &= (TOW)_{GLO} + \Delta t + \frac{\Delta s_{12}}{f_s} + \delta t \\
t_3 &= t_1 + \frac{\Delta s_{13}}{f_s} \\
t_4 &= (TOW)_{GLO} + \Delta t + \frac{\Delta s_{14}}{f_s} + \delta t \\
\delta t &= TOD - (TOW)_{GLO}
\end{aligned} \tag{6.1}$$

In order to know δt, the TOW in GPS time must be converted to the equivalent TOD in GLONASS time $(TOW)_{GLO}$ or vice versa. The conversion of GPS time since the beginning of the week to GLONASS time since the beginning of the day is shown in Equation (6.2).

$$(TOW)_{GLO} = \text{mod}(TOW, 86400) + 3\text{h} - ls - \tau \tag{6.2}$$

In the equation, the mod function is used to transform seconds since the beginning of the week to seconds since the beginning of the day in GPS time. Note that there are 86,400 seconds in a day. GLONASS time is 3 hours ahead of GPS time. GPS maintains time in Greenwich, London, United Kingdom, while GLONASS maintains time in Moscow, Russia; which results in a 3 hour difference. Unlike GPS, GLONASS time keeps track of leap seconds ls with Coordinated Universal Time (UTC) . Currently, as of August 2019, GPS is ahead of UTC by 18 leap seconds [67]. There is also a time discrepancy τ between GLONASS and UTC. According to the GLONASS ICD, $|\tau| < 1$ millisecond, however, τ is generally lower than 1 microsecond [8].

Due to the time discrepancy, error will be induced into the receiver navigation solution when constellation time conversions between GPS and GLONASS are performed. The time discrepancy could be estimated as an additional clock bias state in the navigation estimator to mitigate the issue. If the channels were all converted to GLONASS time, a time discrepancy bias could be estimated for the GPS channels and vice versa. In general, it is simpler and more accurate to keep the channels' receive times on their own constellation timescales in the software receiver. Time conversions are only necessary in the rare case when neither constellation has at least 4 channels, but meets the condition when combined.

When GPS and GLONASS measurements are applied to the navigation processor, they are applied using a common transmission time (CTT) batch approach. This method allows the software receiver to operate with strong stability at the cost of navigation with an ambiguous receive time. Additional information on GPS and GLONASS measurement synchronization in the vector tracking filter can be found in Appendix B.

6.1.2 Vector Tracking Measurement Synchronization

Since vector tracking couples the signal correlation and navigation processes together, an issue of measurement synchronization arises in the algorithm. This issue becomes more complicated when both GPS and GLONASS are included into the centralized vector tracking filter. Figure 6.4 illustrates the issue with four satellite channels. In the figure, the set of colored bars across the channels show the data bits from the GNSS constellation's synchronous data message with respect to receiver time. The data bits enter the receiver at different times because each satellite has its own unique distance from the receiver. These data bits are also the channels' signal tracking integration periods (20 milliseconds). In other words, each set of data bits is associated to a signal tracking integrate and dump cycle for the channels. The circles in the figure represent the start/end of the integration periods. The colors for each set of circles are associated to a common satellite transmission time. In vector processing, the tracking discriminators are used as the measurements, which are generated from correlation across the integration periods. The discriminator residual measurements are best represented at the middle of their integration periods, which are shown as diamonds in the figure. Since the estimated Doppler frequency is constant throughout an integration period, the pseudorange-rate measurements are best represented at the middle of the integration period. From the figure, it is shown that the measurements do not align in time.

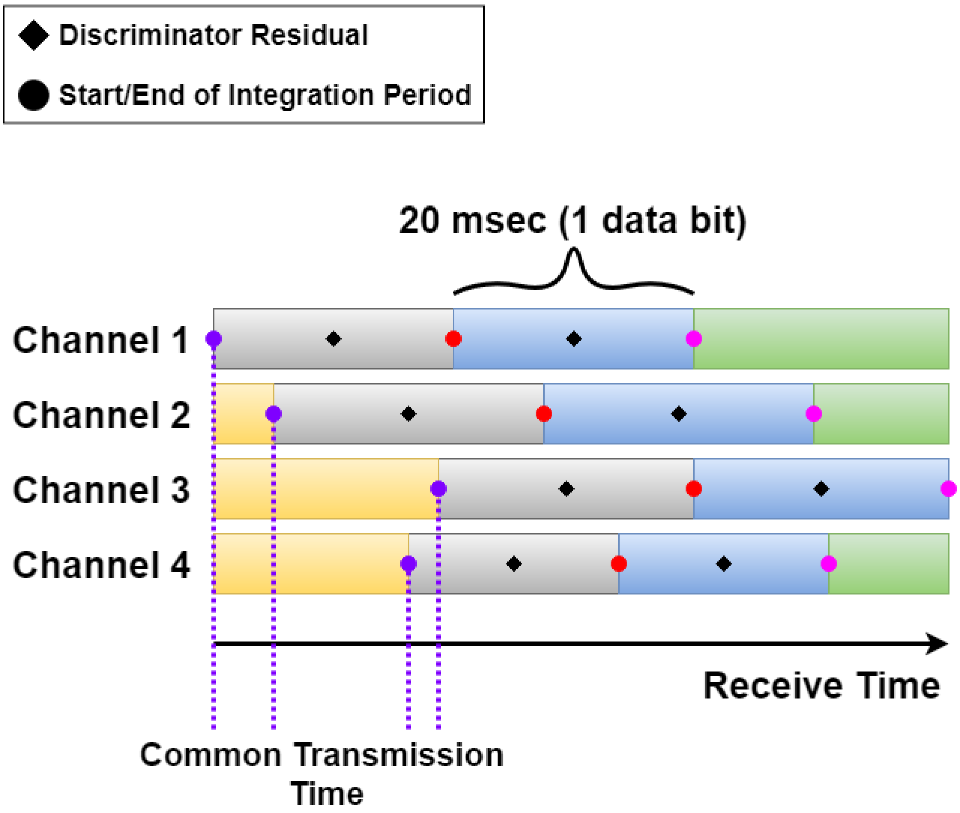

Figure 6.4: Vector Discriminator Measurements with Respect to Channel Receive Time

In general, GNSS measurements do not align in time, but this is not an issue in typical scalar tracking since each channel focuses on its own discriminators. In vector tracking, the discriminators are required to update the navigation solution, which performs signal tracking operations on all channels. The question then becomes, how is the solution updated with asynchronous discriminators to ensure each satellite channel has an accurate navigation state that can effectively be used to track its signal? A few approaches are discussed to resolve this question.

Common Transmission Time Batch Estimation

The simplest, most straight forward way to perform vector tracking is to update the navigation solution with CTT discriminators in a batch estimator. In this approach, the channels' discriminators are assumed to occur at the end of the integration periods, where all the channels' measurements are associated to a common satellite transmission time. The state prediction is performed using a time step equivalent to the signal tracking integration period T. Figure 6.5 illustrates the approach using four satellite channels.

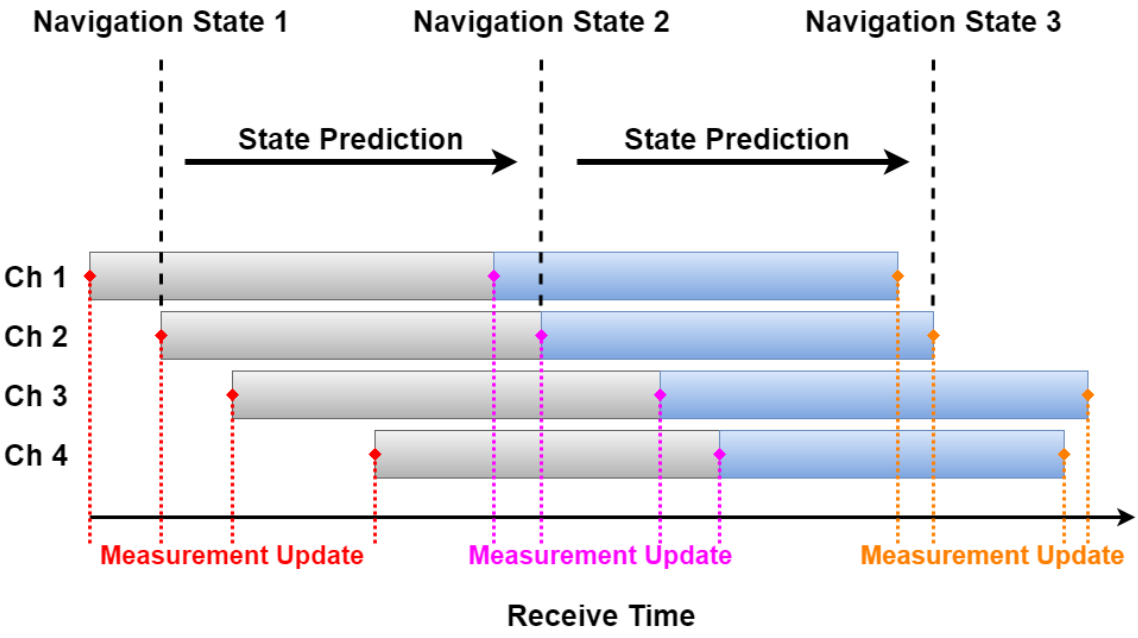

Figure 6.5: Vector Tracking with Common Transmission Time Approach

In this approach, the measurement update is always performed with all the channels' discriminators. The batch estimator provides strong coupling between the channels since all the channels update the navigation state together at one given time. This also means that all satellite channels use the same navigation solution for tracking during an integrate and dump cycle. In general, this is a stable form of vector tracking that can provide an accurate navigation solution. The vector tracking results in Chapter 5 used this approach.

There are some issues with this implementation. This approach uses CTT positioning, which means the navigation solution is not associated to any one true receive time, but a range of receive times. CTT positioning assumes the true receiver position and clock bias is constant across a small range of time. Correct timing of the navigation solution is very important in many applications that use GNSS. Since the discriminators are assumed to occur at the end instead of the middle of the channels' integration periods, this creates inaccuracy in the estimation. Depending how the channels' integration periods are lined-up in time, the CTT batch approach forces some channels to track using an outdated navigation solution, which degrades tracking performance. These issues addressed here become more significant as the receiver dynamics increase.

The CTT batch approach becomes more complicated when GPS and GLONASS are implemented together in a single vector tracking filter. The GPS and GLONASS transmission times will not match up together because the constellations are on different time-scales. The software receiver in [68] combines GPS and GLONASS together with the CTT batch approach. This led to positioning results that were noisier when the constellations were combined compared to using GPS alone. However, the publication in [68] used high process noise tuning in the Kalman filter for stability validation, thus resulting in noisier position estimations. More accurate, stable positioning may be accomplished by implementing GPS and GLONASS vector measurements on receive time, removing any transmission time ambiguity between the constellations. The next approaches discuss sycnchronization using common reception time methods.

Common Reception Time Batch Estimation

The CRT batch approach uses the same concept as the CTT batch approach, where the navigation solution is updated by all the channels' measurements at once. However, the discriminator measurements are generated on a common reception time instead of a common transmission time. The discriminator measurement generation is performed through integrate and dump cycles that are the same for all channels, as seen in Figure 6.6. The integrate and dump cycles for all channels are performed on the same set of signal data, which ensures the discriminator measurements lineup in time. This removes the time ambiguity and outdated navigation solution issues seen in the previous approach. However, in this approach, the correlation process becomes much more complicated because the integrate and dump cycles occur over data bit transitions. If the navigation data message is known for all satellite channels, then the integrate and dump cycles can be performed without the possible loss of correlation.

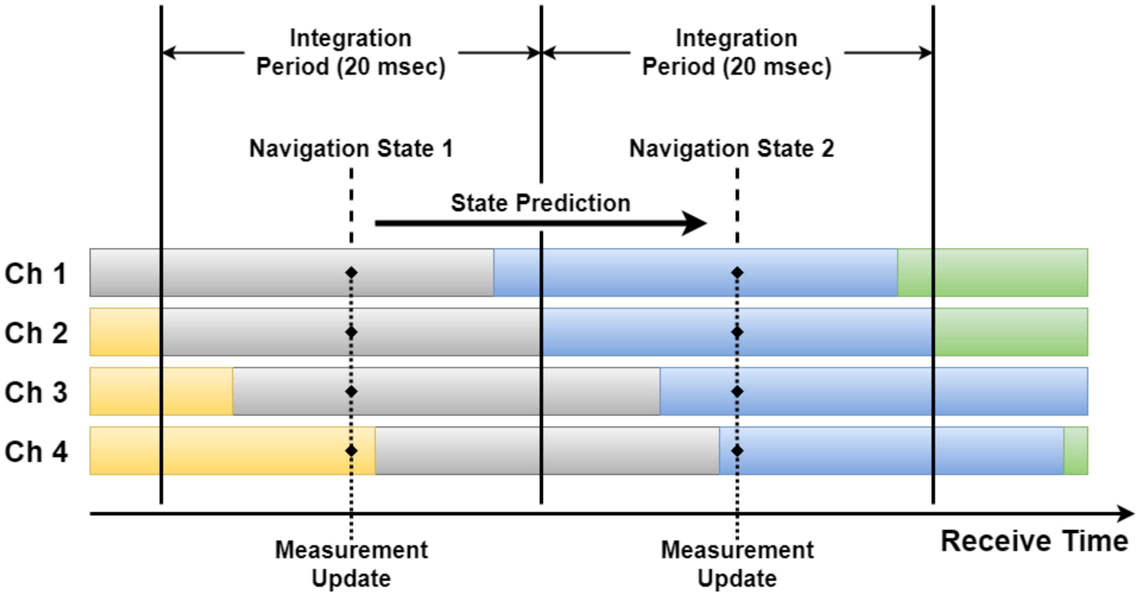

Figure 6.6: Vector Tracking with Common Reception Time Approach

The receiver in this work does not have initial knowledge of the navigation messages, which makes the CRT batch approach challenging to use. To overcome this, the CRT batch discriminators can be generated through averaging 1-millisecond integration period discriminators on each integrate and dump cycle. Averaging 1-millisecond discriminators ensures data bit transitions will not degrade the vector tracking measurements from correlation loss. Unfortunately, the noise observed on averaging twenty 1-millisecond discriminators is much greater than the noise on one 20-millisecond discriminator. The noise is especially degrading to the carrier frequency discriminator.

Figure 6.7 compares ECEF velocity estimations using CRT and CTT batch vector tracking methods. The CRT estimations are significantly noisier than those calculated from CTT. The noise comes from the degraded frequency discriminators when using the CRT method. Since the estimator model uses velocity to update position, the CRT positions are also noisier than the CTT positions. The increase in accuracy from having synchronized measurements in CRT is not enough to overcome the error from additional noise on the discriminator measurements. Without previous knowledge of the navigation data message, CTT outperforms CRT.

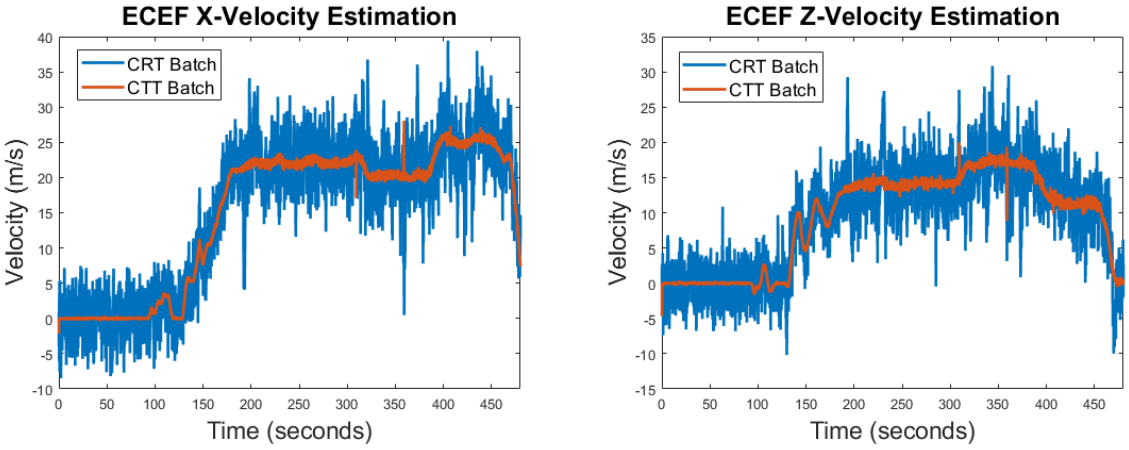

Figure 6.7: ECEF Velocity Estimations using CRT and CTT Batch Vector Tracking

Asynchronous Estimation

Instead of performing batch estimation as the CRT and CTT methods do, vector tracking can be performed recursively using asynchronous measurements. Figure 6.8 illustrates the process of asynchronous estimation. At locations where discriminator measurements occur, the estimated navigation state is propagated to that time point, and a measurement correction is performed. In each integrate and dump cycle, instead of performing the Kalman filter algorithm once (batch), the Kalman filter is performed with m iterations for m satellite channels (recursive). Although more complex than the CTT batch method, this approach guarantees the navigation solution is associated to a single receive time and that each channel is being tracked with the receiver's most current navigation solution.

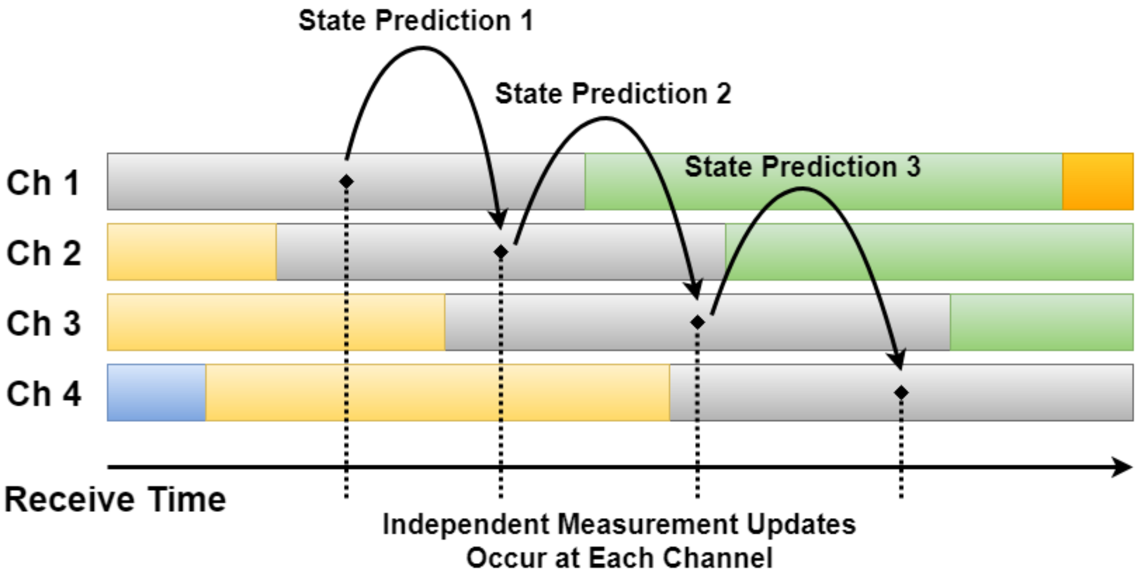

Figure 6.8: Vector Tracking with Asynchronous Measurement Approach

Since the asynchronous approach does not perform batch estimation, there is a small decoupling effect between the channels. The satellite channels are still coupled through the navigation solution, but the coupling effect is weaker because the channels' discriminators do not update the receiver PVT all at one instant. The decoupling effect can make the asynchronous method less stable compared to the CTT batch method.

Discriminator Corrections When a satellite channel's tracking discriminators perform a measurement update on the navigation state, the navigation state must be at the state predicted by the satellite channel. If this is not the case, the discriminator measurements do not fully describe the error in the system. This is not an issue in batch approaches since all the channels are associated to the same predicted state. However, this does become an issue with asynchronous estimation because the navigation state changes with each channel's measurement correction. In other words, the navigation state at a channel's measurement update may not align with the predicted state. To account for this issue, corrections to a channel's discriminator measurements are required when using asynchronous estimation.

Two approaches to the discriminator correction are given here. The first type of discriminator correction is known as a channel correction, where previous discriminators from past satellite channels are used to correct the discriminators for the current channel performing the measurement update. This method was taken from [39], where the issue of discriminator corrections was first brought up in a publication. Equation (6.3) describes the channel correction approach.

$$\begin{aligned} \nu(k+N) &= \nu(k+N)^* - H(k+N)\sum_{m=k}^{k+N-1} A(m,k+N)K(m)\nu(m) \\ \nu(k+N) &= \begin{bmatrix} \lambda_{chip}\phi_{DLL_{k+N}} \\ -\lambda_{wave}\phi_{FLL_{k+N}} \end{bmatrix} \end{aligned} \tag{6.3}$$

In the equation, $\nu(k+N)$ are the corrected tracking discriminators for the current channel, $\nu(k+N)^*$ are the uncorrected tracking discriminators for the current channel, $H(k+N)$ is the measurement observation matrix for the current channel, $A(m,k+N)$ is the estimator's state transition matrix that uses a time step from channel m to the current channel, $K(m)$ is the Kalman gain from channel m, and $\nu(m)$ are the corrected tracking discriminators from channel m. The variable k is associated to the earliest satellite channel for the previous integrate and dump cyle, and the variable N is the number of satellite channels since k.

In implementation, a series of 8 x 1 arrays are maintained as corrections for each satellite channel. When a certain channel's measurement update comes up, its discriminators are corrected by subtracting the product of the channel's measurement observation matrix H and its correction array from the discriminators. After the vector measurements are corrected, the channel's array elements are re-initialized to 0's. This channel's array will then reaccumulate using the other channels' corrected discriminators until its time to perform its measurement update again. Additional information on the discriminator channel correction approach can be found in [6].

A simpler way to correct the discriminators is by adding the scalar residuals to them, as shown by Equation (6.4).

$$\nu(k+N) = \begin{bmatrix} \lambda_{chip}{\phi_{DLL_{k+N}}}^* + \tilde{\rho}_{k+N} - \hat{\rho}_{k+N} \\ -\lambda_{wave}{\phi_{FLL_{k+N}}}^* + \tilde{\dot{\rho}}_{k+N} - \hat{\dot{\rho}}_{k+N} \end{bmatrix} \tag{6.4}$$

The code discriminator is corrected with the scalar pseudorange residual ($\tilde{\rho} - \hat{\rho}$), and the carrier frequency discriminator is corrected with the scalar pseudorange rate residual ($\tilde{\dot{\rho}} - \hat{\dot{\rho}}$). The pseudorange and pseudorange rate measurements for a given satellite channel are generated from the predicted state, for a vector tracking receiver. The estimated pseudorange and pseudorange rate are generated using the current navigation state. Hence, the difference between the measured and estimated pseudoranges/rates describes the corrections to the tracking discriminators.

Figure 6.9 shows positioning results with GPS using the different vector tracking measurement synchronization approaches. The same process noise tuning was used for each of these estimators. As predicted, the CRT batch approach is noisy and somewhat unstable. Initially, the CTT batch and asynchronous scalar approaches are similar in position. The asynchronous channel approach is more biased in position than the other filters. At the interstate bridge, the asynchronous scalar approach loses some stability, while the asynchronous channel approach keeps a consistent, accurate position. The channel correction approach has better stability than the scalar correction approach because it maintains a stronger coupling between the satellite channels.

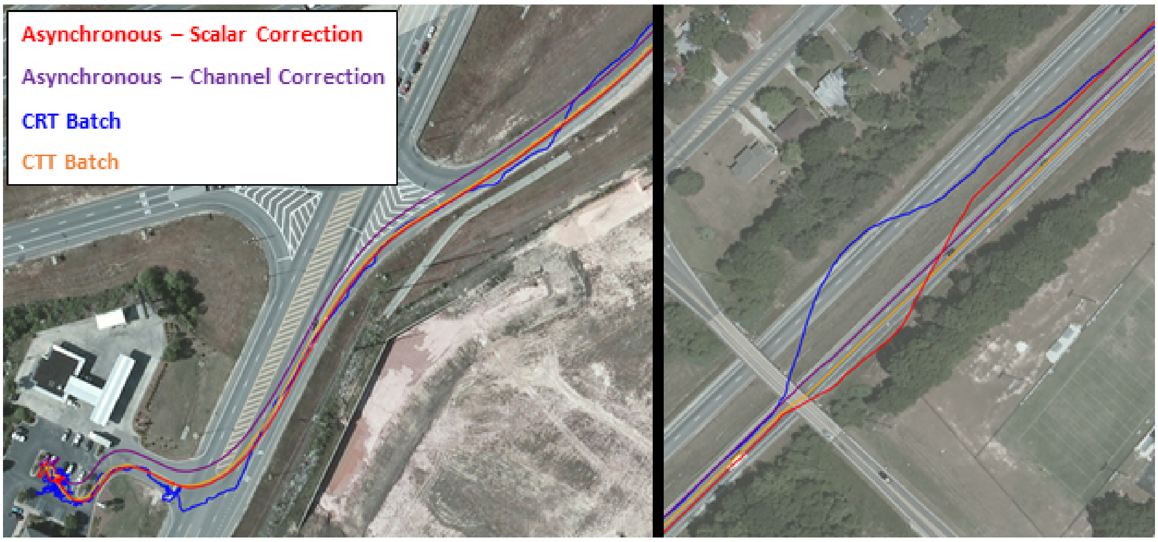

Figure 6.9: Comparisons of Measurement Synchronization Approaches with GPS

In this thesis, the vector tracking software receiver uses the CTT batch approach of measurement synchronization. Although this form of measurement synchronization has an ambiguous receive time, the batch estimation approach provides the strongest form stability in the software receiver's navigation estimator. Additional results comparing the measurement synchronization methods can be found in Appendix B.

6.1.3 PZ-90 to WGS84 ECEF Coordinate Transformation

GPS and GLONASS both refer their satellite positions in ECEF coordinate frames, where the origin of the frame is located at the center of the Earth. However, the coordinate frames used by GPS and GLONASS are different and require a coordinate transformation to move between the frames. GPS uses the World Geodetic System 1984 (WGS84) coordinate frame, which models the Earth based on information in 1984. The WGS84 frame was revised in 2004. GLONASS uses the Parametry Zemli 1990 (PZ-90) coordinate system, which models the Earth based on information in 1990. The PZ-90 system was revised in 2002 (PZ-90.02) and 2011 (PZ-90.11).

Originally, GPS nor GLONASS provided any information on the coordinate transformation between their systems. In 1996, Lincoln Laboratories concluded that the transformation from PZ-90 to WGS84 involved a small rotation and translation [52]. The transformation from PZ-90 (u, v, and w) to WGS84 (x, y, and z) is shown in Equation (6.5).

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 2.5\text{ m} \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & -1.9\text{ x }10^{-6} & 0 \\ 1.9\text{ x }10^{-6} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{6.5}$$

In 2002, GLONASS updated its ECEF coordinate frame to PZ-90.02. The new coordinate frame was fully implemented into satellite system by 2005. Officially, the transformation from PZ-90.02 (u_{02}, v_{02}, and w_{02}) to WGS84 is a small translation [41]. This transformation is shown in Equation (6.6).

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} u_{02} \\ v_{02} \\ w_{02} \end{bmatrix} + \begin{bmatrix} 0.36 \pm 0.10 \text{ m} \\ -0.08 \pm 0.10 \text{ m} \\ -0.18 \pm 0.10 \text{ m} \end{bmatrix} \tag{6.6}$$

In 2011, GLONASS updated its ECEF coordinate frame to PZ-90.11, which was fully implemented into the system by 2014. The Russian Federation provided a transformation from PZ-90.11 to the International Terrestrial Reference Frame 2008 (ITRF2008) coordinate frame [70], which is a translation on the scale of millimeters. The transformation from ITRF2008 to WGS84 consists of a translation on the scale of centimeters. For GNSS pseudorange positioning, errors range from 1-10 meters; hence it can be assumed that the WGS84 and PZ-90.11 coordinate frames are coincident.

Although, officially, WGS84 and PZ-90.11 are nearly the same (within centimeters), experimental data shows otherwise. GPS and GLONASS signal data was collected from static antennas and processed with their respective coordinate frames. Figure 6.10 shows static positioning results in Auburn, Alabama when using 8 GPS satellites in the WGS84 frame and 5 GLONASS satellites in the PZ-90.11 frame. The results are compared to the true antenna position that is known. Clearly, there is a significant offset between the GPS and GLONASS position estimates. Figure 6.11 shows similar results in Des Moines, Iowa. The Des Moines results come from positioning with 9 GPS satellites in WGS84 and 7 GLONASS satellites in PZ-90.11.

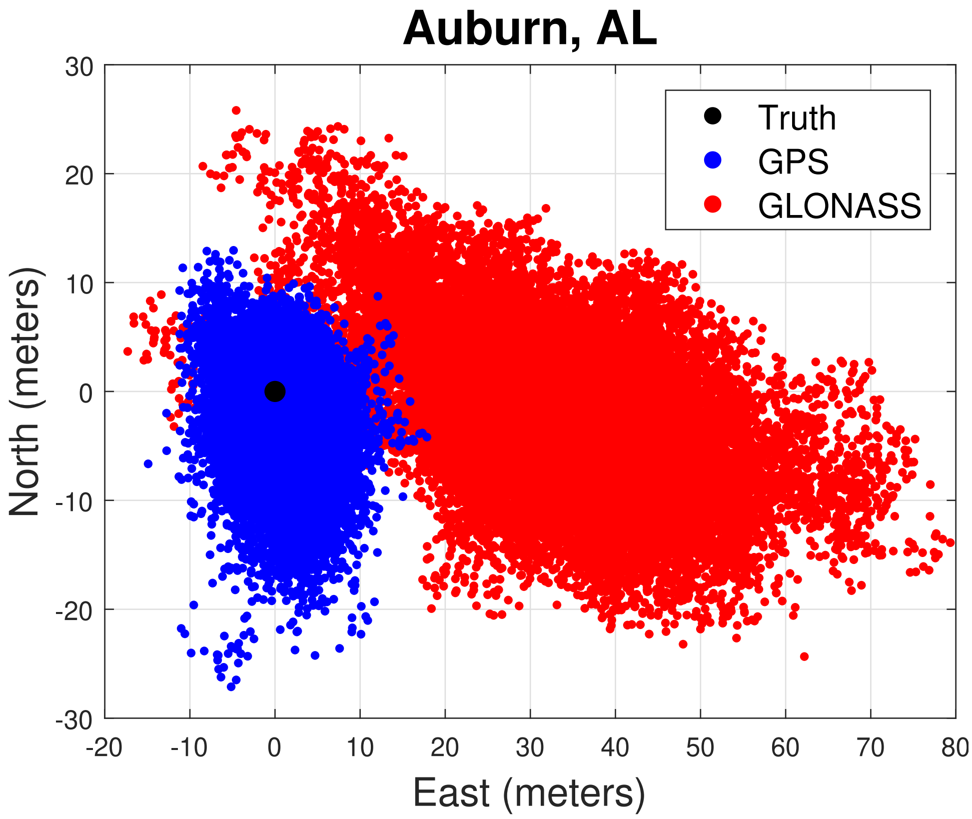

Figure 6.10: Static Positioning Comparisons of GPS and GLONASS in Auburn, AL

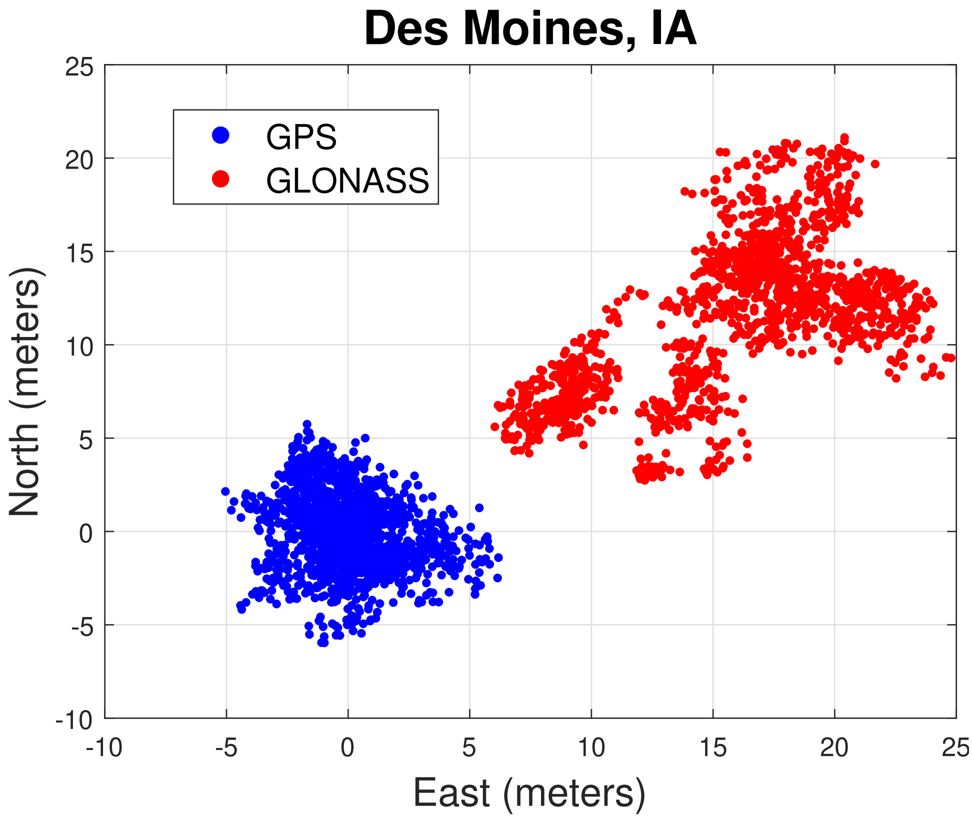

Figure 6.11: Static Positioning Comparisons of GPS and GLONASS in Des Moines, IA

The most likely explanation for this occurrence is that WGS84 and PZ-90.11 are within centimeters when positioning in Russia or high latitudes, but this is not the case in the mid-latitude United States. A series of static GPS and GLONASS positioning data sets were taken in the Auburn, Alabama and Des Moines, Iowa areas to establish a transformation between WGS84 and PZ-90.11. The ECEF x, y, and z-differences between the data sets are shown in Table 6.1. Unfortunately, these data sets were not adjusted with differential or atmospheric corrections, so the position differences may contain significant error. From the table, there appears to be no consistency in the ECEF y and z-differences amongst the data sets. However, the ECEF x-difference is approximately between -20 and -35 meters in all sets.

Table 6.1: Static ECEF Position Differences between WGS84 and PZ-90.11

Location	**X (m)**	**Y (m)**	**Z (m)**
Auburn, AL	-31.2	-16.7	7.38
Auburn, AL	-35.1	-13.3	3.49
Lee County, AL	-31.5	11.3	-10.1
Des Moines, IA	-23.3	11.6	-20.7
Des Moines, IA	-20.1	-25.0	18.2
Des Moines, IA	-15.6	4.42	-18.5

Based on the information from Table 6.1, the transformation from PZ-90.11 (u_{11}, v_{11}, and w_{11}) to WGS84 is approximated in Equation (6.7).

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} u_{11} \\ v_{11} \\ w_{11} \end{bmatrix} + \begin{bmatrix} -30 \text{ m} \\ 0 \text{ m} \\ 0 \text{ m} \end{bmatrix} \tag{6.7}$$

The transformation involves a translation of 30 meters in the ECEF x-position. A significant amount of additional data would be needed to validate this transformation for official use in the United States. This is the transformation adopted by the software receiver developed in this work.

Figure 6.12 compares GLONASS positioning results in Des Moines, Iowa (left) and Auburn, Alabama (right) with and without the coordinate transformation from Equation (6.7). GPS positioning results are used as a reference. In both cases, the GLONASS position estimates are much more similar to GPS with the adopted coordinate transformation.

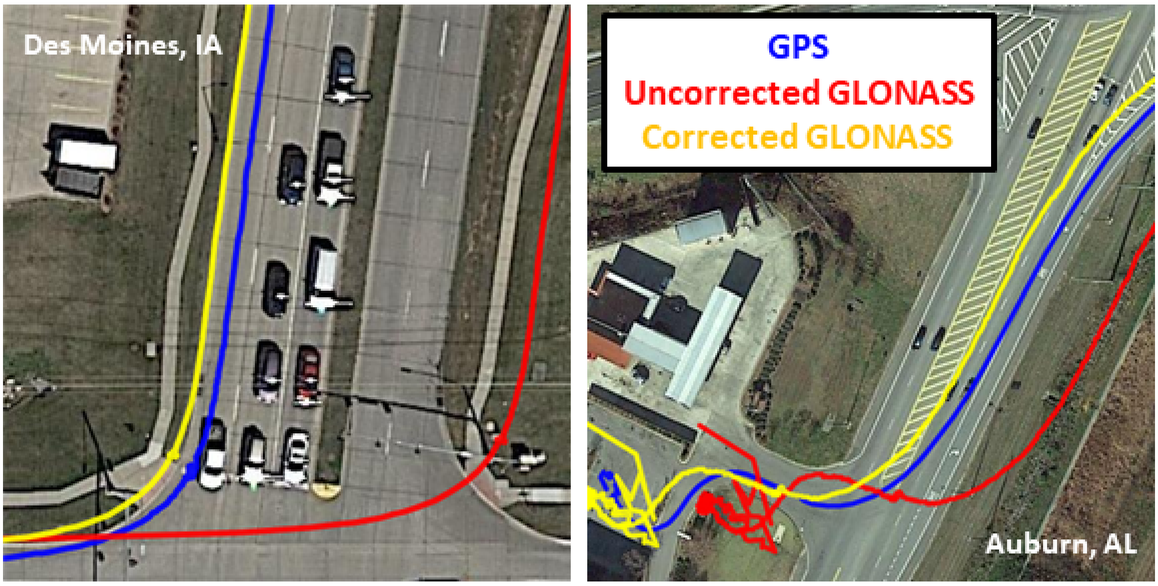

Figure 6.12: GLONASS Positioning Results with the Adopted ECEF Coordinate Transformation

When GPS and GLONASS are both used for navigation, the software receiver transforms GLONASS satellite positions from PZ-90.11 to WGS84, and the solution is estimated in the WGS84 ECEF frame. Figure 6.13 shows positioning results with GPS & GLONASS when the transformation is (corrected) and is not (uncorrected) used. This is the same set of signal data from Figure 6.10. The difference in the corrected and uncorrected position is small, however, only 5 of the 13 satellites in-view to the receiver come from GLONASS. When compared to single constellation positioning in Figure 6.10, using both GPS and GLONASS (with the transformation) gives a more accurate and precise estimate.

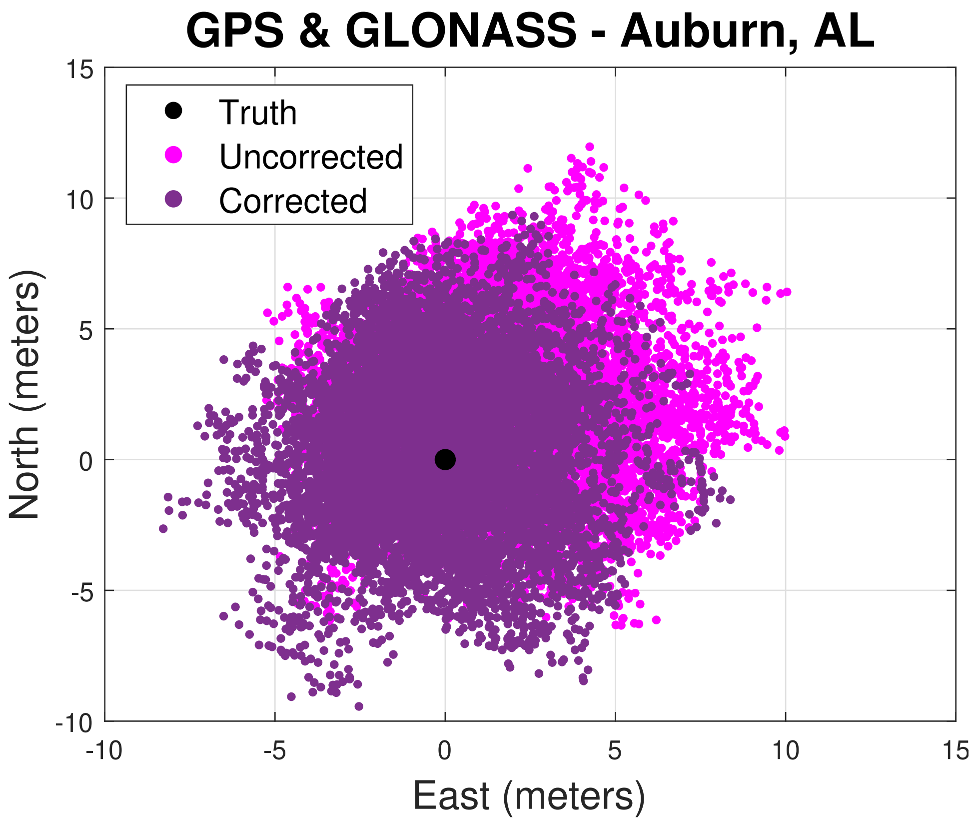

Figure 6.13: Static Positioning in Auburn, AL using GPS & GLONASS

6.2 Simulated Jamming Tests

When GPS and GLONASS are implemented together with vector processing, the constellations aid one another's tracking operations. To prove the software receiver developed in this thesis performs dual-constellation vector tracking, simulated jamming experiments were tested on the receiver. Jamming was simulated by replacing real signal data with random noise. The noise was generated as a zero-mean Gaussian distribution with a standard deviation of 3. Note that this is a very simplistic simulation of jamming that does not consider jammer-to-signal power ratio. The simulation assumes that when a constellation is jammed, the constellations' signals disappear and are replaced with noise at the receiver.

The signal data was recorded by the IFEN SX3 in Auburn, AL. The recording antenna was static for the first half of the experiment and dynamic for the second half, attached to a vehicle driving on the Auburn University campus. Multipath signal degradation occurred from tree foliage at the end of the antenna's route. The antenna was in-view to 8 GPS satellites and 6 GLONASS satellites. Figure 6.14 shows a sky plot of the satellite geometry in the experiment.

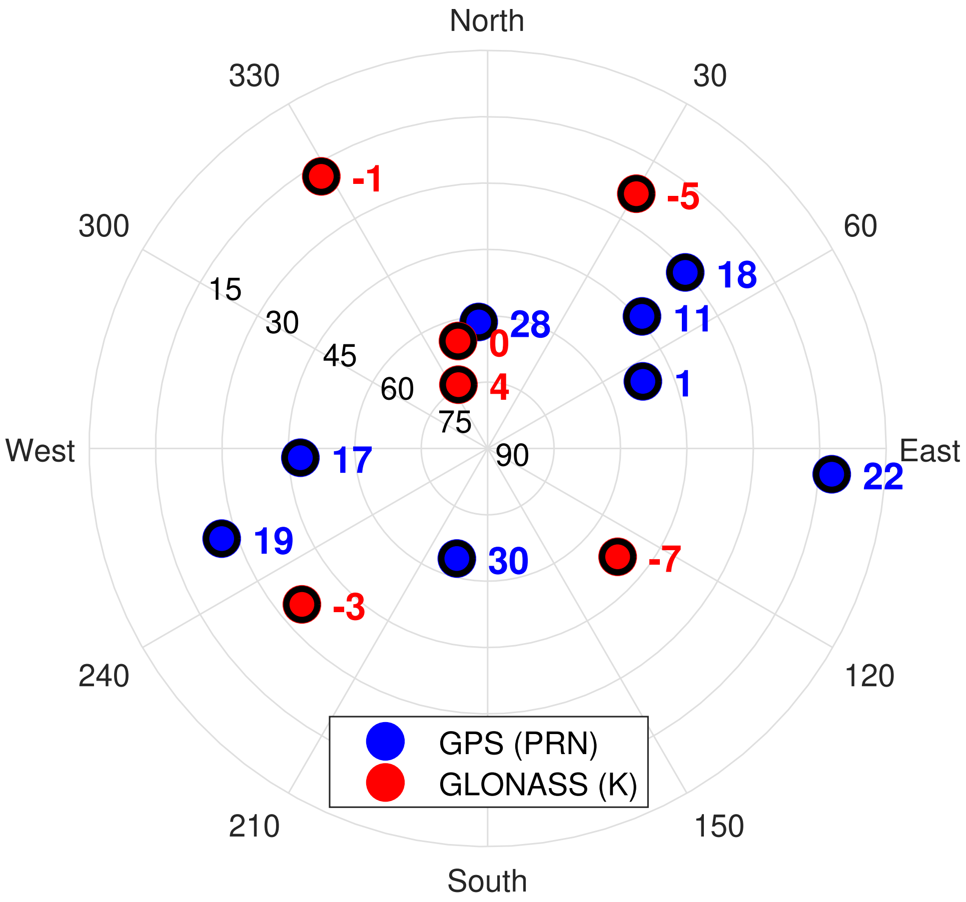

Figure 6.14: Sky Plot of GPS and GLONASS Satellites used in the Jamming Tests

Three tests were performed against the software receiver; GPS jamming, GLONASS jamming, and jamming of both constellations. The GPS jamming test blocks out GPS, while keeping GLONASS available to the receiver. The GLONASS jamming test blocks out GLONASS, while keeping GPS available to the receiver. The GPS and GLONASS jamming test blocks out both constellations. Each test lasted 350 seconds, where jamming was simulated into the signal data for 200 seconds, starting at 75 seconds and ending at 275 seconds.

6.2.1 GPS Jamming Test

When the GPS and GLONASS software receiver experiences GPS jamming, the receiver is able to maintain an accurate position throughout the test. Figure 6.15 shows positioning results for GPS vector tracking, GPS & GLONASS scalar tracking, and GPS & GLONASS vector tracking. When GPS is used alone, the position drifts and becomes inaccurate. The drift occurs because the jammed GPS signals cannot provide tracking lock measurements to the navigation filter. When GPS and GLONASS are both used, GLONASS provides accurate measurements to the estimator, and Fault Detection and Exclusion (FDE) ensures that jammed GPS measurements do not degrade the PVT solution. FDE removes poor measurements from the Kalman filter. More information on FDE can be found in Appendix A. The dual-constellation vector tracking solution is slightly more accurate than the scalar solution, because the GPS measurements reappear (GPS channels relock) after the jamming ends. The GPS & GLONASS scalar tracking receiver is unable to relock onto the GPS channels after the jamming, so the receiver cannot rely on those additional measurements.

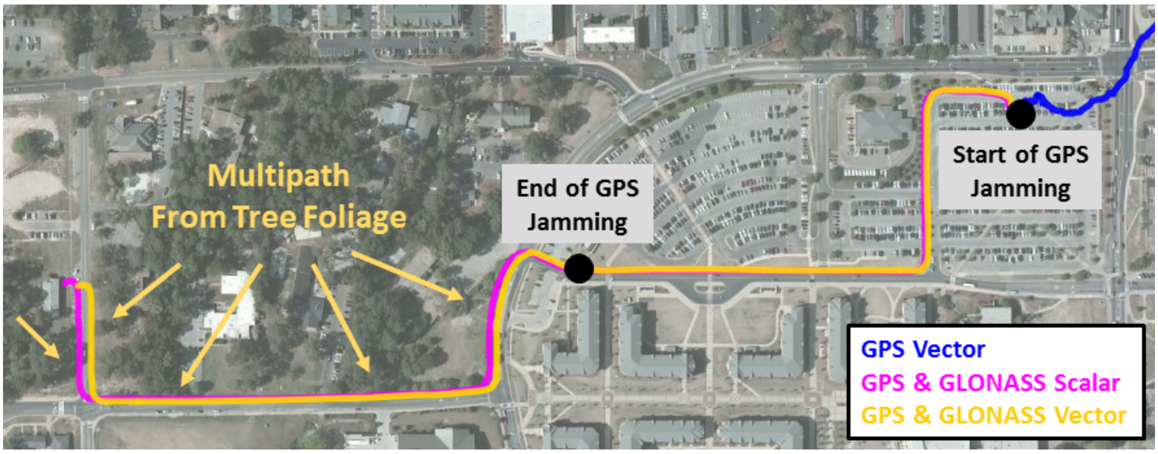

Figure 6.15: Software Receiver Positioning Results with GPS Jamming

Figure 6.16 illustrates the GPS & GLONASS vector receiver's ability to maintain tracking lock on the jammed GPS channels. The figure shows the 8 GPS channels' estimated signal powers during the test. Initially, prior to jamming, the signal powers are high and consistent. During the jamming period, the signal powers drop to the noise power level. After jamming ends, the channel powers reappear. There is some smearing of the signal powers after the jamming, which is due to multipath degradation. Similar results to Figure 6.16 can be found in [68] without multipath degradation. Throughout the jamming period, the 6 GLONASS channels were able to maintain accurate tracking replicas of the 8 GPS channels through the navigation estimator, which allowed the GPS channels to relock.

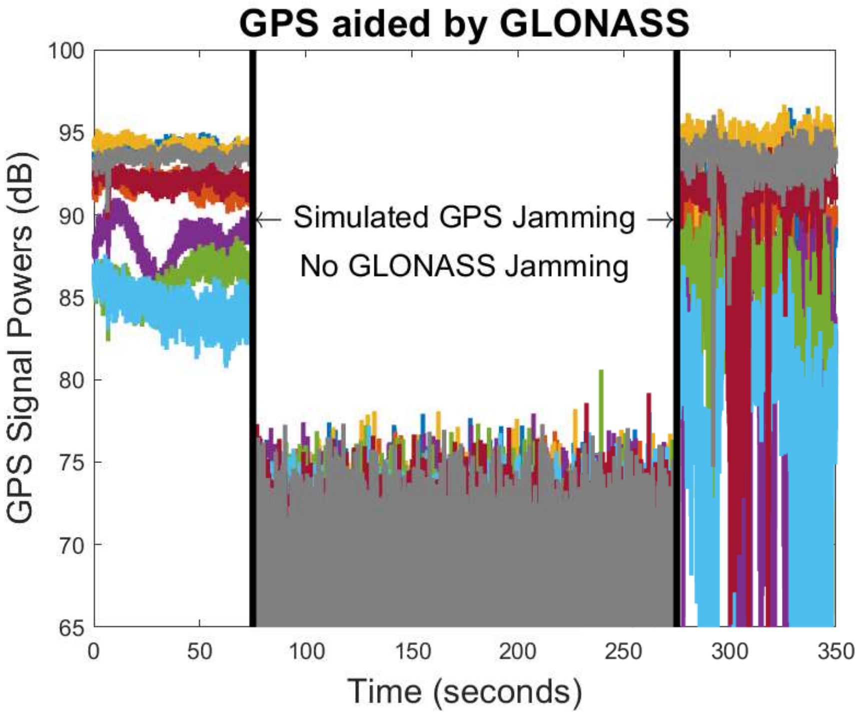

Figure 6.16: Jammed GPS Channels' Signal Powers using the Software Vector Receiver

Figure 6.17 illustrates the advantage of vector tracking over scalar tracking when using GPS and GLONASS in the software receiver. The figure shows the Kalman filter estimated ECEF position variances for vector and scalar tracking architectures. Prior to jamming, the variances converge and are approximately the same for the scalar and vector algorithms. When jamming occurs, the scalar and vector variances increase and converge to new values. The variances increase because both receivers lose access to GPS measurements during the jamming period. After jamming ends, the vector tracking variances drop, because the vector receiver relocks onto the GPS channels and uses their measurements. The scalar tracking variances stay the same after jamming, because the scalar receiver lost lock of the GPS channels and cannot use their measurements. There is some oscillation in the estimated variances at the end of the experiment due to multipath.

Prior to and during the jamming test, the scalar position variances in Figure 6.17 are smaller than the vector position variances in the ECEF y and z-directions. This is due to the nature of noise sharing in the vector tracking algorithm. In scalar tracking, the loop filters remove noise from the discriminators independently on each channel. In vector tracking, the discriminators are filtered through the navigation estimator in which noise from the channels' discriminators may be shared amongst each other and the PVT solution. Prior to jamming, in the clear sky environment, the loop filters in the scalar receiver provide better filtering than the navigation processor in the vector receiver. During the jamming, the discriminators are generated from pure noise. The scalar receiver attenuates these full noise discriminators with loop filters while the vector receiver directly navigates with them. The discriminators are also less likely to be removed by FDE in vector tracking because the GLONASS channels maintain accurate pseudorange and pseudorange rate measurements of the jammed GPS channels. This is not the case in scalar tracking, the GPS measurements will become poor and thrown out by FDE.

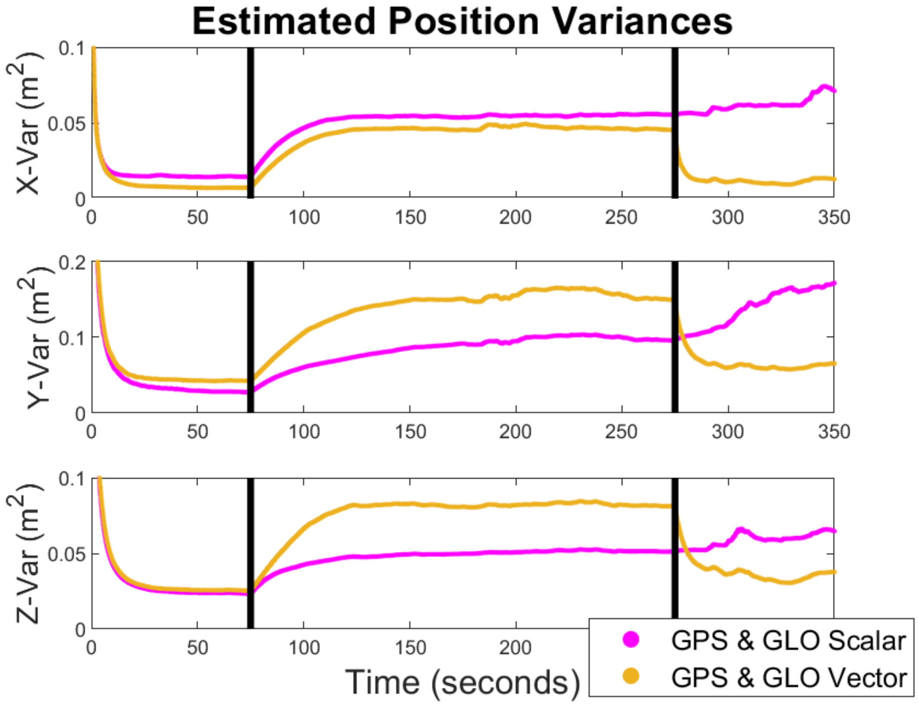

Figure 6.17: Position Variances using GPS & GLONASS Scalar and Vector Tracking

6.2.2 GLONASS Jamming Test

The previous test showed that GLONASS can aid GPS tracking through vector processing. The same is true for the case when GLONASS is jammed; GPS can aid GLONASS tracking using the dual-constellation vector receiver. Figure 6.18 shows the GLONASS channels' estimated signal powers during the jamming test with the vector receiver. The 6 GLONASS channels are able to relock onto their satellite signals due to tracking aid from the 8 GPS channels. This is seen by the spike in the GLONASS signal powers after the jamming period ends. As stated previously, the smearing of the signal powers is due to multipath when the test vehicle drove near tree foliage.

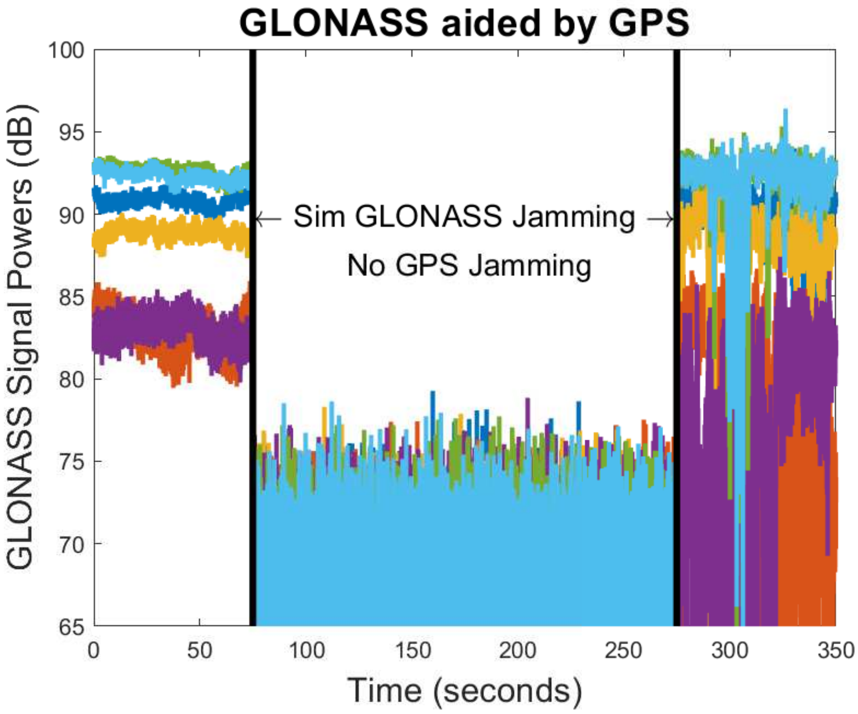

Figure 6.18: Jammed GLONASS Channels' Signal Powers using the Software Vector Receiver

Carrier tracking performance during the GLONASS jamming test was analyzed for the different receivers. The Kalman filter estimated clock drift variances, shown in Figure 6.19, are used to simplify the analysis of carrier performance. The figure shows clock drift variances for GLONASS vector tracking, GPS & GLONASS scalar tracking, and GPS & GLONASS vector tracking. Note that the vector receivers use VPLL's, where the centralized VFLL is combined with PLL aiding on each satellite channel.

Initially, prior to the jamming, the GPS & GLONASS vector variance is lower than the GLONASS vector variance. This occurs because the dual constellation vector receiver uses more channels in its VFLL than the GLONASS vector receiver. The GPS & GLONASS scalar variance is lower than its vector counterpart, which is due to noise sharing in the vector tracking algorithm.

During the jamming period, the GPS & GLONASS variances converge to higher magnitudes due to the to the drop in satellite measurements. The GLONASS vector variance continually increases over the period due to a degrading, inaccurate clock drift solution. At the end of the jamming test, the GLONASS vector recevier begins to settle to a new clock drift variance, which indicates the receiver's Kalman filter is most likely using the model to predict the clock drift. The GLONASS vector measurements have very high measurements and may be thrown out by FDE, therefore providing little input to the Kalman filter's estimation. After the jamming ends, the GLONASS vector variance begins increasing again, indicating loss of lock on the satellite signals. The GPS & GLONASS vector variance drops after jamming as relocked GLONASS channels provide measurements to the VFLL. The GPS & GLONASS scalar variance stays approximately the same as the scalar receiver cannot relock onto the GLONASS channels.

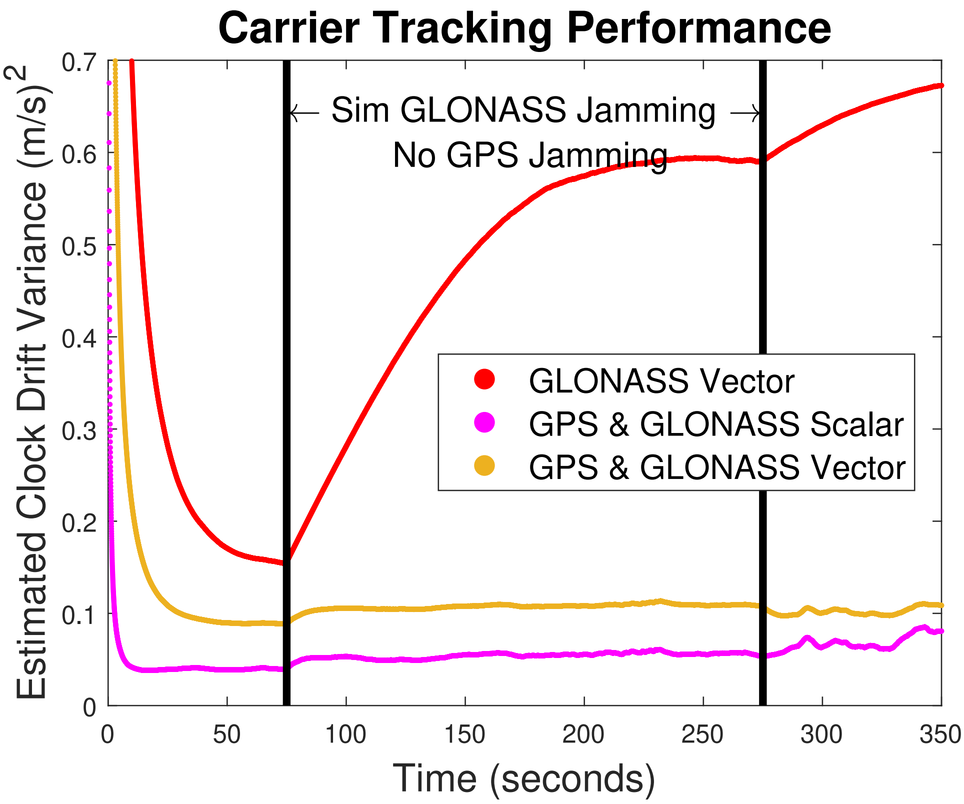

Figure 6.19: Clock Drift Variances for Different Receivers in the GLONASS Jamming Test

In the GLONASS jamming test, carrier tracking performance is better when the dual constellation software receiver uses scalar over vector processing. The channels' carrier replicas are more accurate when using individual PLL loop filters without including the centralized VFLL. The VFLL shares tracking information amongst all channels, which includes those that are jammed or degraded by multipath. FDE mitigates noise sharing in the VFLL, but not completely. However, without the VFLL, the GLONASS channels could not be relocked after jamming without performing signal acquisition.

6.2.3 GPS and GLONASS Jamming Test

In the case when both GPS and GLONASS are jammed, the software receiver fails. Figure 6.20 shows the position error for scalar and vector GPS & GLONASS receivers during the test. Once jamming begins, the position errors grow drastically for both receivers. After jamming ends, the position errors continue to grow, indicating that tracking lock has been lost on the GPS and GLONASS satellite channels.

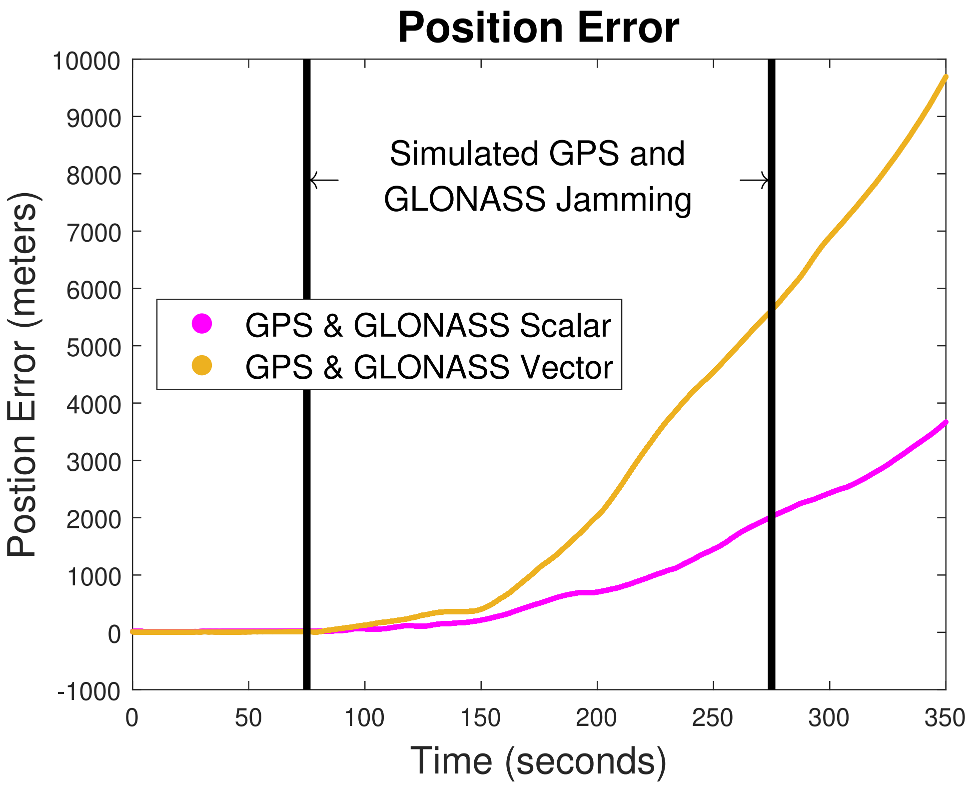

Figure 6.20: GPS & GLONASS Position Errors when Both Constellations are Jammed

As with any vector tracking filter, the GPS & GLONASS receiver will fail to maintain lock on the satellite signals without an accurate navigation solution. When the receiver's navigator only relies on GPS and GLONASS measurements, the receiver must have at least 4 observable satellite channels to maintain the PVT estimation. This condition was satisfied when only GPS or GLONASS was jammed. However, when both constellations are jammed, this condition is not satisfied. The receiver would need information from another sensor, such as an IMU, or another constellation that is not jammed to maintain accurate tracking replicas on the GPS and GLONASS channels.

6.3 Experimental Performance Results

The GPS & GLONASS vector software receiver was tested in different environments for performance analysis. Signal data was collected in a vehicle using the IFEN SX3 front-end and a Novatel pinwheel antenna. Figure 6.21 shows pictures of the data collection system. The receiver was tested in two different geographical areas; Auburn, Alabama and Des Moines, Iowa.

Figure 6.21: Setup of Signal Data Collection with Vehicle

During the experiments, data was also collected from a Ublox EVK-M8T receiver, shown in Figure 6.22. The Ublox receiver has the capability to position with both GPS and GLONASS, providing a good comparision with the software receiver in this work. Like the IFEN SX3, the Ublox also operates with a TCXO clock. From the receiver's product summary sheet [69], Ublox claims the receiver uses an advanced tracking algorithm that gives a tracking and navigation sensitivity of -167 dBm. This may mean the Ublox receiver also performs vector tracking or a form of navigation-aided tracking.

The product summary sheet also states the Ublox uses Satellite Based Augmentation Systems (SBAS). SBAS provides the Ublox with satellite orbit, satellite clock, and ionospheric corrections that improve the receiver's position estimate. Since the signal data was collected in the United States, Ublox is most likely using the Wide-Area Augmentation System (WAAS) to provide corrections for its GPS satellite channels. According to the Federal Aviation Administration, WAAS provides receivers with a position accuracy of approximately 1 meter in the United States and Canada [16]. The software receiver designed in this thesis does not use WAAS, hence the Ublox receiver has an advantage over the the software receiver.

The Ublox EVK-M8T was configured with a dynamic automotive state model for its estimator during the performance runs. The Ublox outputs position estimates every second (1 Hz), while the receiver outputs navigation and tracking results every 20 milliseconds (50 Hz).

Figure 6.22: Picture of the GPS and GLONASS Ublox Receiver tested in the Experiments

The GPS and GLONASS receivers were tested in four different challenging signal environments. The environments included light tree foliage and heavy tree foliage that induced multipath, a parking deck that attenuated satellite signal power, and an urban canyon that contained multipath reflections and signal blockages. In each experiment, the vehicle starts in an open sky environment to allow the software receiver to initialize with an accurate navigation solution. The GPS & GLONASS software receiver is compared to the Ublox M8T, as well as when the software receiver is configured to only use a single GNSS constellation.

6.3.1 Light Tree Foliage Results

Signal data was collected in a small residential neighborhood with light tree foliage in Clive, Iowa. The tree foliage is considered light because the sides of the test vehicle are covered by trees, and the antenna overhead is open sky. The collection was also performed during early spring when leaves were not on the trees yet. Figure 6.23 shows a picture of the residential road driven by the test vehicle.

Figure 6.23: Picture of Light Tree Foliage Environment in Clive, IA

The software receiver operated with 9 GPS satellites and 7 GLONASS satellites. Figure 6.24 shows a sky plot of the GPS and GLONASS satellites used in the experiment. Table 6.2 provides satellite geometry DOP values for the experiment.

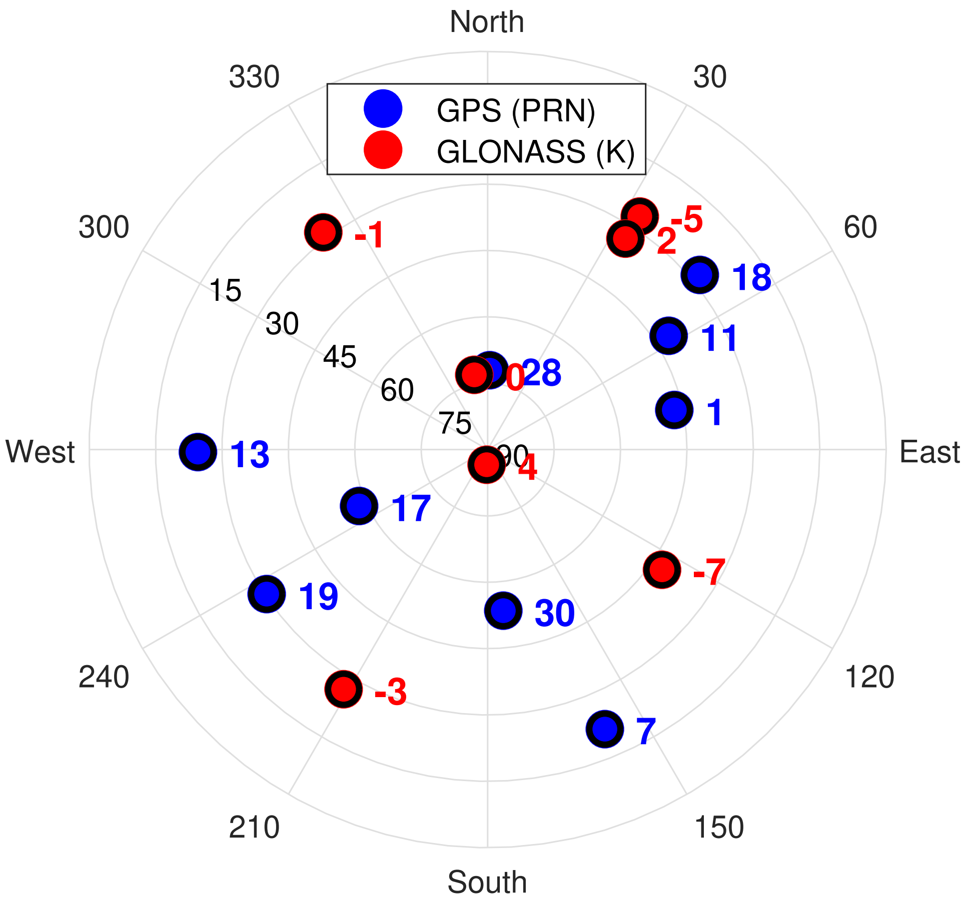

Figure 6.24: Sky Plot of GPS and GLONASS Satellites used in Clive, IA

Table 6.2: DOP Values for the Light Tree Foliage Experiment

Constellation	**PDOP**	**HDOP**	**VDOP**	**TDOP**
GPS	2.0040	1.0101	1.7308	1.1709
GLONASS	2.0938	1.1412	1.7555	1.2244
GPS & GLONASS	1.4021	0.7103	1.2088	0.8166

Figure 6.25 shows positioning results for the GPS & GLONASS software receiver when using scalar and vector tracking in light tree foliage. The software receiver is compared to the Ublox receiver. All the receivers maintain accurate positioning results throughout the experiment. Ublox provides the most accurate estimations and is used as a reference when comparing the different configurations of the software receiver.

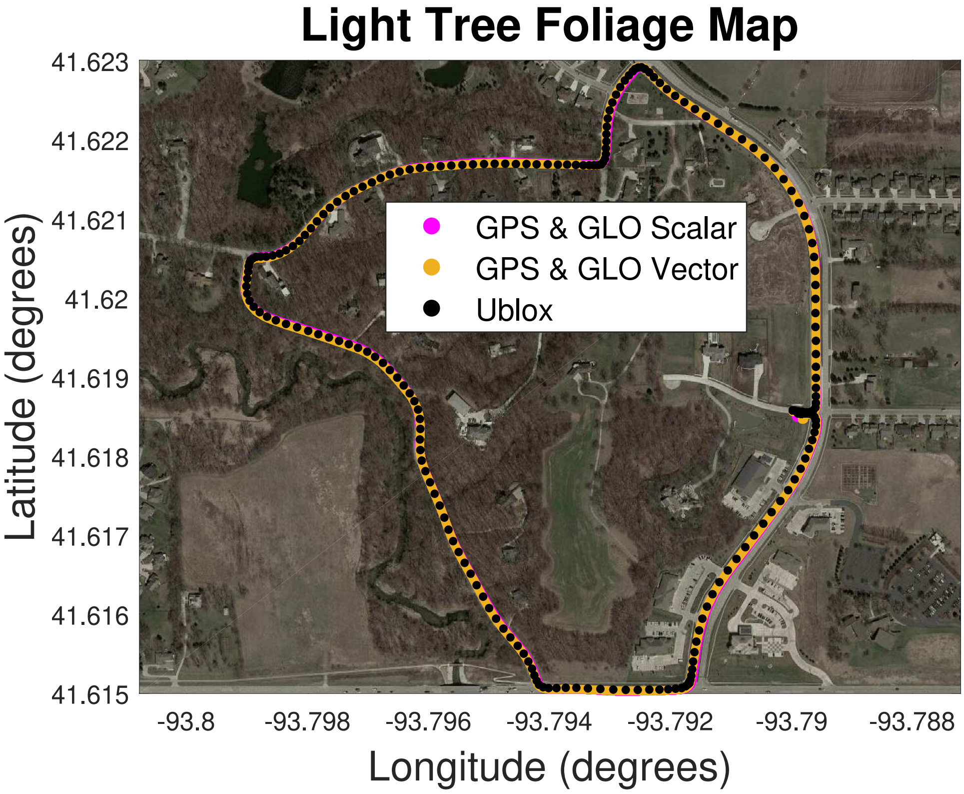

Figure 6.25: GPS & GLONASS Positioning Results in Clive, IA

Figure 6.26 shows the software receiver estimated C/N_O ratios for the GPS and GLONASS channels in the experiment. When the receiver is in the light tree foliage area, the C/N_O ratios are smeared by the multipath reflections that enter the antenna. The software receiver is able to withstand the multipath as the C/N_O ratios are stable after the receiver exits the forested area.

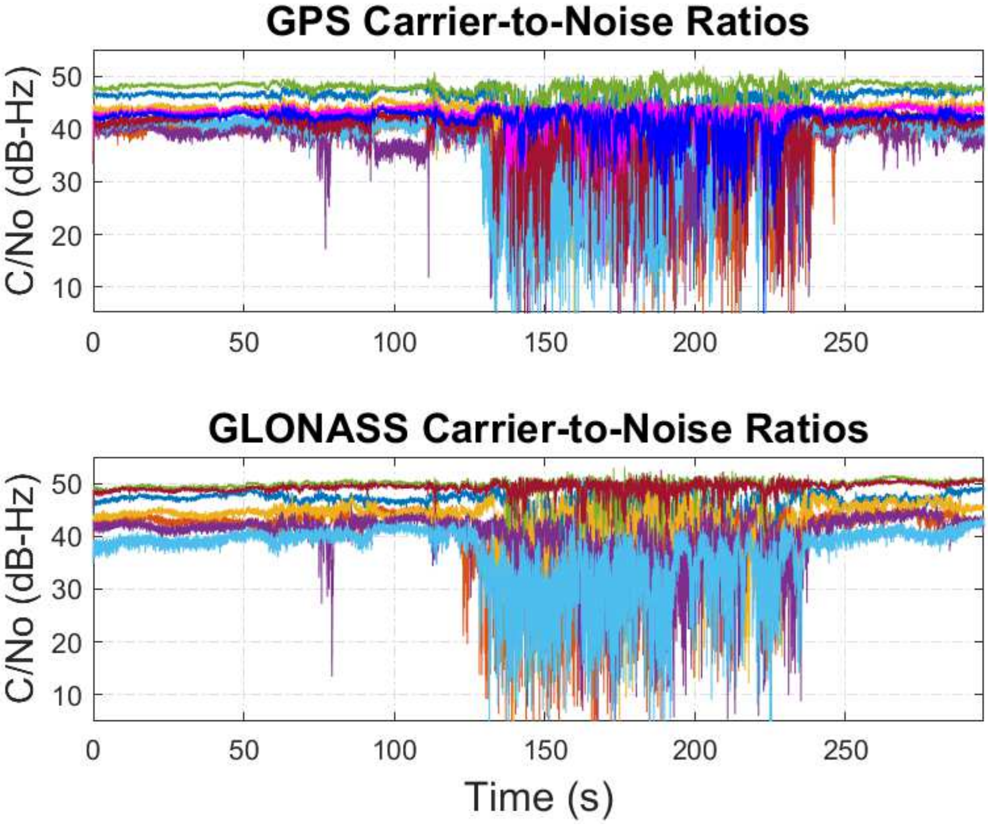

Figure 6.26: GPS and GLONASS C/N_O Ratios in the Light Tree Foliage Experiment

Figure 6.27 compares position estimations when the software receiver performs GLONASS vector tracking, GPS vector tracking, GPS & GLONASS scalar tracking, and GPS & GLONASS vector tracking. The GLONASS vector tracking estimations are the least accurate with respect to the Ublox. At times, the GLONASS vector tracking estimations are biased, which is from the multipath and error in the adopted coordinate transformation. The GPS vector tracking, the GPS & GLONASS scalar tracking, and the GPS & GLONASS vector tracking estimations are approximately the same in accuracy, matching well with the Ublox positions. Because the signal environment is not significantly harsh, GPS & GLONASS positioning matches up with GPS positioning. Although GLONASS positioning is biased, it does not greatly affect the combined solution.

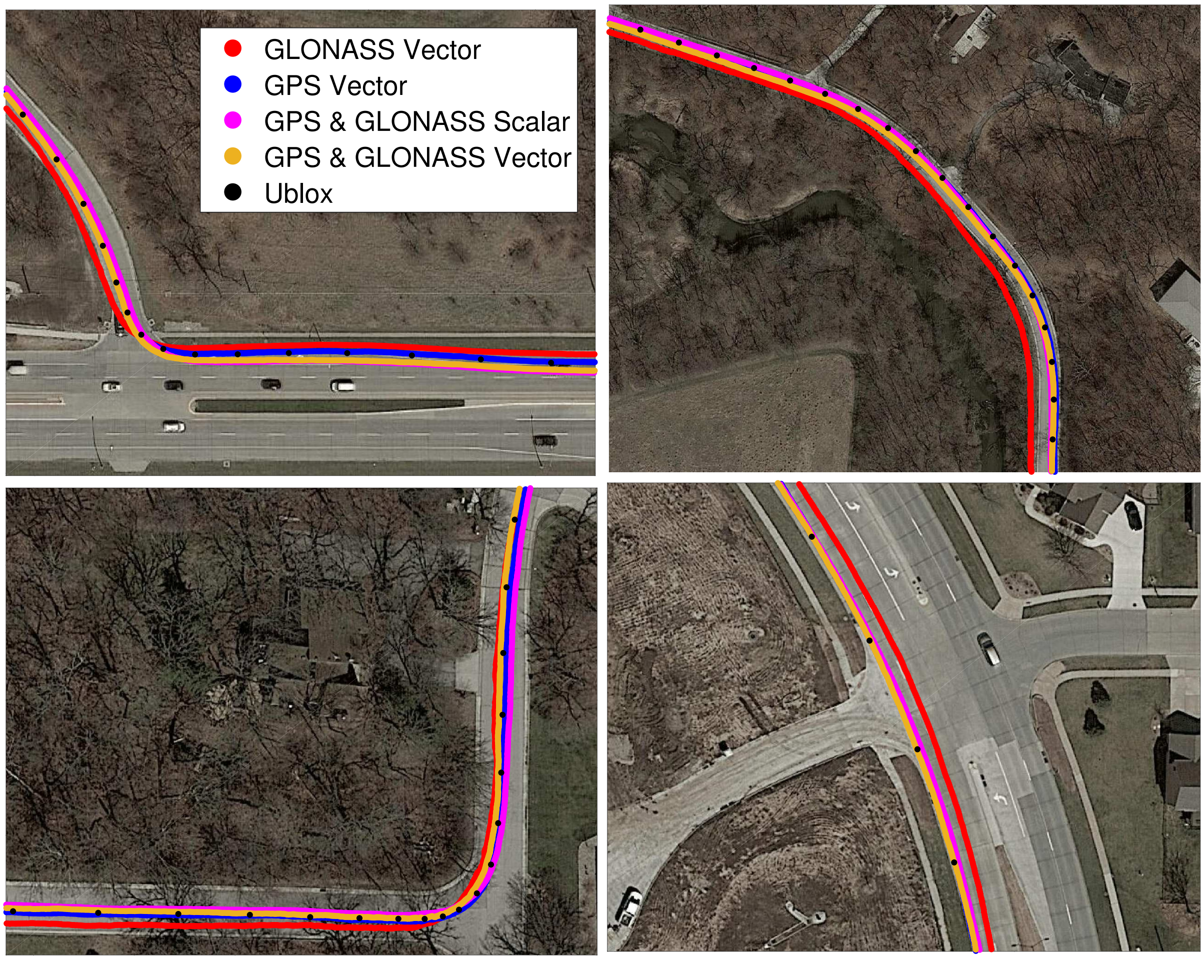

Figure 6.27: Light Tree Foliage Positioning Results

Figure 6.28 shows the Kalman filter estimated ECEF position variances for the different software receiver configurations tested. All of the configurations' estimated variances increase where the receiver channels' C/N_0 ratios decreased in Figure 6.26. The Kalman filters have a higher uncertainty in their position estimations when the receiver experiences multipath. The GPS and GPS & GLONASS configurations maintain stable variances, where they are approximatley the same for each software platform. Their variances also drop after the receiver exits the forested area. The GLONASS position variances grow after the receiver exits the forest, indicating a decrease in position accuracy. This suggests that the adopted coordinate transformation may not be the main contributor to the GLONASS position error.

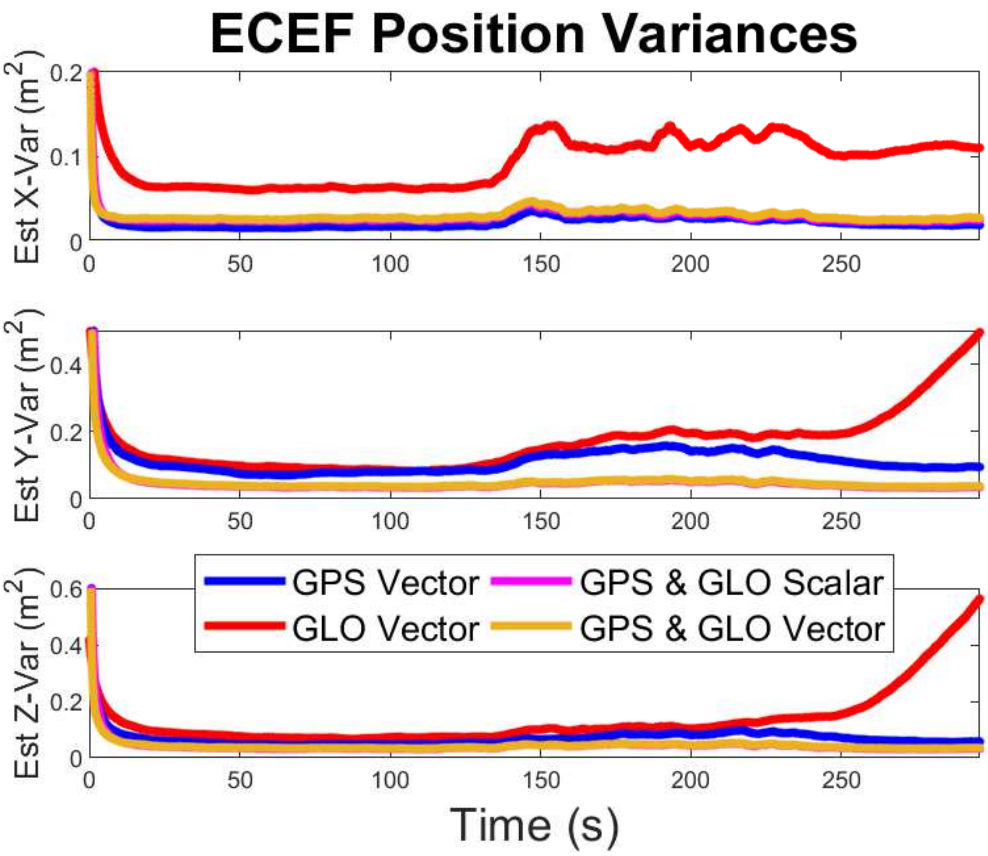

Figure 6.28: Estimated Position Variances in the Light Tree Foliage Experiment

Figure 6.29 shows altitude positioning results for the software receiver when operating with GLONASS vector tracking, GPS vector tracking, GPS & GLONASS scalar tracking, and GPS & GLONASS vector tracking. Altitude results from the Ublox are also shown in the figure. In general, because the satellites are overhead of the receiver, altitude positioning is poor in GNSS.

In the light tree foliage experiment, the vehicle antenna starts and ends at the same location. Hence, the altitude should be the same at the start and end of the experiment. It is seen that positioning with only GPS or GLONASS results in poor altitude estimations. The altitude difference from start to end is large for the single constellation receivers, indicating poor altitude estimation. The GLONASS altitude appears to destabilize, which agrees with the position variance results in Figure 6.28. Altitude positioning is most likely worse for GLONASS than GPS due to satellite geometry in the experiment and the limited positioning accuracy of GLONASS. The accuracy of positioning with GNSS BPSK signals is approximately 10% of the PRN code chip width. Since the GLONASS PRN code is transmitted at half the rate of the GPS C/A code, theoretically, GLONASS has half the position accuracy of GPS.

The altitude results from the Ublox appear to be accurate, as the estimates are the same at the start and end of the experiment. The Ublox receiver, even when using GPS & GLONASS for positioning, cannot achieve this accuracy without differential corrections. This strongly suggests that the Ublox takes advantage of WAAS, giving the commerical receiver an advantage over the software receiver.

Although still poor when compared to the Ublox, the GPS & GLONASS altitude estimates are a major improvement from using GPS or GLONASS alone. Scalar tracking is slightly more accurate in altitude than vector tracking when using both constellations. These results show that navigating with both constellations can improve altitude positioning.

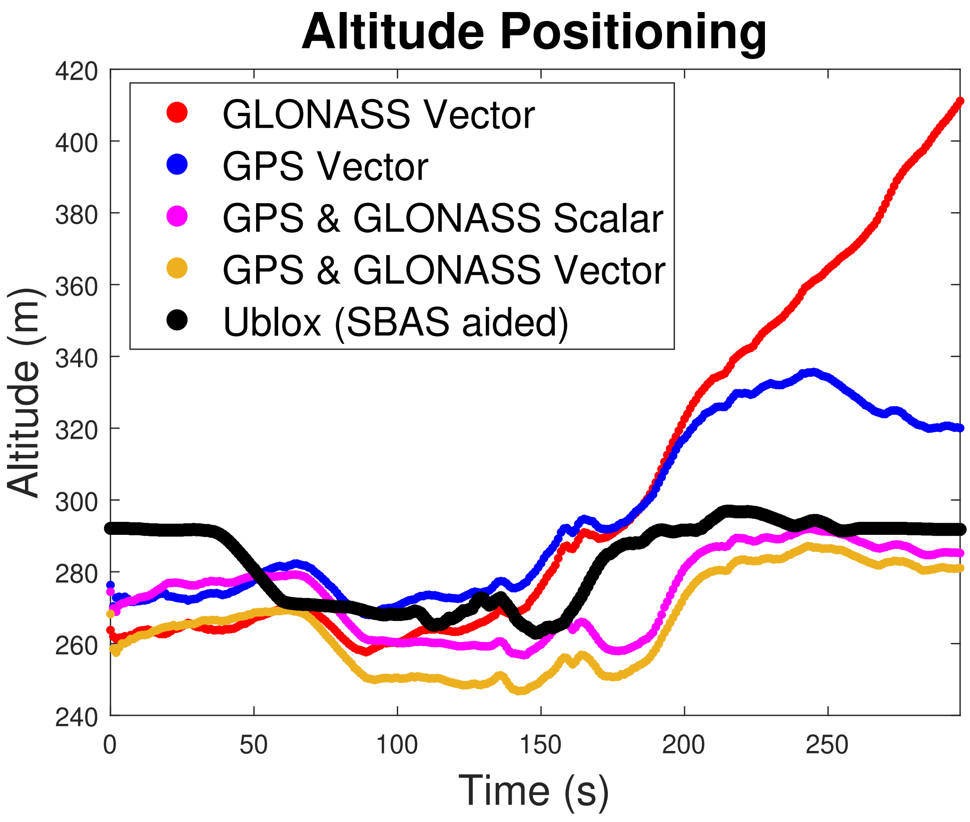

Figure 6.29: Estimated Altitudes in the Light Tree Foliage Experiment

Figure 6.30 shows carrier Doppler frequency estimations for GPS channel PRN #19 and GLONASS channel K #2 in the experiment. The results are shown when the channel uses scalar tracking, single constellation (GPS or GLONASS) vector tracking, and dual constellation (GPS and GLONASS) vector tracking. The Doppler estimations become significantly noisy at the middle of the experiment due to multipath from the light tree foliage. The architectures behave similarly because they all use loop filters with identical noise bandwidth tunings on the PLL.

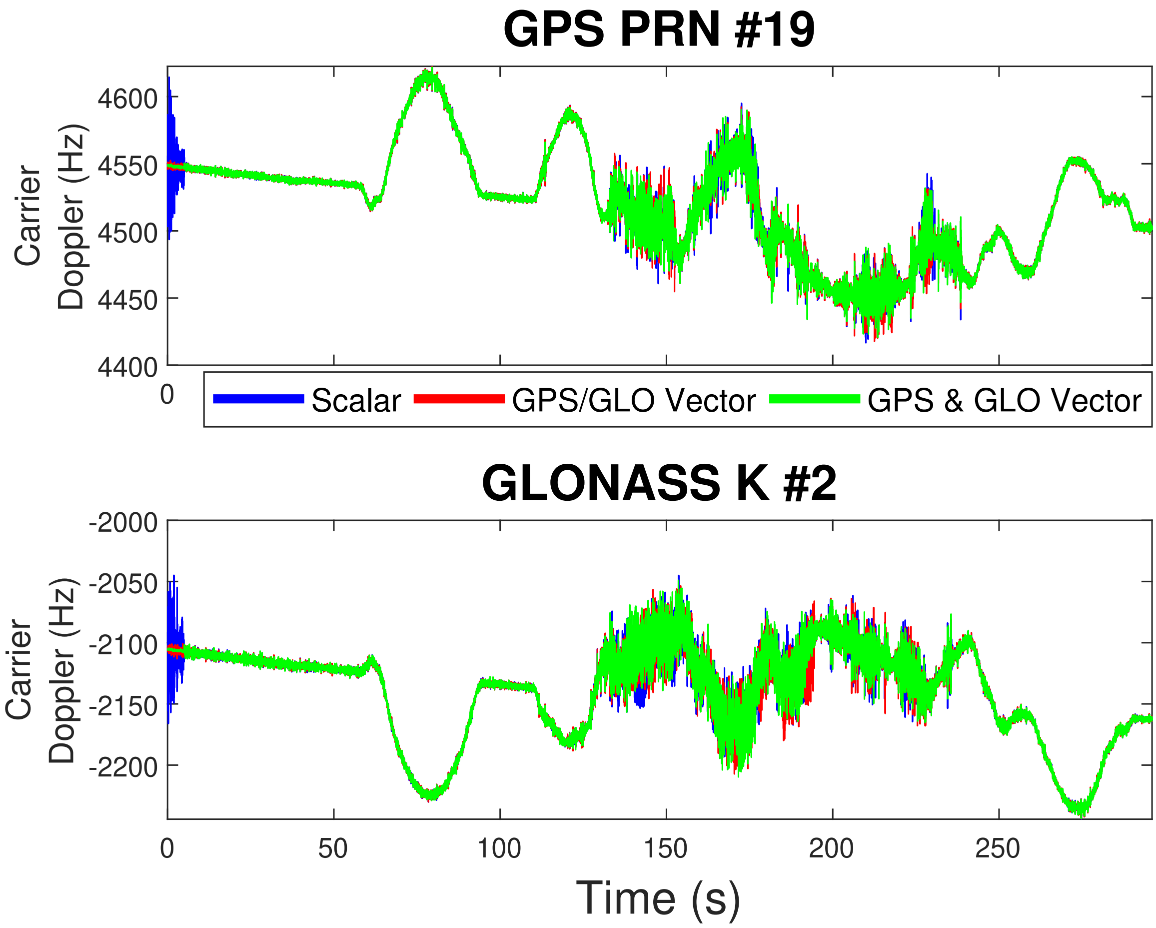

Figure 6.30: GPS and GLONASS Carrier Doppler Frequencies in Light Tree Foliage

6.3.2 Heavy Tree Foliage Results

The test vehicle drove a route in rural Alabama to collect signal data in the presence of heavy tree foliage. Initially, the vehicle starts on a state highway and moves onto a county road that enters Tuskegee National Forest. The tree foliage is considered heavy because the sides of the road are densely covered by trees with leaves. At points in the route, tree foliage blocks out the entire road from the sky. Figure 6.31 provides photos of the state highway (left) and county road (right) that the signal data was collected on. At the end of the run, the test vehicle stops on a bridge that crosses an interstate. The interstate bridge is clear from the tree foliage and contains an open sky environment.

Figure 6.31: Pictures of Heavy Tree Foliage Environment in Rural Alabama

The software receiver operated with 7 GPS satellites and 7 GLONASS satellites. Figure 6.32 shows a sky plot of the GPS and GLONASS satellites used in the experiment. Table 6.3 provides satellite geometry DOP values for the experiment.

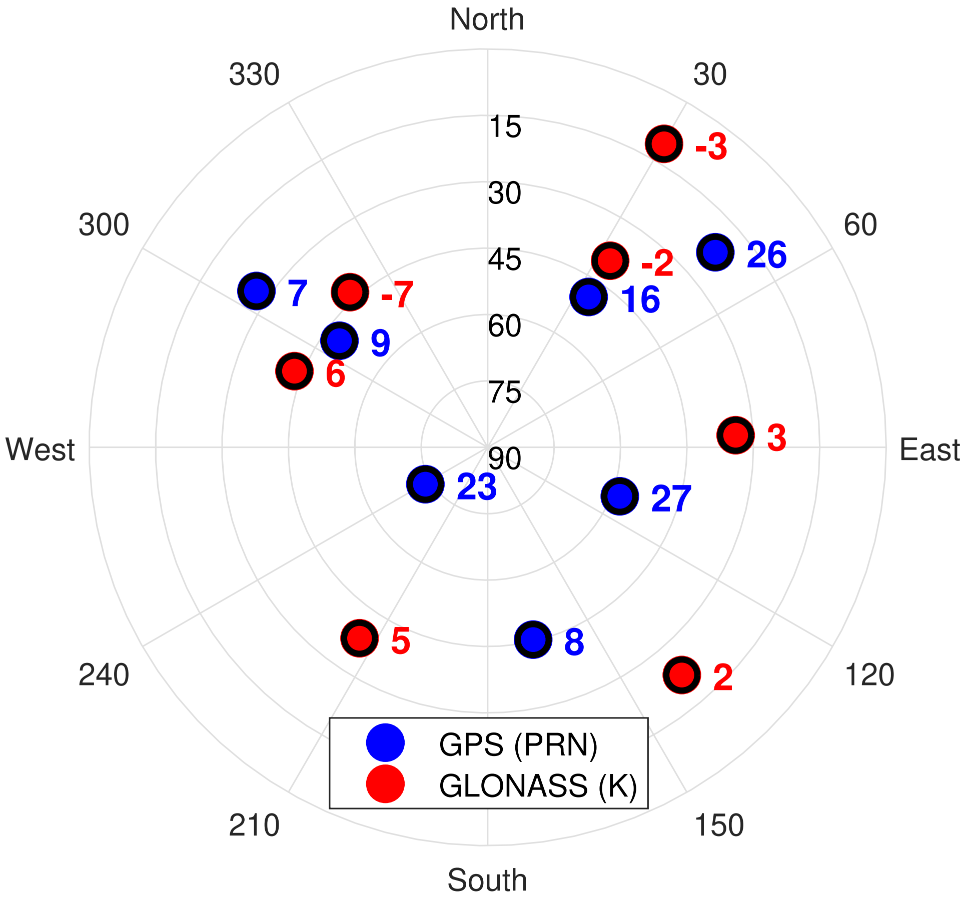

Figure 6.32: Sky Plot of GPS and GLONASS Satellites used in Tuskegee National Forest, AL

Table 6.3: DOP Values for the Heavy Tree Foliage Experiment

Constellation	PDOP	HDOP	VDOP	TDOP
GPS	2.7156	1.2795	2.3953	1.8002
GLONASS	2.8943	1.0489	2.6976	1.5536
GPS & GLONASS	1.6670	0.7584	1.4845	0.9901

Figure 6.33 shows positioning results for the GPS & GLONASS software receiver using scalar and vector tracking in heavy tree foliage. The software receiver results are compared to the Ublox, which maintains an accurate position on the road. It is uncertain whether the Ublox outperforms the software receiver because of WAAS or better multipath filtering. The software receiver can maintain stable positioning in both scalar and vector modes.

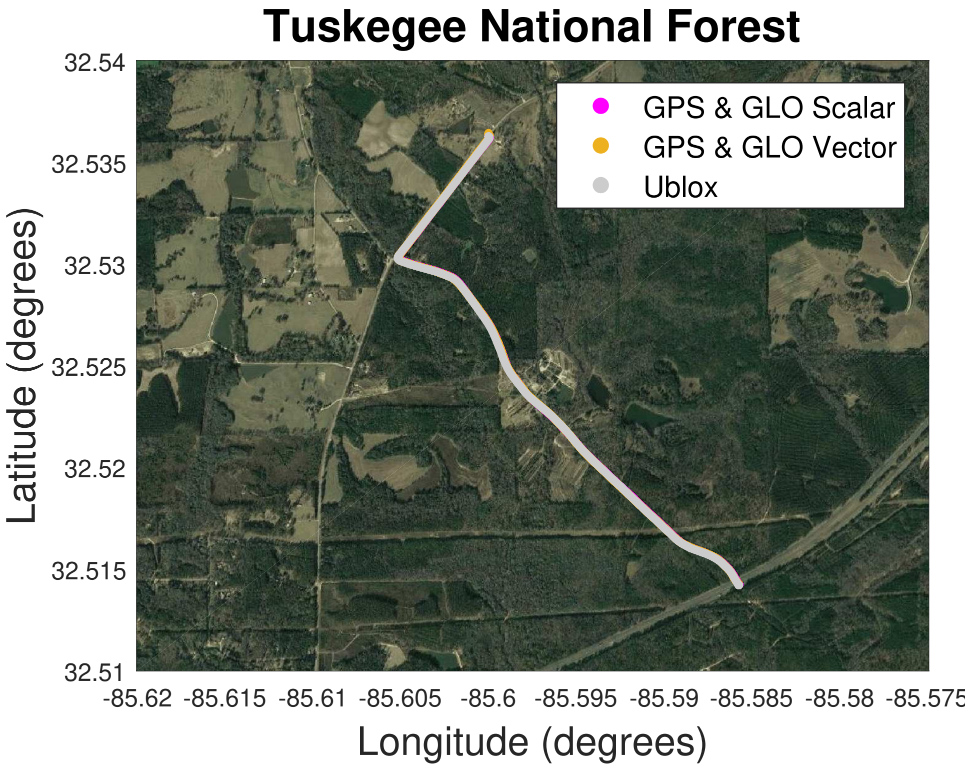

Figure 6.33: GPS & GLONASS Positioning Results near Tuskegee National Forest

Figure 6.34 shows the receiver estimated C/N_O ratios for the GPS and GLONASS channels in the experiment. As in the light tree foliage experiment, the heavy tree foliage smears the channels' C/N_O ratios. The smearing effect is worse in the heavy tree foliage due to stronger multipath effects. The GPS and GLONASS C/N_O ratios restabilize when the receiver reaches the clear sky interstate bridge, indicating the software receiver can maintain track of the satellite signals in the heavy forested area.

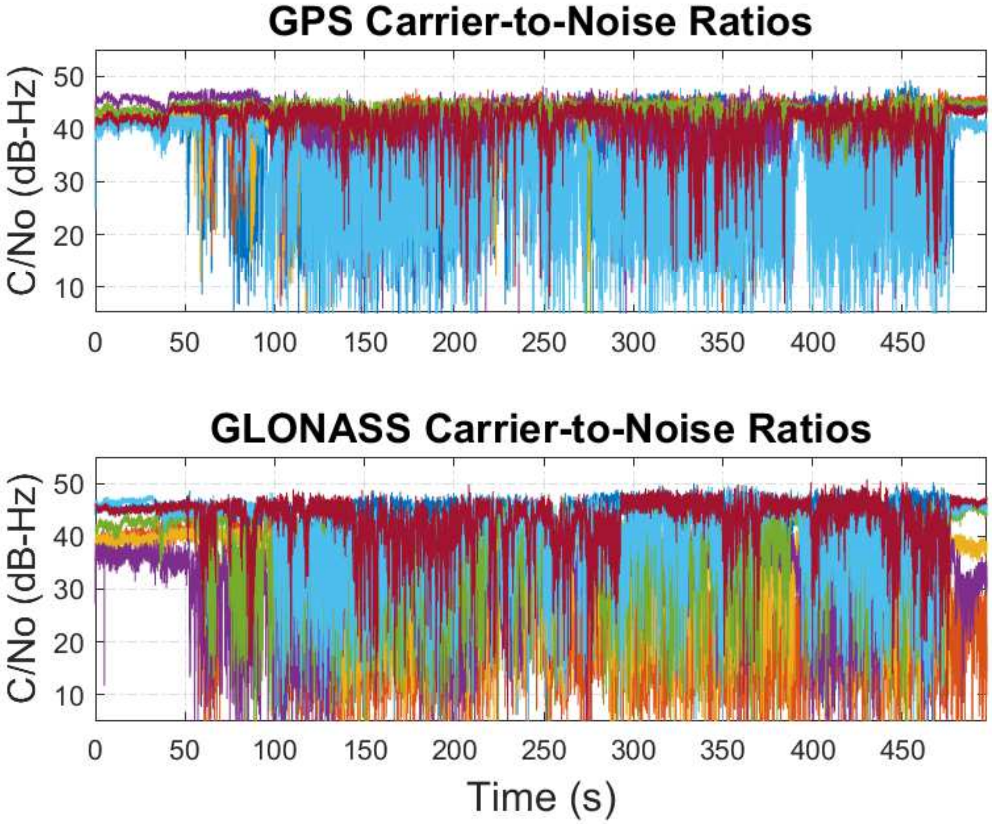

Figure 6.34: GPS and GLONASS C/N$_0$ Ratios for the Heavy Tree Foliage Experiment

Figure 6.35 shows positioning results for the software receiver when using scalar tracking with GPS, GLONASS, and both constellations. GPS outperforms GLONASS throughout the test. The GLONASS position estimates are biased as they were in the light tree foliage experiment. Over time, the GPS position results become more biased due to the multipath induced by the forest. By the time the receiver reaches the open sky bridge, the GPS position is so biased that is not contained onto the bridge. The GLONASS position does not make it to the bridge because the multipath delayed the response so significantly. When scalar tracking is performed with both GPS and GLONASS in the software receiver, the position estimation follows the Ublox more accurately. GPS & GLONASS scalar tracking is able to maintain a biased position from the Ublox on the interstate bridge at the end of the test.

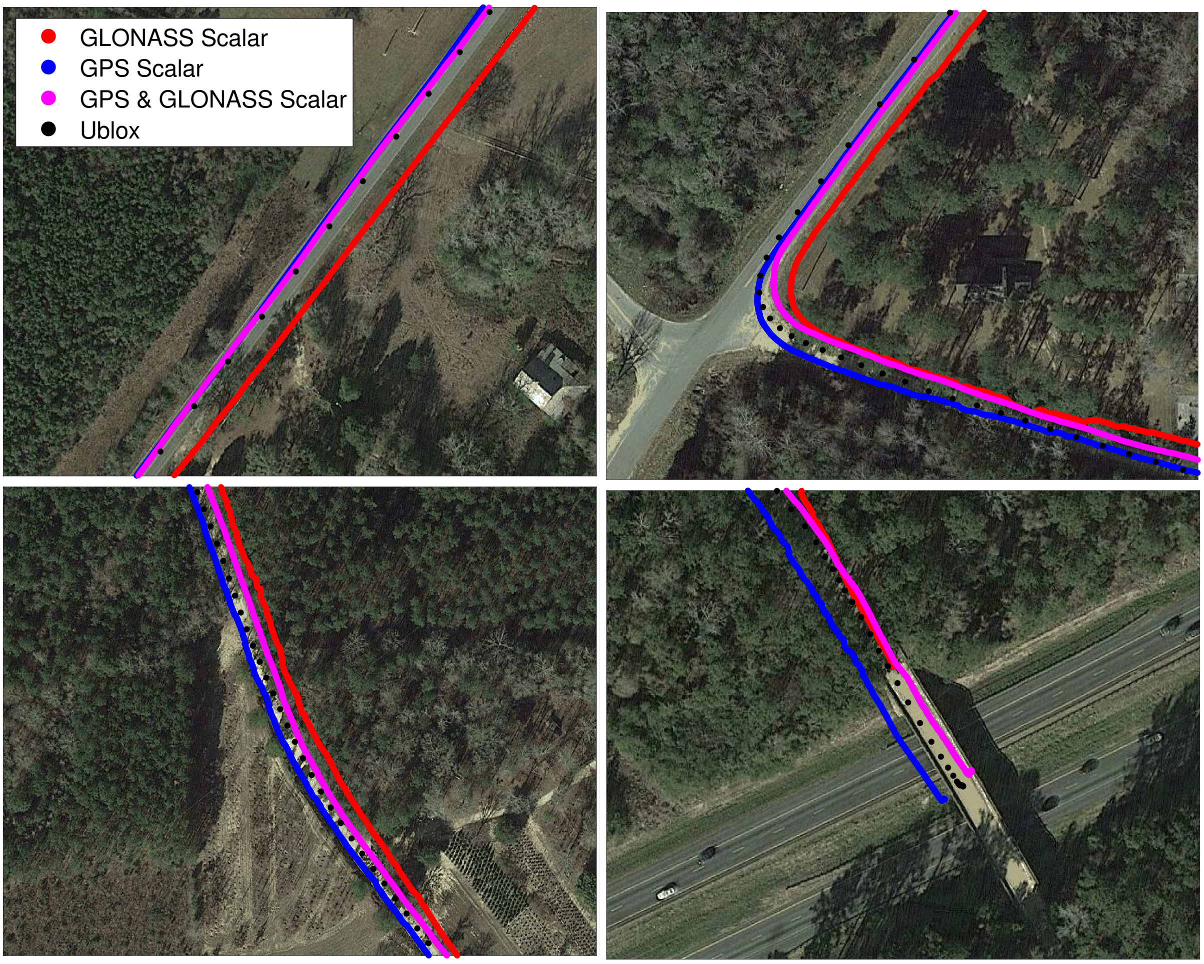

Figure 6.35: Heavy Tree Foliage Positioning Results with Scalar Tracking

Figure 6.36 shows positioning results for the software receiver when using vector tracking with GPS, GLONASS, and both constellations. Overall, each configuration is improved by replacing scalar tracking with vector tracking. Each software configuration follows the Ublox receiver more accurately by using vector processing. Coupling the satellite channels together through the navigator attenuates more multipath error than the scalar loop filters. A scalar loop filter can drop its noise bandwidth to mitigate multipath, but it then may be unable to track the signal dynamics. The GLONASS configuration sees the biggest improvement by using vector tracking, but still contains biased estimates. The GPS position is still biased at the interstate bridge, but the bias is less significant than it was when performing scalar processing. The GPS & GLONASS vector tracking receiver is able to produce a position estimate that is very close to the Ublox at the interstate bridge.

Figure 6.36: Heavy Tree Foliage Positioning Results with Vector Tracking

Figure 6.37 shows Kalman filter estimated ECEF position variances in the experiment. Specifically, variances are shown for GLONASS vector tracking, GPS vector tracking, GPS & GLONASS scalar tracking, and GPS & GLONASS vector tracking. All of the estimated variances are approximately the same for each configuation other than GLONASS. As in the light tree foliage experiment, the GLONASS vector tracking variances are much larger. This may be from poor altitude positioning as in the light tree foliage experiment. However, the GLONASS position variances drop at the end of the experiment when the receiver is on the bridge in the open sky environment. The quality of satellite geometry and signal strength between GPS and GLONASS is similar as was shown in Table 6.3 and Figure 6.34. The estimated position variances suggest that GLONASS suffers more from multipath than GPS, which may be from its use of FDMA for satellite identification.

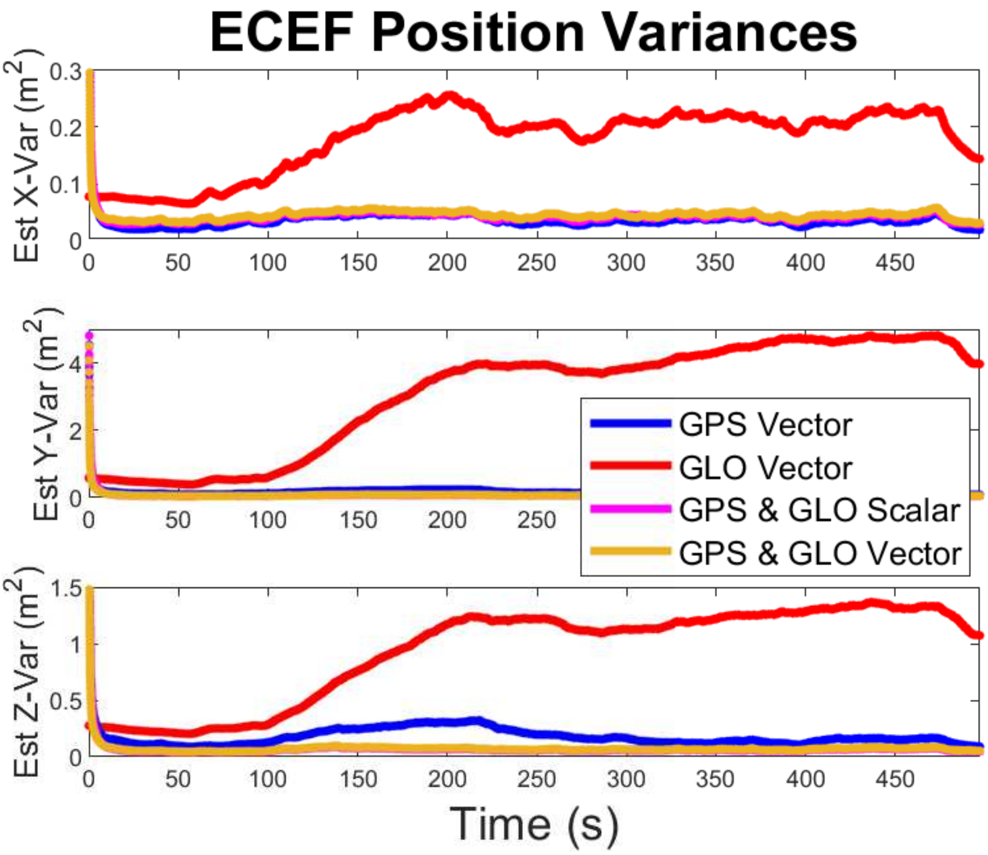

Figure 6.37: Estimated Position Variances in the Heavy Tree Foliage Experiment

Figure 6.38 shows positioning results when the software receiver performs GLONASS vector tracking, GPS vector tracking, GPS & GLONASS scalar, and GPS & GLONASS vector tracking. In these results, the software receiver starts at the interstate bridge and moves through the forest, back to its original location. Although the bridge is open to an overhead clear sky, the northwest and southeast sides of the antenna still receive attenuated satellite signals due to the forestry. The software receiver is compared to the Ublox when using only GPS. As seen in the figure, the Ublox receiver is less accurate when it only uses GPS, unable to maintain a position on the road at all times. However, the Ublox still outperforms the software receiver due to its use of WAAS and the possiblity that the Ublox uses advanced filtering techniques to remove multipath. Overall, the GPS vector tracking configuration performs the best in this experimental run.

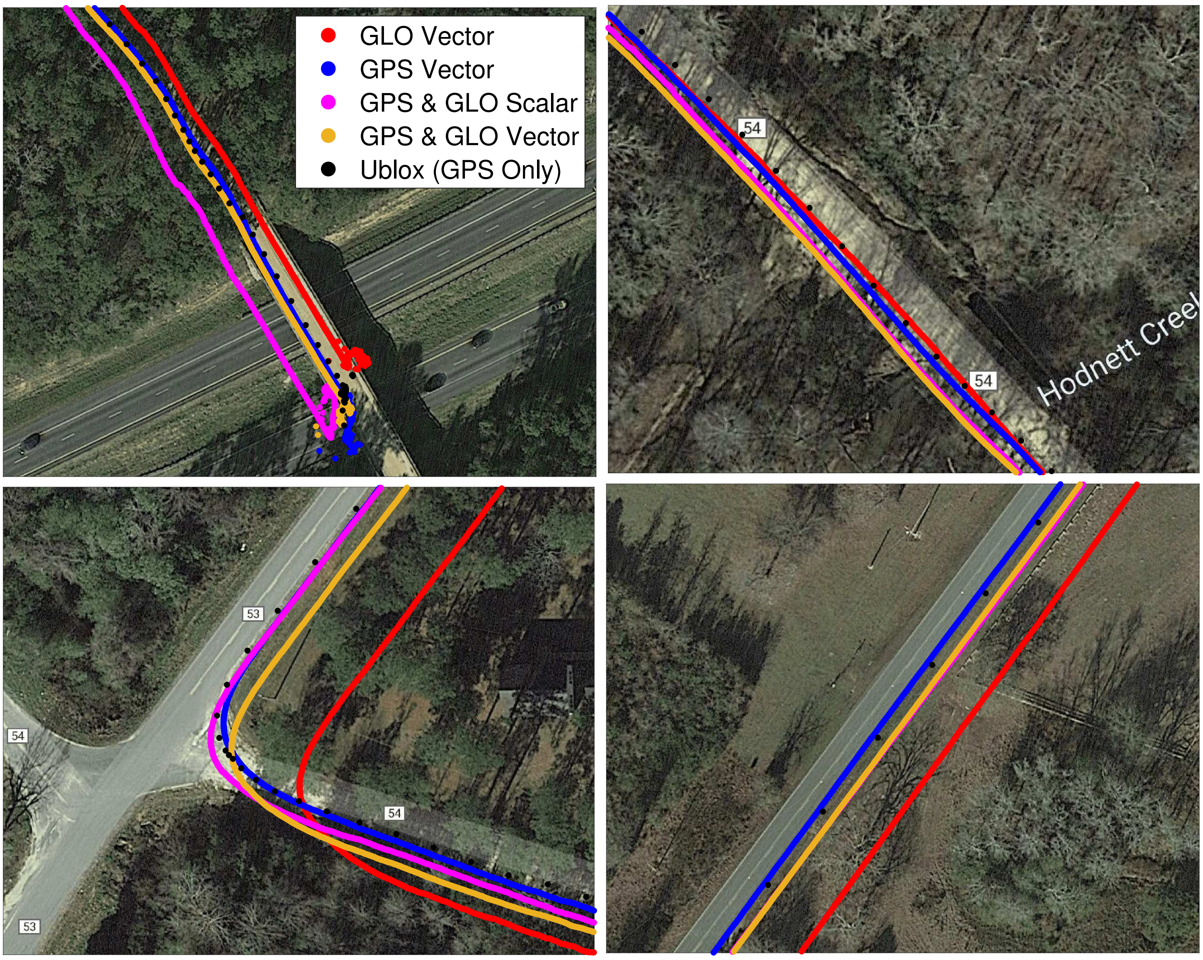

Figure 6.38: Positioning Results near Tuskegee National Forest when Starting at the Interstate Bridge

Figure 6.39 shows the carrier phase discriminators for GPS PRN #23 and GLONASS K #6 in the heavy tree foliage experiment. The discriminators are shown when the software receiver uses scalar tracking, single constellation (GPS or GLONASS) vector tracking, and dual constellation (GPS and GLONASS) vector tracking. The discriminators are approximately the same for each tracking architecture because they all use PLL aiding, which performs the majority of the signal processing. The GPS and GLONASS discriminator magnitudes go above 50 degrees often, because the multipath from the tree foliage distorts the satellite signals' carrier phases. When the receiver arrives to an open sky environment at the end of the experiment, the GPS and GLONASS carrier phase discriminators drop in magnitude, indicating carrier phase lock.

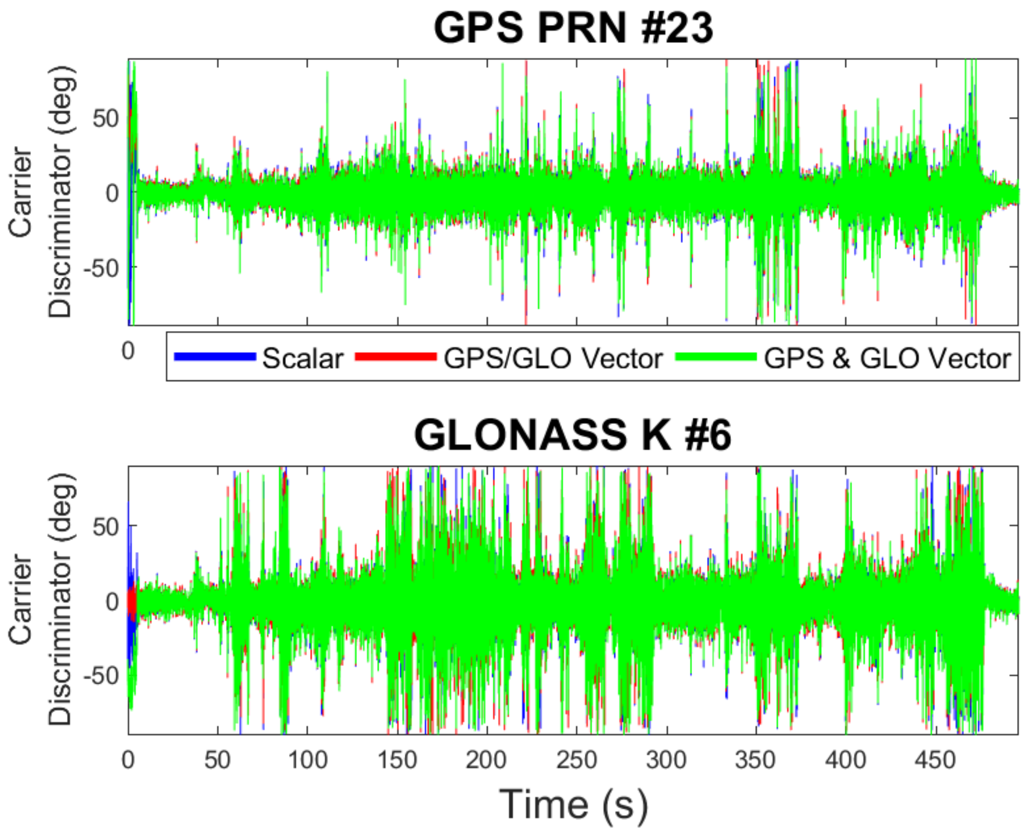

Figure 6.39: GPS and GLONASS Carrier Phase Discriminators in Heavy Tree Foliage

6.3.3 Parking Deck Results

Signal data was collected at a parking deck attached to the Valley West Mall in West Des Moines, Iowa. The purpose of this collection was to test the software receiver against low C/N_O satellite signals. In the route, the vehicle drives under two pedestrian walk bridges and enters a two-level parking deck. The parking deck roof attenuates satellite broadcasts overhead of the vehicle and allows low elevation angle signals to enter the antenna from the sides of the parking deck. Note that low elevation satellite broadcasts will also have low C/N_O ratios. A turn is performed inside the parking deck to ensure the software receiver cannot solely rely on its state model. Figure 6.40 shows pictures of the trajectory with arrows and numbers indicating the vehicle's path. The pedestrian walk bridges have been highlighted by circles.

Figure 6.40: Pictures of Parking Deck Trajectory in West Des Moines, IA

The software receiver operated with 9 GPS satellites and 10 GLONASS satellites. Figure 6.41 shows a sky plot of the GPS and GLONASS satellites used in the experiment. Table 6.4 provides satellite geometry DOP values for the experiment.

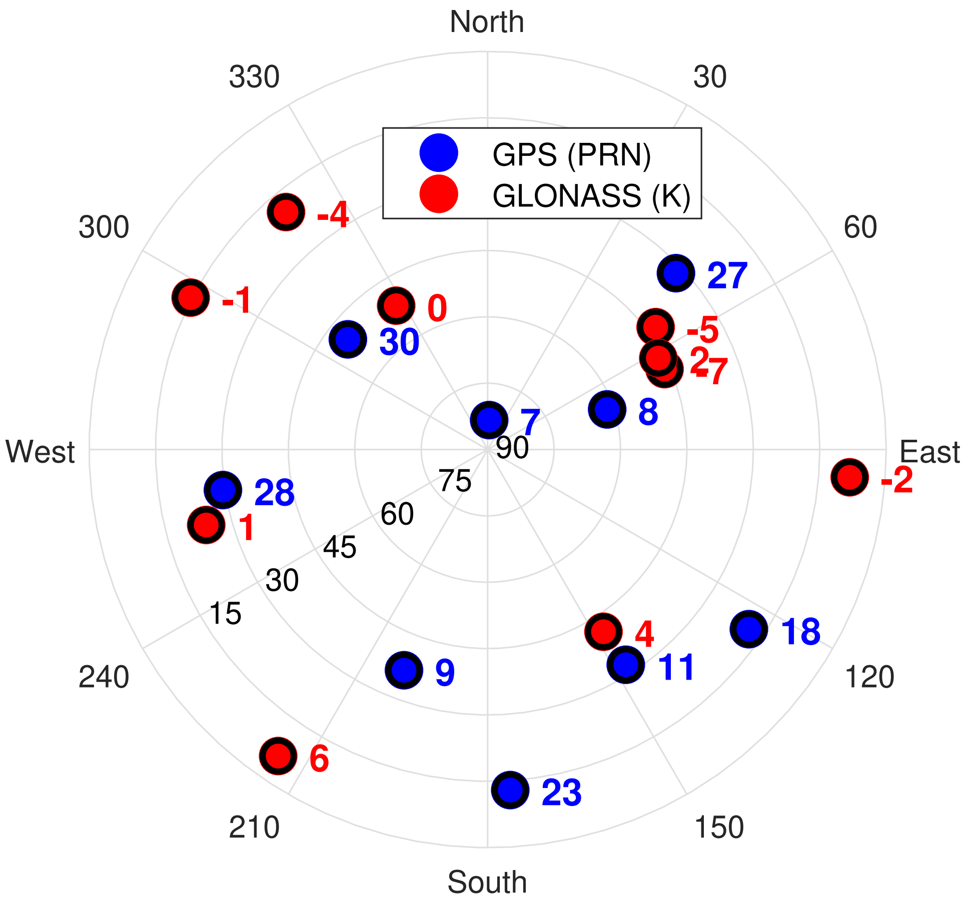

Figure 6.41: Sky Plot of GPS and GLONASS Satellites used in West Des Moines, IA

Table 6.4: DOP Values for the Parking Deck Experiment

Constellation	**PDOP**	**HDOP**	**VDOP**	**TDOP**
GPS	2.0681	1.0322	1.7920	1.2232
GLONASS	1.7015	0.8632	1.4663	0.7499
GPS & GLONASS	1.1788	0.6088	1.0094	0.5903

Figure 6.42 shows positioning results with the GPS & GLONASS software receiver when using scalar and vector tracking in the parking deck test. The software receiver is compared to the Ublox that maintains an accurate position throughout the experiment. Both software configurations maintain accurate position estimates. However, when the software receiver is in vector mode, the position estimate agrees more with the Ublox estimate.

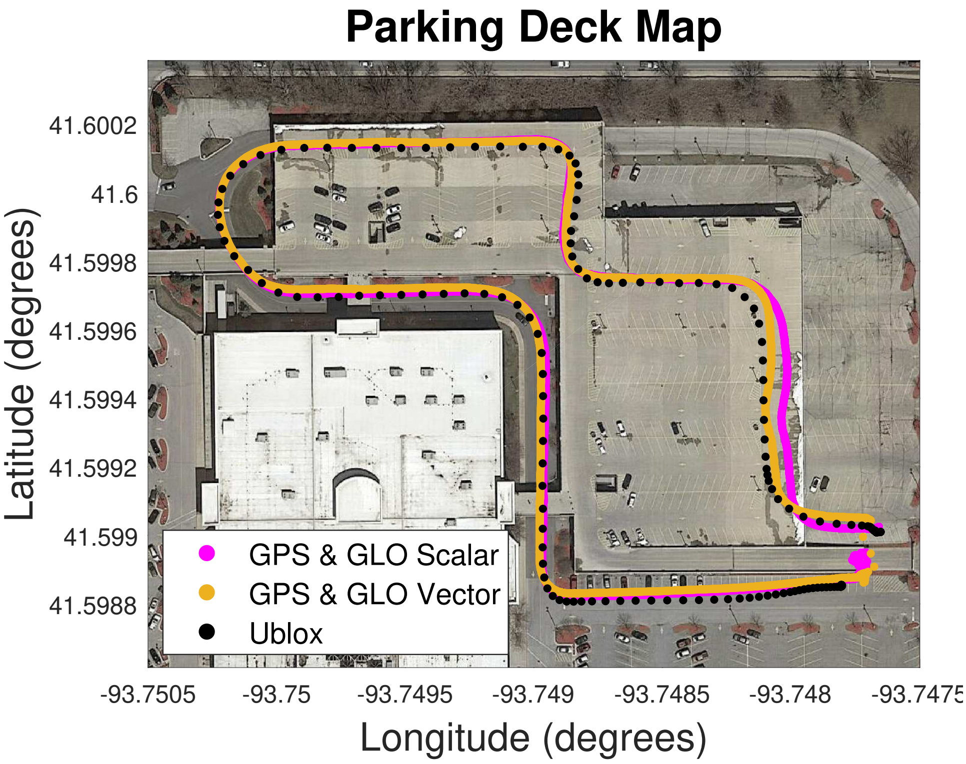

Figure 6.42: GPS & GLONASS Positioning Results in West Des Moines, IA

Figure 6.43 shows the receiver estimated C/N_O ratios for the GPS and GLONASS channels in the parking deck experiment. In this test, the C/N_O ratios drop instead of smearing, as they did in the tree foliage experiments. The pedestrian walk bridges and the parking deck attenuate the satellite signals when the receiver antenna is underneath them. Once the receiver is back in a clear sky environment at the end of the test, the channels' C/N_O ratios return back to normal levels. This indicates that the GPS & GLONASS software receiver can successfully track the satellite signals while traveling under the parking deck.

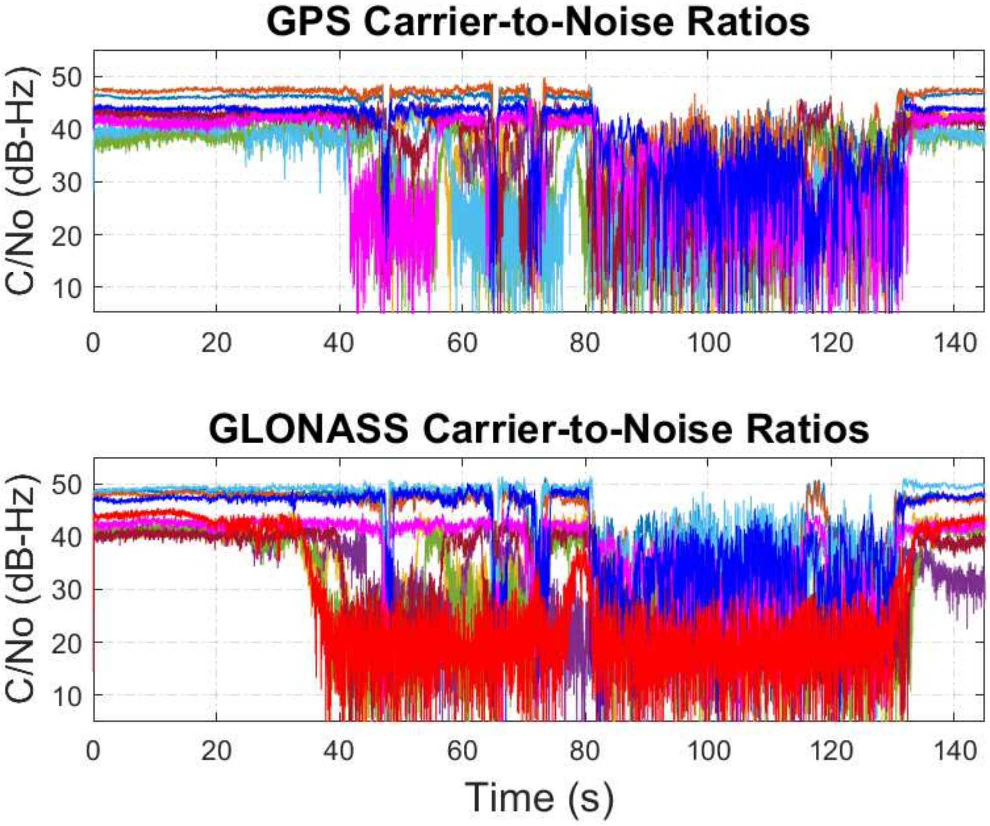

Figure 6.43: GPS and GLONASS C/N_O Ratios in the Parking Deck Experiment

Figure 6.44 shows positioning results when the software receiver performs scalar tracking with GPS, GLONASS, and both constellations. The receiver estimate is most accurate when using both GPS and GLONASS, following the Ublox's trajectory well. The GPS scalar tracking results are also accurate, but the position estimate is noisier under the parking deck due to signal attenuation. A bias is observed on the GLONASS position, which may come from coordinate transformation error. GLONASS may also suffer from multipath reflections induced by the mall building and parking deck.

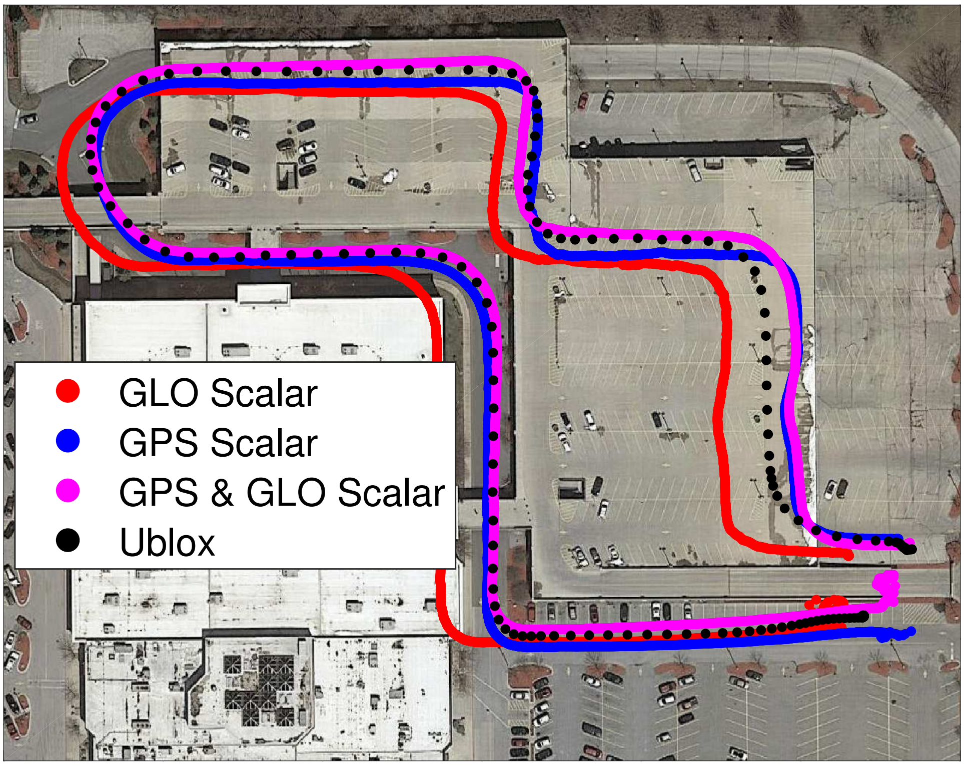

Figure 6.44: Parking Deck Positioning Results with Scalar Tracking

Figure 6.45 shows positioning results for the configurations when the software receiver uses vector tracking. All the configurations are improved by transitioning from scalar to vector tracking, in comparison to the Ublox trajectory. The biases in the GLONASS positions are reduced when using vector processing. The multipath errors induced from reflections off the mall building and parking deck were mitigated with vector processing. Clear sky GLONASS channels helped mitigate multipath errors in channels affected by reflections.

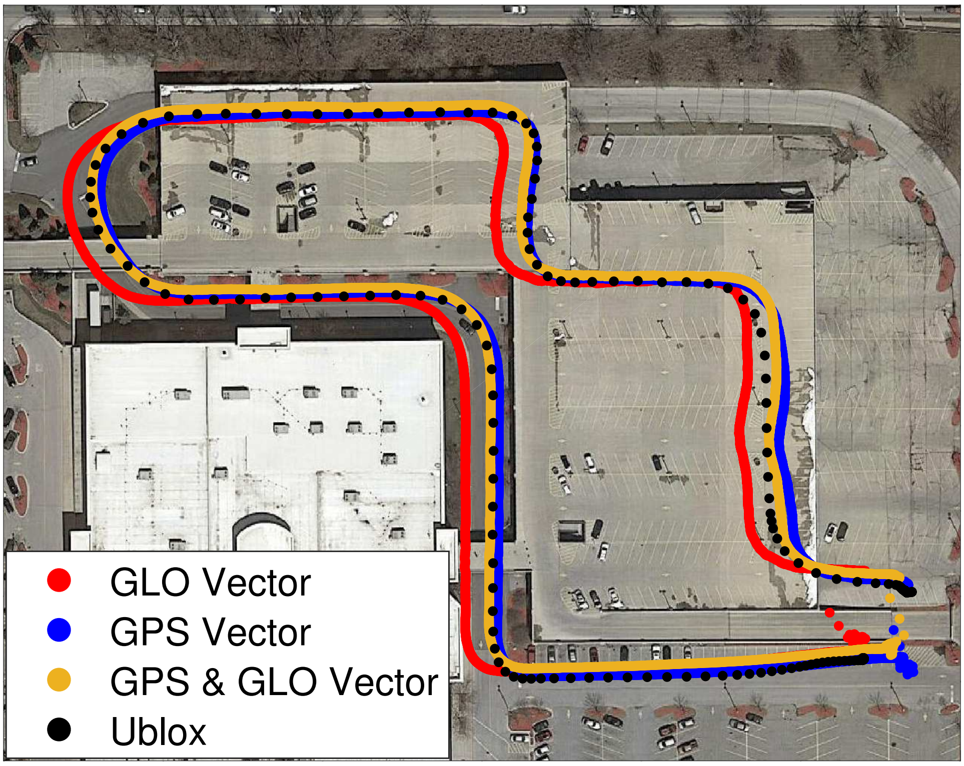

Figure 6.45: Parking Deck Positioning Results with Vector Tracking

Figure 6.46 show the Kalman filter estimated ECEF position variances when the software receiver uses GLONASS vector tracking, GPS vector tracking, GPS & GLONASS scalar tracking, and GPS & GLONASS vector tracking. The variances remain stable for each receiver configuration. Positioning with both GPS and GLONASS lowers the variances, with the vector filter maintaining lower magnitudes than scalar tracking. The variances increase where the channels' C/N_O ratios dropped in Figure 6.43. When the vehicle antenna goes under the parking deck, the receiver architectures converge to new, higher variances due to satellite signal attenuation. Signal attenuation increases noise on the residual measurements, which causes a noisier position estimation and a higher estimated position variance. After the vehicle exits the parking deck, the architectures transition back to the original variances they had at the start of the test.

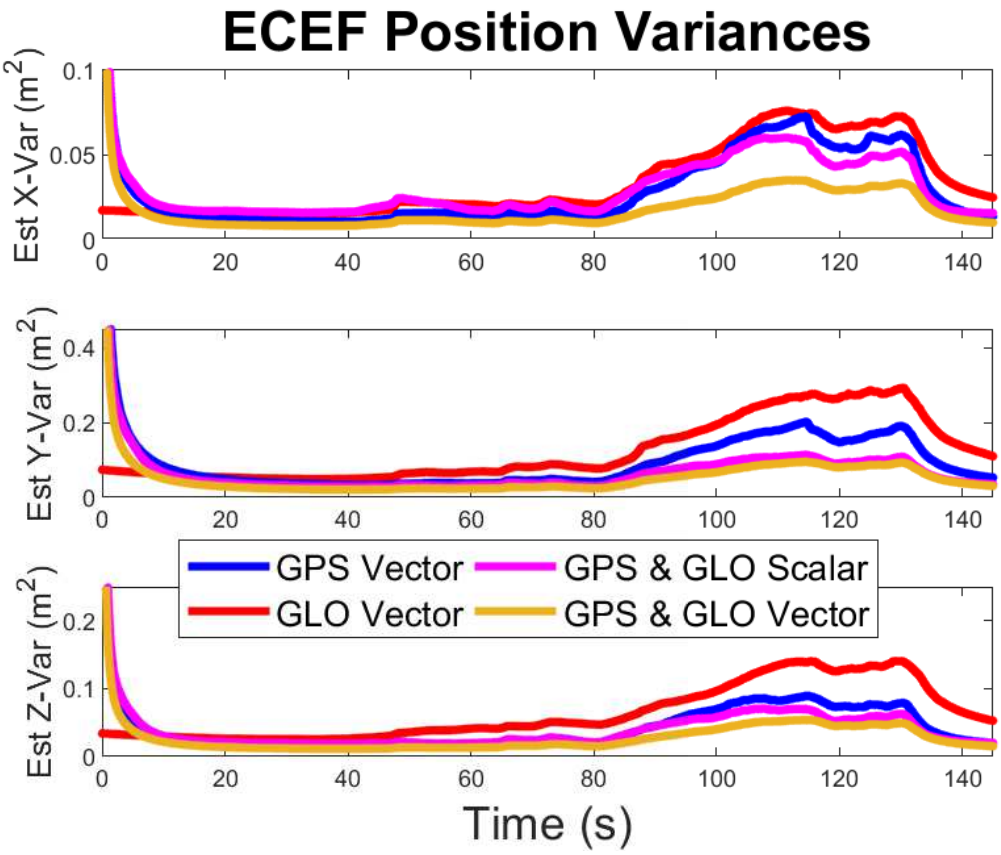

Figure 6.46: Estimated Position Variances in the Parking Deck Experiment

Figure 6.47 shows the code phase discriminators for GPS PRN #30 and GLONASS K #-4. The discriminators are shown when the software receiver performs scalar tracking, single constellation (GPS or GLONASS) vector tracking, and dual constellation (GPS and GLONASS) vector tracking. The channels' discriminators become noisy from signal attenuation when the vehicle passes under the pedestrian bridges and the parking deck roof. When scalar tracking is performed, the code phase discriminators are zero mean for both channels. Scalar tracking accurately tracks the PRN codes with the discriminator feedback, resulting in the zero mean errors.

When vector tracking is performed, the discriminators appear to be biased (not zero mean), this results from poor atmospheric delay tracking in the algorithm. Vector processing maintains track of a channel's PRN code with the receiver position and clock bias estimation. When the receiver position and clock bias solution is accurate, estimation error from atmospheric delays is attenuated. To accurately track the PRN code, the atmospheric delay must also be tracked.

Scalar processing successfully tracks the atmospheric delay with its feedback loop. Vector processing attenuates the atmospheric error in the position and time solution, resulting in the delays not properly being tracked when the navigation solution is fed back to the channels' replicas.

For the GPS channel, single constellation vector tracking gives the discriminator a negative bias, and dual constellation vector tracking negatively biases the discriminator further. This indicates that the atmospheric delay is not properly tracked in the channel's code replica with GPS vector tracking. The atmospheric delay tracking becomes worse with dual constellation vector tracking. It is also a strong indicator that the GPS & GLONASS position estimates are more accurate than the GPS position estimates, since the atmospheric error is filtered more in the navigation solution when using both constellations in the software receiver.

In the GLONASS channel, single constellation vector tracking gives the discriminator a positive bias, and dual constellation vector tracking positively biases the discriminator further. Like the GPS channel, vector tracking does not properly track the atmospheric delay and becomes worse when both constellations are used in the navigation processor.

When GPS and GLONASS are both used in the vector tracking algorithm, the code phase discriminators have larger biases because the receiver position solution has become more accurate. This is justified in Figure 6.45, assuming the Ublox trajectory is accurate. The increase in position accuracy comes at the cost of poor atmospheric delay tracking in the channels' code replicas. In other words, the vector algorithm enhanced position accuracy, but at the cost of degraded code tracking. Note that the code phase discriminators' biases from poor atmospheric delay tracking will not become significantly large, as the atmospheric bias plays a small role in the PRN code dynamics. If the discriminator bias becomes too large, approximately greater than 0.1 chip, this indicates the vector filter is estimating an inaccurate position.

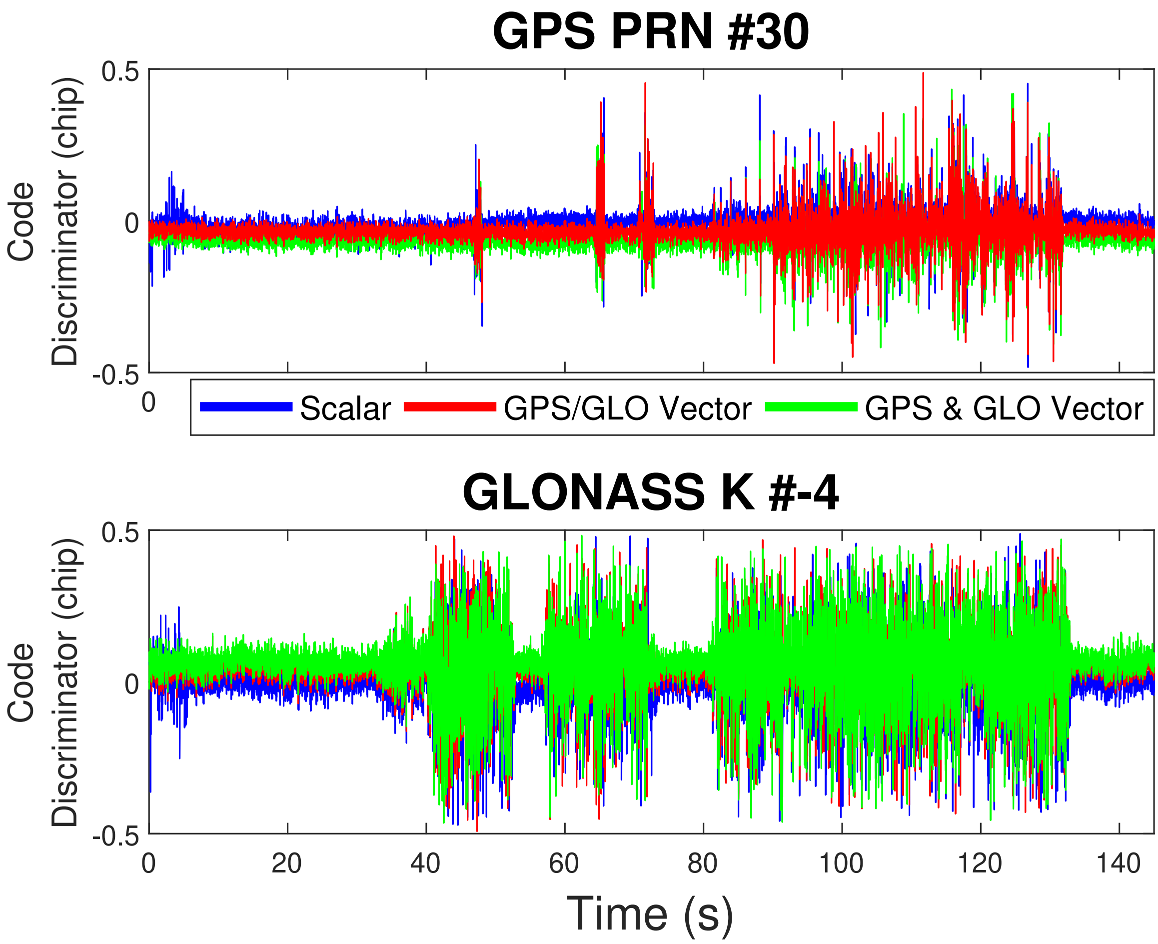

Figure 6.47: GPS and GLONASS Code Phase Discriminators in the Parking Deck Experiment

6.3.4 Urban Canyon Results

The final testing location of the software receiver was downtown Des Moines, Iowa, which serves as an urban canyon environment. Urban canyons are typically challenging environments for GNSS receivers. The large buildings and skyscrapers can block and reflect satellite signals, inducing attenuation and multipath errors in the receiver. This downtown area also has a skywalk network that attenuates signals overhead of the vehicle antenna. Figure 6.48 shows pictures of downtown Des Moines where the test vehicle drove.

Figure 6.48: Pictures of Downtown Des Moines, IA

The software receiver operated with 8 GPS satellites and 8 GLONASS satellites. Figure 6.49 shows a sky plot of the GPS and GLONASS satellites used in the experiment. Table 6.5 provides satellite geometry DOP values for the experiment.

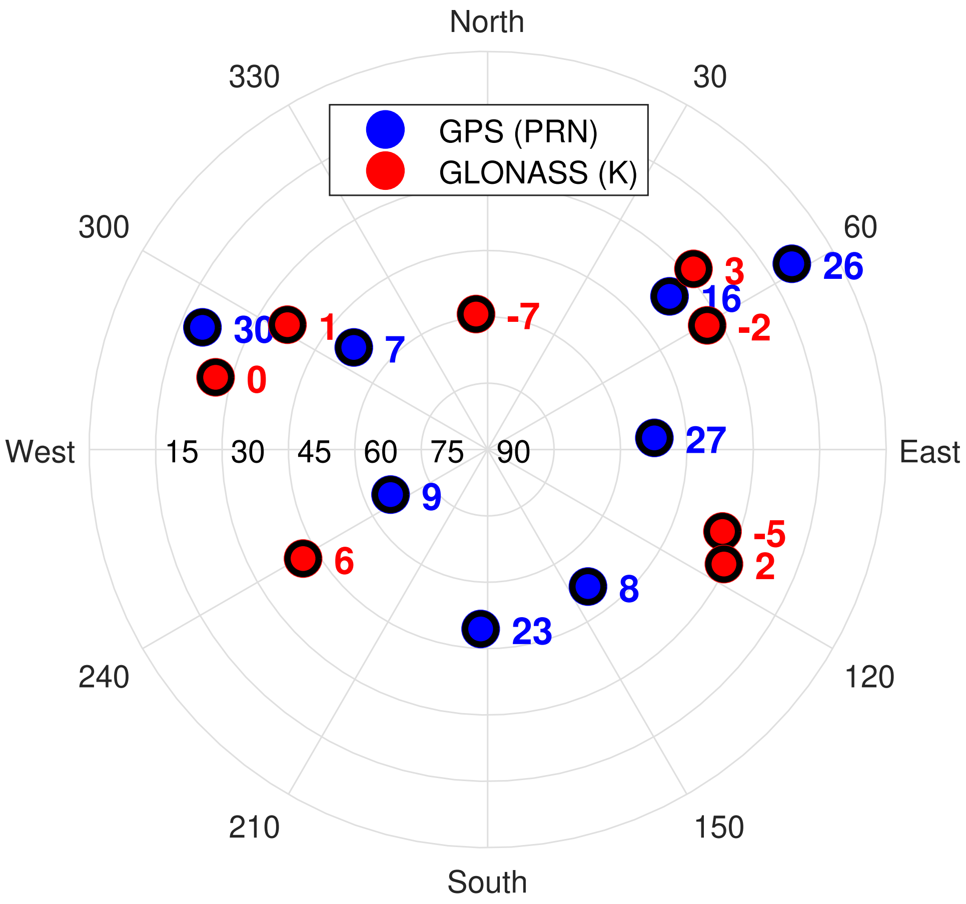

Figure 6.49: Sky Plot of GPS and GLONASS Satellites used in Downtown Des Moines, IA

Table 6.5: DOP Values for the Urban Canyon Experiment

Constellation	**PDOP**	**HDOP**	**VDOP**	**TDOP**
GPS	2.2320	1.2139	1.8730	1.3017
GLONASS	3.1609	1.0591	2.9782	1.7766
GPS & GLONASS	1.6220	0.7757	1.4245	0.9423

Figure 6.50 shows positioning results for the GPS & GLONASS software receiver in the urban canyon experiment. The results are compared to the Ublox receiver. Overall, each receiver performed accurate, stable positioning. The receivers, including the Ublox, did struggle to maintain position estimates on the road where the urban canyon environment became severe. The software receiver matched the Ublox estimate more closely when using vector tracking over scalar tracking.

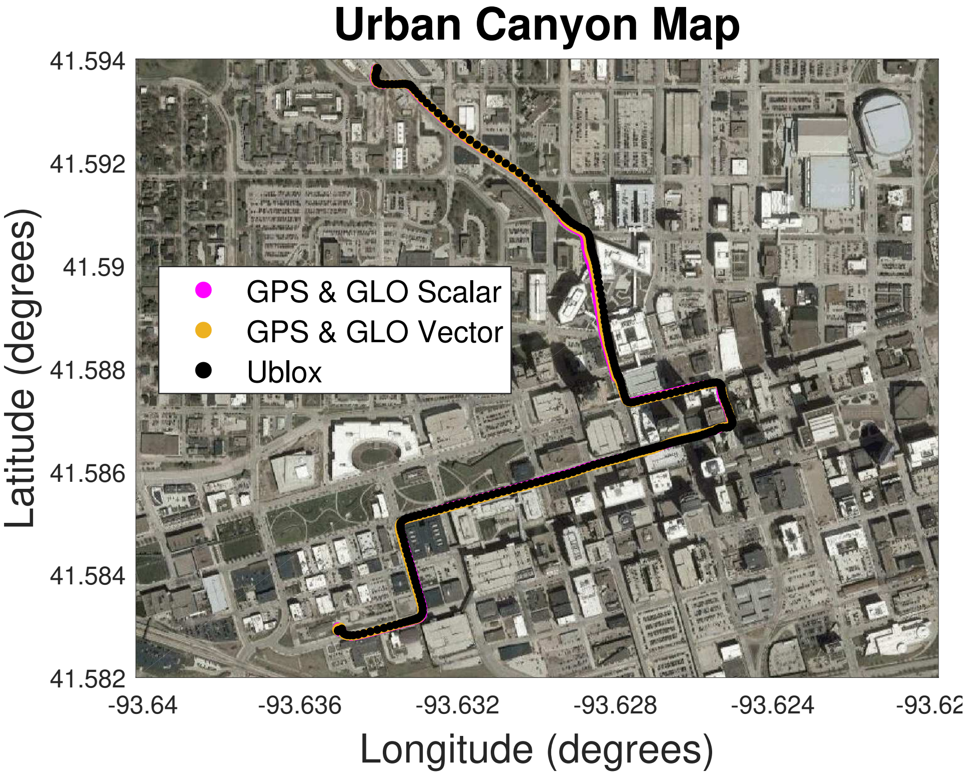

Figure 6.50: GPS & GLONASS Positioning Results in Downtown Des Moines, IA

Figure 6.51 shows the receiver estimated C/N_O ratios for the GPS and GLONASS channels in the experiment. The channels' C/N_O ratios are either attenuated, smeared, or a combination of both while in the urban canyon environment. Attenuation comes from buildings blocking satellite signals, and smearing comes from signal reflections off the buildings entering the vehicle antenna. It is shown in the figure that all the channels' C/N_O ratios returned to their normal values after exiting the urban canyon environment. This indicates that the GPS & GLONASS vector tracking software receiver was able to maintain accurate tracking estimations while in the environment.

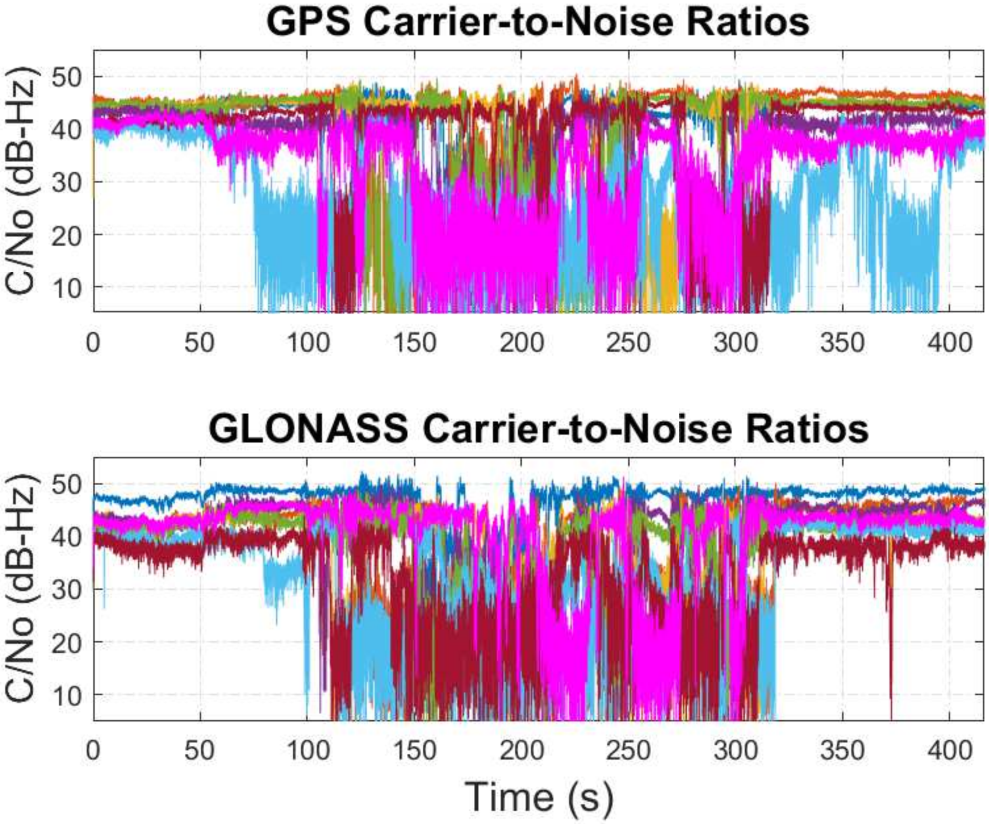

Figure 6.51: GPS and GLONASS C/N_O Ratios in the Urban Canyon Experiment

Figure 6.52 shows positioning results when the software receiver performs scalar tracking with GPS, GLONASS, and both constellations. GLONASS scalar tracking becomes unstable in the urban canyon, resulting in poor position estimates. The GPS and GPS & GLONASS configurations are able to maintain stable navigation results. Overall, GPS & GLONASS outperformed GPS in terms of position accuracy.

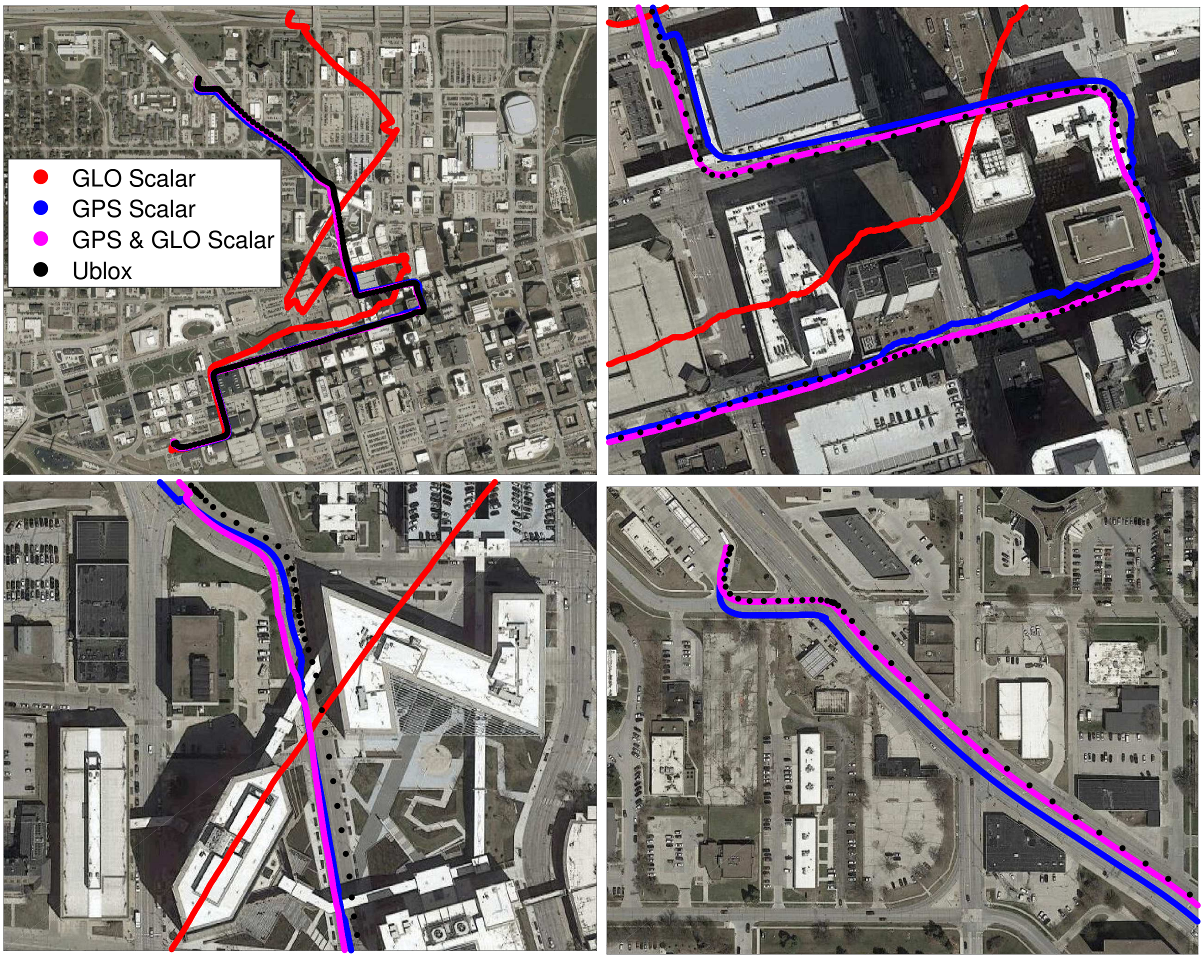

Figure 6.52: Urban Canyon Positioning Results with Scalar Tracking

Figure 6.53 shows position results when the software receiver performs vector tracking with GPS, GLONASS, and both constellations. Each of the receiver configurations are improved by transitioning from scalar to vector processing. The GLONASS results are no longer unstable, but positioning is still poor. At the most severe location of the urban canyon, GPS & GLONASS vector tracking does not provide significant improvement over GPS vector tracking. In this case, because GLONASS performs so poorly, adding GLONASS into the vector algorithm degrades positioning results at some locations in the test.

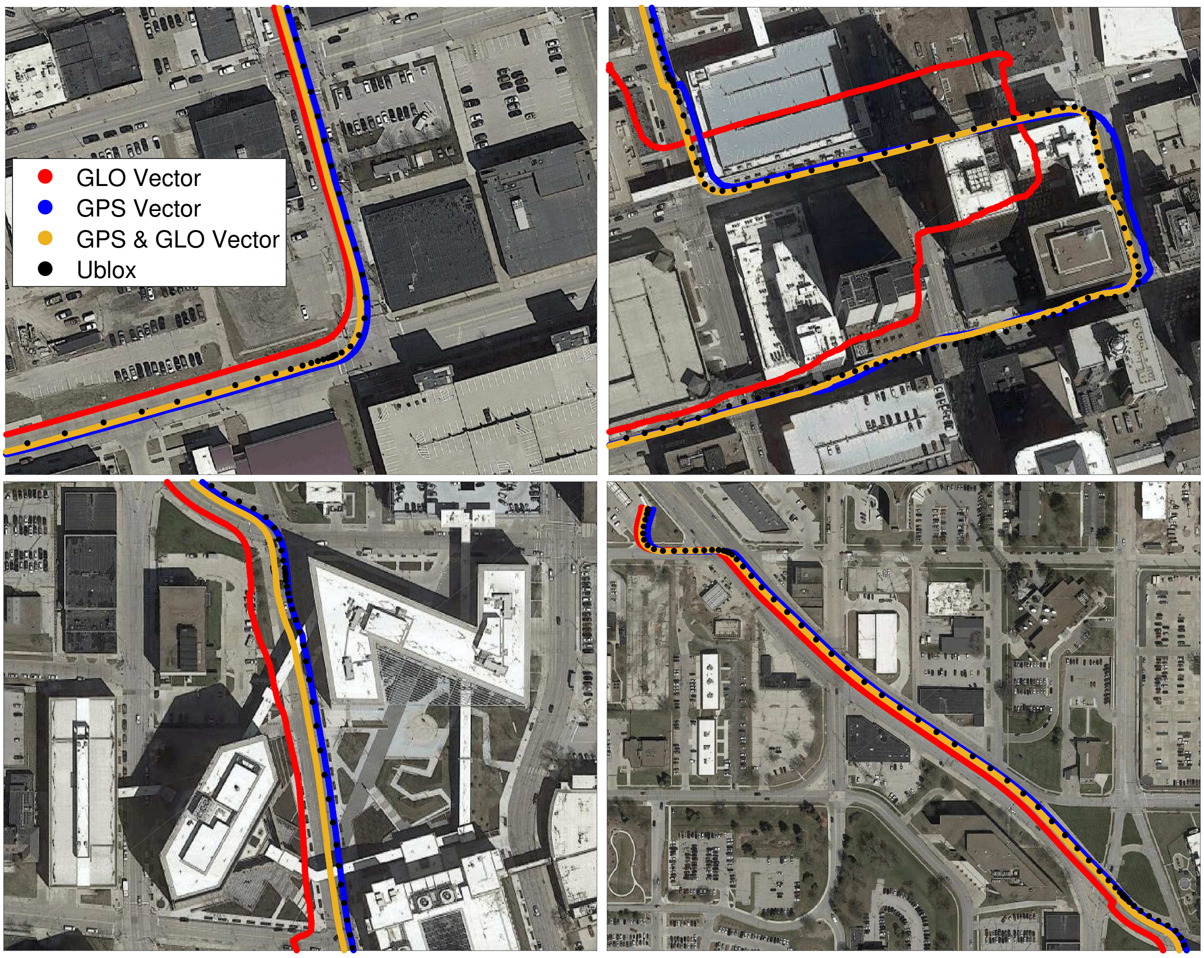

Figure 6.53: Urban Canyon Positioning Results with Vector Tracking

Figures 6.54 through 6.57 show additional positioning results for the vector receivers and Ublox in the urban canyon experiment. Note that these positioning results do not include altitude information, only latitude and longitude. In Figure 6.56, the vehicle was traveling in the left lane, next to the parking lane and the building. In Figure 6.57, the vehicle was traveling in the right most lane, next to the line of parked cars. In the urban canyon positioning figures, **GPS vector tracking** is shown in blue, **GLONASS vector tracking** is shown in red, **GPS & GLONASS scalar tracking** is shown in magenta, **GPS & GLONASS vector tracking** is shown in yellow, and **Ublox** is shown in black.

Figure 6.54: Positioning Results for the Urban Canyon Experiment in Google Earth

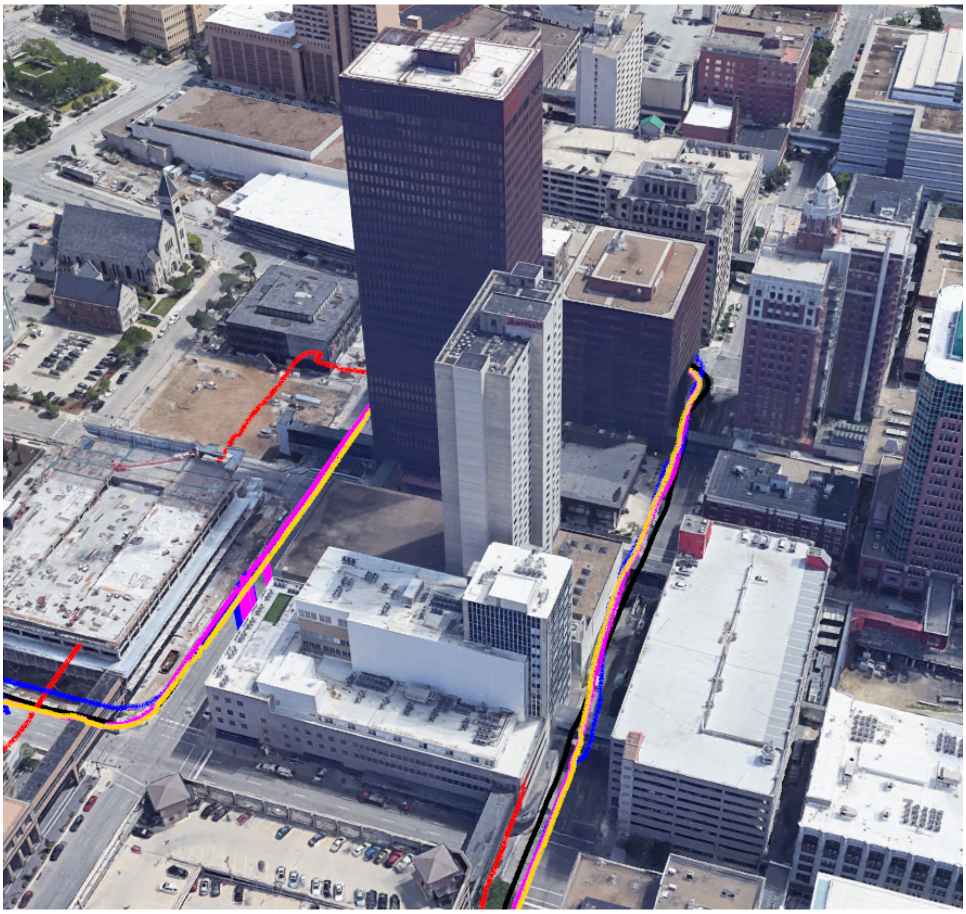

Figure 6.55: Positioning Results for the Severe Urban Environment in Google Earth

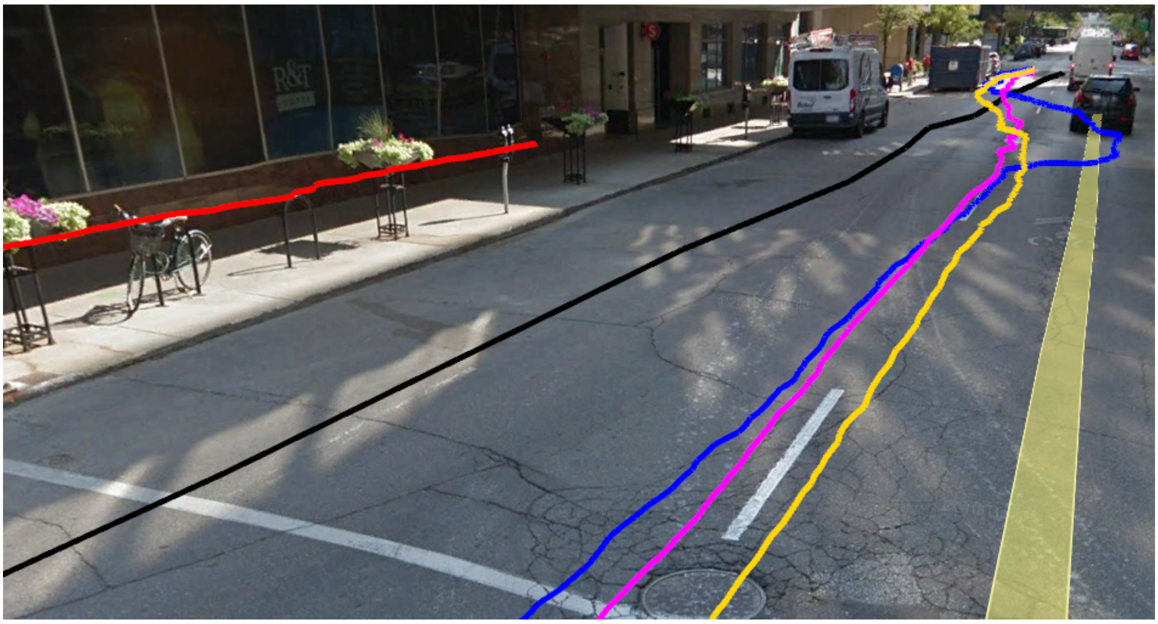

Figure 6.56: Google Earth Pedestrian View of Positioning Results in Downtown Des Moines

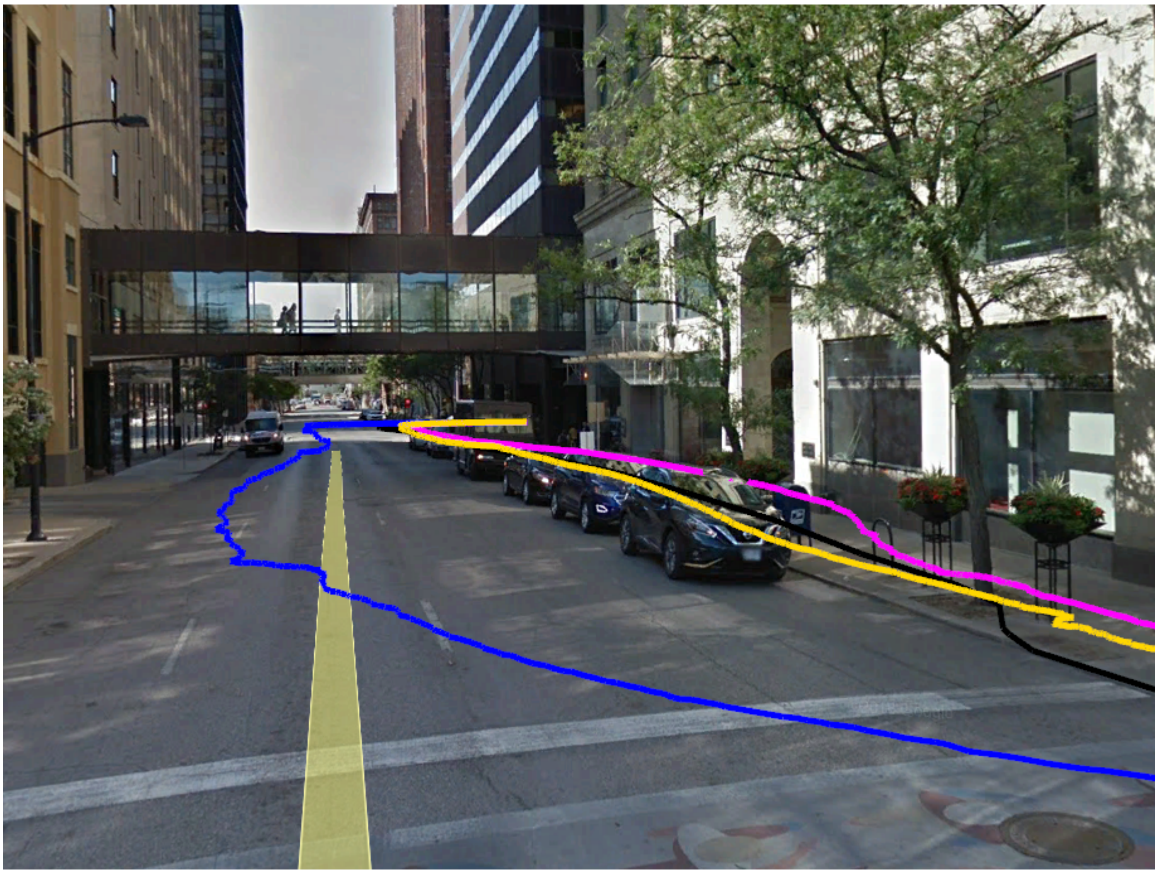

Figure 6.57: Google Earth Pedestrian View of Postioning Results in Severe Urban Environment

Figure 6.58 shows the Kalman filter estimated ECEF position variances for GPS vector tracking, GLONASS vector tracking, GPS & GLONASS scalar tracking, and GPS & GLONASS vector tracking. Each tracking architecture maintains low variances at the beginning and end of the experiment. The variances spike at the middle of the test when the vehicle antenna is in the most severe urban canyon area. This location is also associated to where the GPS and GLONASS channels' C/N_O ratios significantly dropped. The variance spikes are larger for the single constellation architectures than the dual constellation architectures. GPS & GLONASS has approximately the same variances for both scalar and vector tracking.

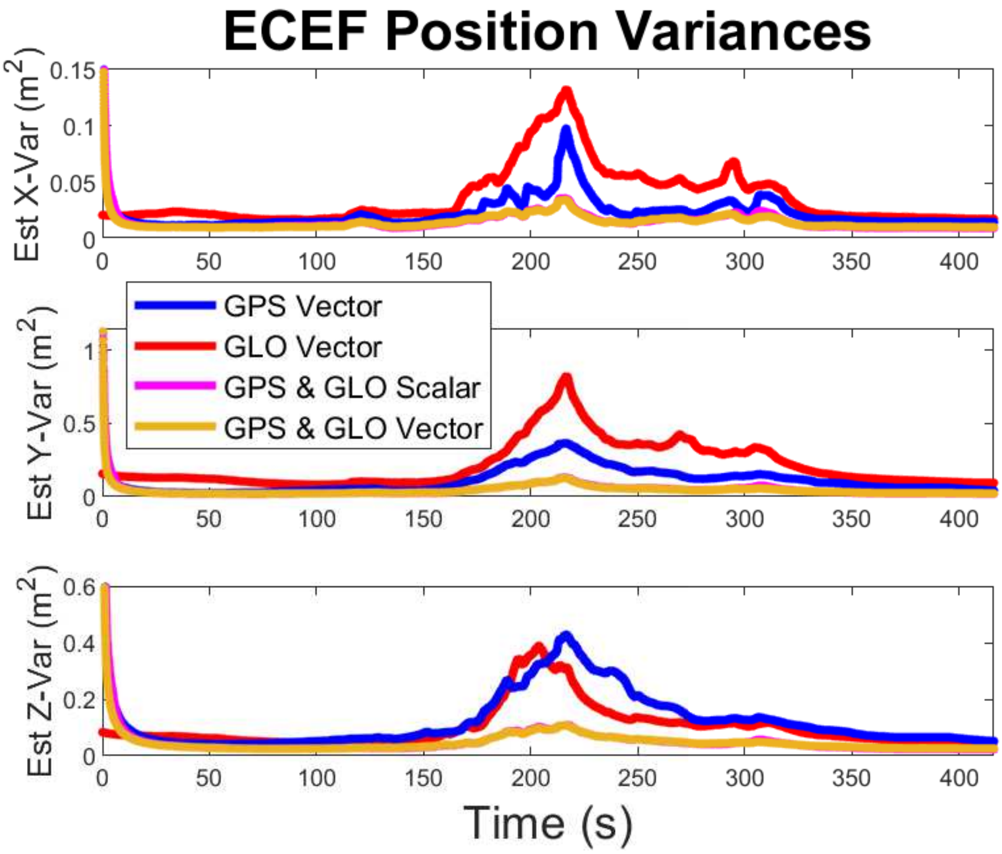

Figure 6.58: Estimated Position Variances in the Urban Canyon Experiment

Figure 6.59 shows estimated PRN code Dopplers for GPS PRN #7 and GLONASS K #-5 in the urban canyon experiment. The results are shown for scalar tracking, single constellation (GPS or GLONASS) vector tracking, and dual constellation (GPS and GLONASS) vector tracking. The scalar frequencies for both channels became noisy in locations where the channels are blocked by buildings. This induced noise is significantly reduced when using the vector tracking architectures. Overall, the vector tracking frequencies are much more precise due to tightly tuned Kalman filters in the vector receivers. Dual constellation vector tracking attenuates noise slightly more than single constellation vector tracking. This occurs because the GPS & GLONASS receiver maintains a more precise navigation solution than the receivers that only use GPS or GLONASS.

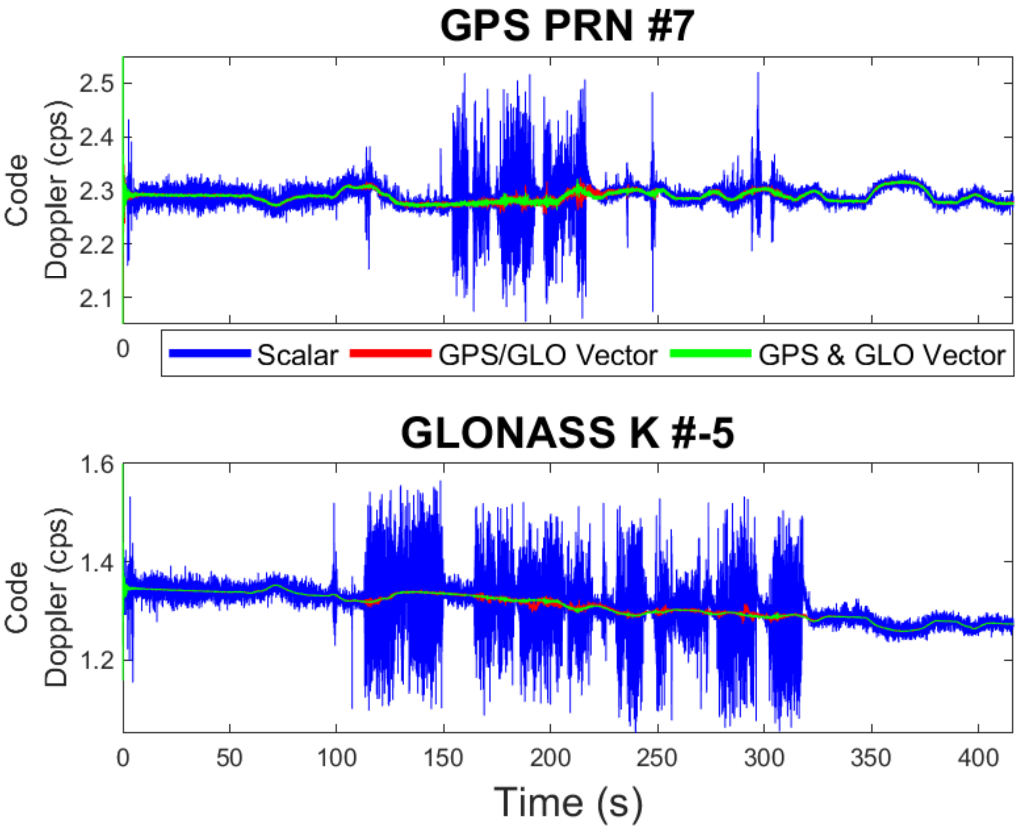

Figure 6.59: GPS and GLONASS Code Frequencies in the Urban Canyon Experiment

6.4 Conclusion

This chapter discussed the implementation and performance analysis of GPS & GLONASS vector tracking. Implementing the constellations together into a single navigation estimator requires consideration of time and synchronization parameters. It was chosen to estimate only one clock bias for the GPS and GLONASS constellations. Measurement synchronization techniques of GPS and GLONASS measurements in the vector tracking filter were discussed. Navigation with common transmission time measurements was selected over common reception time measurements for stability purposes. A coordinate transformation between the WGS84 frame used by GPS and the PZ-90.11 frame used by GLONASS was developed empirically from static data sets. The coordinate transformation requires a translation in the ECEF x-direction.

Data sets in this chapter showed that navigating with GPS & GLONASS improved positioning results in good signal environments. An improvement in altitude position by using both constellations was also discovered. In general, vector tracking can bias code phase discriminators due to poor tracking of atmospheric delays in the receiver signals' PRN codes. The biases observed on the code phase discriminators can become more significant when performing vector tracking with GPS and GLONASS due to additional filtering of the atmospheric errors in the receiver position and time solution.

Simulated jamming tests proved the software receiver's ability to have the satellite constellations aid each other through vector tracking. When GPS was jammed, the GLONASS channels in the vector software receiver were able to maintain accurate tracking estimates of the GPS channels. When GLONASS was jammed, the GPS channels in the vector software receiver were able to maintain accurate tracking estimates of the GLONASS channels. The simulated jamming tests also showed that dual constellation vector processing outperformed dual constellation scalar processing.

The chapter concluded with software receiver results in challenging signal environments. The software receiver was compared to a Ublox receiver that also uses GPS and GLONASS. The Ublox outperformed the software receiver overall, however, this was due to the Ublox receiver's use of WAAS. In the challenging signal environments, the software receiver performed better with vector tracking than scalar tracking. GPS results outperformed GLONASS results in each test. The experimental data suggests GLONASS suffers more error from multipath than GPS. Depending on the test, using both GPS and GLONASS in the vector tracking algorithm improved or degraded results with respect to GPS vector tracking. If the GLONASS solution was significantly poorer than the GPS solution, the GPS solution would outperform the GPS & GLONASS solution.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The development of the GPS & GLONASS L1 vector tracking software-defined receiver posed interesting technical challenges. The discussion of single constellation vector tracking in the thesis was straightforward as a significant amount of work has been published on the topic. Specific, but important details of the VDFLL algorithm were considered that have not been discussed thoroughly in the navigation community. The inclusion of scalar measurement residuals with the tracking discriminators in the VDFLL was analyzed. Results showed that including the scalar pseudorange residual into the VDLL enhanced the positioning accuracy by correcting quantization effects from the receiver front-end sampling frequency. Including the scalar pseudorange rate residual is not necessary in the VFLL unless carrier phase aiding from a PLL is also inputted into the NCO command. Different types of VDLL code NCO calculations were analyzed and shown to be approximately the same, having only small variations in code frequency between them.

The issue of discriminator measurement synchronization in the VDFLL was also addressed. Vector tracking measurements can be inputted into a batch estimator (all channels at once) on a common transmission time or a recursive (one channel at a time) estimator on a common reception time. It is preferred to navigate on a common reception time to ensure the PVT solution is associated to one specific time. Results showed that the vector tracking filter was more stable with common transmission time measurements since the coupling effect of the channels is stronger in a batch estimator. The common transmission time measurements also provided more accurate position estimates with respect to the Ublox receiver. Using a well-tuned Kalman filter, precision is approximately the same for common transmission and common reception time measurements. It was chosen to have the software-receiver perform vector processing with common transmission time discriminator measurments, at the expense of an ambiguous time of navigation.

The software receiver in this thesis processes GPS and GLONASS signal data from the same clock. Since both constellations are processed by the same clock at the receiver, they will both have the same clock drift error. However, because the constellations operate with different time systems, their clock bias errors will not be the same. Experimental results with different oscillator qualities showed that the clock bias difference between the satellite systems is approximately a small, constant time offset when using the front-end's internal TCXO clock. This was not the case for tested OCXO, rubidium, and cesium atomic clocks. Using the TCXO with the software receiver, it was assumed the time offset could be ignored, and one clock bias was estimated for both constellations. A discussion was also given on transforming between GPS and GLONASS time, however, it was not necessary for the software receiver to do this in any of the experiments. GPS measurements stayed on GPS time and GLONASS measurements stayed on GLONASS time in the software receiver.

The ECEF coordinate transformation between WGS84, used by GPS, and PZ-90.11, used by GLONASS, was also investigated. A coordinate transformation between the frames was developed empirically from data taken in the Auburn, Alabama and Des Moines, Iowa areas. The transformation consists of a translation in the ECEF x-direction. Much more data would be required to develop an official transformation between the coordinate frames. In the GPS and GLONASS navigator, the GLONASS satellite positions are transformed from PZ-90.11 to WGS84 for positioning in the GPS coordinate frame. Using the coordinate transformation, GPS and GLONASS static positioning outperformed GPS static positioning and GLONASS static positioning. Results showed that altitude estimation was improved by navigating with both constellations.

Performance analysis of the GPS & GLONASS L1 vector tracking software-defined receiver gave unique insights on vector processing across satellite systems. Simulated jamming results proved the capabilities of GPS and GLONASS to aid one another through vectorized signal tracking. In the GPS jamming test, 6 GLONASS channels were able to maintain accurate tracking replicas of 8 GPS channels. In the GLONASS jamming test, 8 GPS channels were able to maintain accurate tracking replicas of 6 GLONASS channels. The simulation results validate the capability of the GPS & GLONASS vector tracking algorithm developed in the software receiver.

Experimental results taken in challenging signal environments gave significant conclusions on the capabilites of GPS & GLONASS vector tracking. The challenging signal environments included areas of light tree foliage, heavy tree foliage, attenuated signals from a parking deck, and an urban canyon. Originally meant for comparison, a Ublox receiver that processes GPS and GLONASS was used as a reference for the experiments. The Ublox receiver used atmospheric GPS corrections from WAAS, giving it an advantage that was not accessible to the software receiver. In each experiment, vector tracking outperformed scalar tracking as expected. The results also show that GPS gives better positioning results than GLONASS. This is partially due to the error associated with the adopted WGS84 to PZ-90.11 transformation used in the software receiver. However, the variance estimations from the GLONASS filters suggest the constellation degrades more significantly by multipath than GPS. This may be from the use of FDMA for satellite identification by GLONASS.

In the experimental runs, GPS & GLONASS scalar tracking always outperformed GPS scalar tracking. The additional GLONASS measurements in the dual constellation software receiver provided the navigation processor extra information to estimate an accurate solution. If the GLONASS measurements were poor, the Kalman filter's FDE would remove the measurements. GPS & GLONASS vector tracking outperformed GPS vector tracking in the heavy tree foliage and attenutated signal environments. GPS & GLONASS vector tracking performed with approximately the same accuracy as GPS vector tracking in the light tree foliage and urban canyon experiments. Noise sharing amongst the GPS and GLONASS channels degraded the receiver solution in these environments. The GPS and GLONASS vector receiver had to maintain more degraded signals than the GPS vector receiver, resulting in worse performance. In other words, vectorized signal tracking performance can be decreased by adding weak satellite channels to the filter, as the channels bring more noise and error to the estimator than navigation aiding.

Overall, the experimental results showed that navigating in challenging signal conditions can be improved by transitioning from scalar processing to vector processing. In the scenarios where the inclusion of GLONASS did not add several degraded channels into the filter, GPS & GLONASS vector tracking can improve navigation in challenging signal conditions when compared to GPS vector tracking.

7.2 Future Work

The conclusions from the thesis open up several opportunities of future research to enhance the software-defined vector tracking receiver. Because the addition of GLONASS seemed to help or degrade navigation with respect to GPS in different environments, a detailed study of the GLONASS constellation should be performed. This study would address the coordinate transformation issue between WGS84 and PZ-90.11, preferably with several data sets that include differenential or atmospheric corrections. An analysis of estimating single or separate receiver clock biases for the satellite systems may provide useful information about GLONASS time accuracy. Results in the thesis suggested that GLONASS is more affected by multipath, modeling this multipath error would help the software receiver perform integrity monitoring on GLONASS [46]. The thesis provided clear sky positioning results, however, more clear sky positioning analysis can give strong insight on how GLONASS helps or hurts navigation.

The EVK-M8T Ublox receiver, that uses both GPS and GLONASS in an advanced tracking algorithm, was used as a reference for the software receiver. Comparing software receiver results with other receivers can also give insight on expected performance with both constellations. The Javad receiver uses GPS and GLONASS in a CoOP tracking algorithm [43], similar to the vector tracking filter discussed here. There are many types of vector tracking estimators, investigation of these different algorithm designs may show improvement in robustness. The first step would be to study pure GPS and GLONASS VDFLL, without carrier phase aiding or FDE.

Along with analyzing the vector processor, the software receiver's navigation filter can be complemented with other sensors to enhance performance. Results showed that the receiver failed when GPS and GLONASS were both jammed. Including an IMU into the vector receiver's navigation processor would mitigate jamming and result in a deeply integrated GNSS/INS algorithm, a new topic in navigation. Other satellite systems could also be included into the vector receiver, such as Galileo (European Union) and Beidou (China). Multi-constellation vector tracking is also a new topic to the field of navigation.

References

[1] A. Joseph. Measuring GNSS Signal Strength. *Inside GNSS*, November/December 2010 Edition, December 2010, pp. 20-25.

[2] A. Tabatabaei, M.R. Mosavi, H.S. Shahhoseini, and K. Borre. Vectorized and Federated Software Receivers Combining GPS and GLONASS. *The Journal of GNSS: GPS Solutions*, 21(3): 1331-1339, March 2017.

[3] Amir Tabatabaei and Mohammad-Reza Mosavi. Performance Analysis of GLONASS Integration with GPS Vectorized Receiver in Urban Canyon Positioning. *Survey Review*, June 2018, pp. 1-12.

[4] Anthony S. Abbott and Walter E. Lillo. Global Positioning Systems and Inertial Measuring Unit UltraTight Coupling Method, El Segundo, CA, February 2003. US Patent 6,516,021.

[5] Benjamin J. Clark. *Fault Detection and Exclusion in Deeply Integrated GPS/INS Navigation*. PhD Dissertation, Auburn University, December 2012.

[6] Brian A. Keyser. *Design and Implementation of a SoC-Based Real-Time Vector Tracking GPS Receiver*. Master's Thesis, Auburn University, May 2015.

[7] Car GPS Blocker Anti Signal Tracking Blocker Car Cigarette Lighter Power Supply. www.tomtop.com. 2019.

[8] Chao-heh Cheng. *Calculations for Positioning with the Global Navigation Satellite System*. Master's Thesis, Ohio University, August 1998.

[9] Christopher R. Hamm. *Analysis of Simulated Performance of Integrated Vector Tracking and Navigation Loops for GPS*. Master's Thesis, Auburn University, December 2005.

[10] David E. Lewis. Ultra-Tightly Coupled GPS/INS Tracking Performance. In *AIAA's 3rd Annual Aviation Technology, Integration, and Operations (ATIO) Tech*, Denver, CO, November 2003, pp. 1-7.

[11] Don Benson. Interference Benefits of a Vector Delay Lock Loop (VDLL) GPS Receiver. In *Proceedings of the 63rd Annual Meeting of the Institute of Navigation*, Cambridge, Massachusetts, April 2007.

[12] Dennis M. Akos. *A Software Radio Approach to Global Navigation Satellite System Receiver Design*. PhD Dissertation, Ohio University, August 1997.

[13] D.W. Lim, H.W. Kang, S.L. Cho, S.J. Lee, and M.B. Heo. Performance Evaluation of a GPS Receiver with VDFLL in Harsh Environments. *IGNSS Symposium 2013*, Outrigger Gold Coast, Australia, July 2013.

[14] E.M. Copps, G.J. Geier, W.C. Fidler, and P.A. Grundy. Optimal Processing of GPS Signals. *Navigation: The Journal of the Institute of Navigation*, 27(3):171-182, Fall 1980.

[15] Ernest J. Ohlmeyer. Analysis of Ultra-Tight Coupled GPS/INS System in Jamming. In *Proceedings of the 2006 IEEE/ION Position, Location, and Navigation Symposium (PLANS)*, San Diego, CA, April 2006, pp. 44-53.

[16] FAA William J. Hughes Technical Center NSTB/WAAS T&E Team. Wide-Area Augmentation System Performance Analysis Report # 26. Atlantic City, NJ, United States, October 2008.

[17] GPS/GNSS Simulators. www.steatite-rugged.co.uk. 2018.

[18] GPS Timing Applications. www.gps.gov/applications/timing/. 2018.

[19] H. So, T. Lee, S. Jeon, C. Kim, C. Kee, T. Kim, and S. Lee. Implementation of a Vector-based Tracking Loop Receiver in a Pseudolite Navigation System. *Sensors*, 10:6324-6346, 2010.

[20] James J. Brewer. *The Differential Vector Phase-Locked Loop For Global Navigation Satellite System Signal Tracking*. PhD Dissertation, Air Force Institute of Technology, June 2014.

[21] James J. Spilker. Vector Delay Lock Loop Processing of Radiolocation Transmitter Signals, Stanford, CA, March 1995. US Patent 5,398,034.

[22] Jay Prakash. *Development of a GLONASS SDR*. Singapore University of Technology and Design, April 2014.

[23] John P. Costas. Synchronous Communications. In *Proceedings of the IRE*, Syracuse, New York, August 1956, pp. 1713-1718.

[24] J. Sennott and D. Senffner. Navigation Receiver with Coupled Signal-Tracking Channels, Bloomington, IL, August 1994. US Patent 5,343,209.

[25] Jeff M. Horslund and Jonathan R. Hooker. Increase Jamming Immunity by Optimizing Processing Gain for GPS/INS Systems, Dallas, TX, November 1999. US Patent 5,983,160.

[26] Kai Borre, Dennis Akos, Nicolaj Bertelsen, Peter Rinder, and Soren Holdt Jensen. *A Software-Defined GPS and Galileo Receiver: A Single Frequency Approach*. Birkhauser, 2007.

[27] L. Hsu, P.D. Groves, and S. Jan. Assessment of the Multipath Mitigation Effect of Vector Tracking in an Urban Environment. In *Proceedings of the ION 2013 Pacific PNT Meeting*, Honolulu, Hawaii, April 2013, pp. 498-509.

[28] Lyle Johnson and Frank van Diggelen. Advantages of a Combined GPS+GLONASS Precision Sensor for Machine Control Applications in Open Pit Mining. In *Proceedings of*

IEEE PLANS 1998 Session C5: Agriculture and Machine Control Applications, Palm Springs, CA, April 1996 (Published 1998), pp. 549-554.

[29] Mark L. Psiaki and Hee Jung. Extended Kalman Filter Methods for Tracking Weak GPS Signals. In *Proceedings of ION GPS 2002*, Portland, Oregon, September 2002, pp. 2539-2552.

[30] Matthew V. Lashley. *Kalman Filter Based Tracking Algorithms For Software GPS Receivers*. Master's Thesis, Auburn University, December 2006.

[31] Matthew V. Lashley. *Modeling and Performance Analysis of GPS Vector Tracking Algorithms*. PhD Dissertation, Auburn University, December 2009.

[32] M. Lashley and D. M. Bevly. Analysis of Discriminator Based Vector Tracking Algorithms. In *Proceedings of the 2007 National Technical Meeting of the Institute of Navigation*, San Diego, CA, January 2007, pp. 570-576.

[33] Matthew Lashley, David M. Bevly, and John Y. Hung. Impact of Carrier to Noise Power Density, Platform Dynamics, and IMU Quality on Deeply Integrated Navigation. In *Proceedings of the 2008 IEEE/ION Position, Location, and Navigation Symposium (PLANS)*, Monterey, CA, May 2008, pp. 9-16.

[34] M. Lashley and D. M. Bevly. Vector Delay/Frequency Lock Loop Implementation and Analysis. In *Proceedings of the 2009 International Technical Meeting of the Institute of Navigation*, Anaheim, CA, January 2009, pp. 1073-1086.

[35] Matthew Lashley and David M. Bevly. What About Vector Tracking Loops? *Inside GNSS*, May/June 2009 Edition, April 2009, pp. 16-21.

[36] M. Lashley, D.M. Bevly, and J.Y. Hung. Performance Analysis of Vector Tracking Algorithms for Weak GPS Signals in High Dynamics. *IEEE Journal of Selected Topics in Signal Processing*, 3(4):661-673, August 2009.

[37] M. Lashley, D.M. Bevly, and J.Y. Hung. A Valid Comparison of Vector and Scalar Tracking Loops. In *Proceedings of the 2010 IEEE/ION Position, Location, and Navigation Symposium (PLANS)*, Indian Wells, CA, May 2010, pp. 464-474.

[38] Matthew Lashley, David M. Bevly, and John Y. Hung. Analysis of Deeply Integrated and Tightly Coupled Architectures. In *Proceedings of the 2010 IEEE/ION Position, Location, and Navigation Symposium (PLANS)*, Indian Wells, CA, May 2010, pp. 382-396.

[39] M. Lashley and D.M. Bevly. Comparison in the Performance of the Vector Delay/Frequency Lock Loop and Equivalent Scalar Tracking Loops in Dense Foliage and Urban Canyon. In *Proceedings of the 24th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2011)*, Portland, OR, September 2011, pp. 1786-1803.

[40] Matthew Lashley and David Bevly. Vector Delay Lock Loops. *Inside GNSS*, September/October 2012 Edition, September 2012, pp. 28-34.

[41] Ministry of Defense of the Russian Federation. System of Geodetic Parameters "Parametry Zemli 1990". 2014.

[42] M. Rao and G. Falco. Code Tracking and Pseudoranges. *Inside GNSS*, January/February 2012 Edition, January 2012, pp. 26-33.

[43] M. Zhodzishsky, S. Yudanov, V. Veitsel, and J. Ashjaee. Co-OP Tracking for Carrier Phase. In *Proceedings of the 11th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS 1998)*, Nashville, TN, September 1998, pp. 653-664.

[44] Navstar GPS Joint Program Office. ICD-GPS-200H (GPS Interface Control Document). September 2014.

[45] Naval History and Heritage. The Navy Navigation Satellite System. 1967.

[46] N. Norris, F. van Graas, and E. Vinande. Implementation and Evaluation of GPS/GLONASS RAIM. In *Proceedings of the 2019 International Technical Meeting of The Institute of Navigation*, Reston, Virginia, January 2019, pp. 802-815.

[47] Pedro Filipe Faria Nogueira Ferrão. *Positioning with Combined GPS and GLONASS Observations*. Master's Thesis, Technico Lisboa, May 2013.

[48] P. D. Groves. *GNSS, Inertial, and Multisensor Integrated Navigation Systems*. Artech House, Second Edition, 2013.

[49] P. Misra and P. Enge. *Global Positioning Systems: Signals, Measurements, and Performance*. Ganga-Jamuna Press, 2011.

[50] Phillip Ward. Performance Comparison Between FLL, PLL, and a Novel FLL-Assisted-PLL Carrier Tracking Loop Filter Under RF Interference Conditions. In *Proceedings of The Institute of Navigation International Technical Meeting*, Nashville, Tennesse, September 1998, pp. 783-795.

[51] Pratap Misra. Integrated Use of GPS and GLONASS in Civil Aviation. *Lincoln Laboratory Journal*, 6(2):231-248, 1993.

[52] Pratap Misra. Integrated Use of GPS and GLONASS: Transformation Between WGS84 and PZ-90. In *Proceedings of ION GPS 1996*, Kansas City, MO, September 1996, pp. 307-314.

[53] P.W. Ward. *Understanding GPS/GNSS Principles and Applications*. Artech House, Third Edition, 2017, pp. 459-463.

[54] R. G. Brown and P. Y. C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering*. John Wiley and Sons, Fourth Edition, 2012.

[55] Russian Institute of Space Device Engineering. GLONASS Interface Control Document, Edition 4.0. 1998.

[56] Russian Institute of Space Device Engineering. GLONASS Interface Control Document, Edition 5.1. 2008.

[57] Salomon Honkala. *GLONASS Satellite Navigation Signal Implementation in a Software-defined Multi-constellation Satellite Navigation Receiver*. Master's Thesis, Aalto University, March 2016.

[58] Saloomeh Abbasiannik. *Multichannel Dual Frequency GLONASS Software Receiver in Combination with GPS L1 C/A*. Master's Thesis, University of Calgary, April 2009.

[59] Scott M. Martin and David M. Bevly. Improved GPS Carrier Phase Tracking in Difficult Environments Using Vector Tracking Approach. *2014 PNT Symposium*, Stanford University, California, October 2014.

[60] Scott M. Martin. *Closely Coupled GPS/INS Relative Positioning For Automated Vehicle Convoys*. Master's Thesis, Auburn University, May 2011.

[61] Scott M. Martin. *GPS Carrier Phase Tracking in Difficult Environments Using Vector Tracking For Precise Positioning and Vehicle Attitude Estimation*. PhD Dissertation, Auburn University, May 2017.

[62] Senlin Peng. *Implementation of Real-Time Sofware Receiver for GPS or GLONASS L1 Signals*. Master's Thesis, Virginia Polytechnic Institute and State University, January 2010.

[63] S. Kiesel, C. Ascher, D. Gramm, and G. Trommer. GNSS Receiver with Vector Based FLL-Assisted PLL Carrier Tracking Loop. In *Proceedings of ION ITM 2008*, Savannah, Georgia, September 2008, pp. 197-203.

[64] S. Stevanovic and B. Pervan. Phase Locked Loop Performance Metrics Based on the Distribution of the Discriminator Output. In *Proceedings of the 2018 International Technical Meeting of the Institude of Navigation*, Reston, Virginia, January 2018, pp. 402-411.

[65] Susmita Bhattacharyya. *Performance and Integrity Analysis of the Vector Tracking Architecture of GNSS Receivers*. PhD Dissertation, University of Minnesota, April 2012.

[66] S. Zhao and D. Akos. An Open Source GPS/GNSS Vector Tracking Loop - Implementation, Filter Tuning, and Results. In *Proceedings of the 2011 International Technical Meeting of The Institute of Navigation*, San Diego, CA, January 2011, pp. 1293-1305.

[67] Tom Van Baak. www.LeapSecond.com. 2019.

[68] T. Watts, S. Martin, and D. Bevly. A GPS and GLONASS L1 Vector Tracking Software-Defined Receiver. In *Proceedings of the 2019 International Technical Meeting of The Institute of Navigation*, Reston, Virginia, January 2019, pp. 162-176.

[69] U-blox. UBX-16000801 - R06, Product Summary NEO/LEA-M8T series. 2018.

[70] V. Vdovin and A. Dorofeeva. Global Geocentric Coordinate System of the Russian Federation. WG D – Reference Frames, Timing and Applications, November 7, 2012.

[71] W. Luke Edwards, Benjamin J. Clark, and David M. Bevly. Implementation Details of a Deeply Integrated GPS/INS Software Receiver. In *Proceedings of IEEE/ION Position, Location, and Navigation Symposium*, Indian Wells, CA, May 2010, pp. 1137-1146.

[72] W. Luke Edwards. *Development of a GPS Software Receiver on an FPGA for Testing Advanced Tracking Algorithms*. Master's Thesis, Auburn University, August 2010.

[73] Z. He. Maximum Likelihood Vector Tracking. *Inside GNSS*, July/August 2013 Edition, January 2013, pp. 27-30.

Appendices

Appendix A

GNSS Kalman Filtering for Navigation

Initially, GNSS receivers use iterative least squares to localize PVT. From that convergence point, receivers move to error-state estimators to precisely navigate over time. These error-state estimators use the current navigation solution as well as GNSS measurements to determine the PVT state at the next time step. This appendix discusses a series of error-state estimators, moving up to the Kalman filter used in the software receiver.

All error-state estimators require the state vector x that contains the estimated receiver PVT, the measurement vector Y that contains the pseudorange ρ and pseudorange rate $\dot{\rho}$ differences between measurements and predictions, and the measurement observation matrix H that relates the state vector to the measurement vector. Equations (A.1) through (A.3) express these variables for m satellite channels, where each satellite channel provides a pseudorange and pseudorange rate measurement.

$$x = \begin{bmatrix} \hat{x}_r & \hat{\dot{x}}_r & \hat{y}_r & \hat{\dot{y}}_r & \hat{z}_r & \hat{\dot{z}}_r & c\hat{b}_r & c\hat{\dot{b}}_r \end{bmatrix}^T \tag{A.1}$$

$$Y = \begin{bmatrix} \delta\rho_1 & \dots & \delta\rho_m & \delta\dot{\rho}_1 & \dots & \delta\dot{\rho}_m \end{bmatrix}^T \tag{A.2}$$

$$H = \begin{bmatrix} -a_{x_1} & 0 & -a_{y_1} & 0 & -a_{z_1} & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -a_{x_m} & 0 & -a_{y_m} & 0 & -a_{z_m} & 0 & 1 & 0 \\ 0 & -a_{x_1} & 0 & -a_{y_1} & 0 & -a_{z_1} & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -a_{x_m} & 0 & -a_{y_m} & 0 & -a_{z_m} & 0 & 1 \end{bmatrix} \quad \text{(A.3)}$$

The state vector has an array size of 8 x 1, the measurement vector has an array size of $2m$ x 1, and the measurement observation has a matrix size of $2m$ x 8.

The estimation results shown in this Appendix come from using 10 GPS satellites and 5 GLONASS satellites, while driving on I-85 in Auburn, Alabama. The GNSS measurements are generated from scalar tracking operations using CRT synchronization.

A.1 Recursive Least Squares Estimation

The simplest error-state estimator that provides reasonable accuracy is recursive least squares (RLS), shown by Equation (A.4).

$$x_{k+1} = x_k + (H^T H)^{-1} H^T Y \quad \text{(A.4)}$$

In RLS, the measurement vector and the least squares solution are used to update the PVT estimation. RLS provides minimal filtering because the least squares solution simply generates the “best-fit” for the GNSS measurements provided, with respect to satellite geometry. This best-fit will remove some error from the estimation, but will be noisy overall. Including more unique satellite channels into the estimator will improve the solution because the estimator has more information to generate a better fit.

Least squares can be improved by weighing the pseudoranges and pseudorange rates with a measurement covariance matrix R. This estimator is known as weighted RLS (WRLS), and its operation is shown by Equation (A.5).

$$x_{k+1} = x_k + (H^T R^{-1} H)^{-1} H^T R^{-1} Y \tag{A.5}$$

The measurement covariance matrix weighs the GNSS measurements, so the least-squares fit relies more on accurate measurements and less on noisy measurements. The covariance matrix R is shown by Equation (A.6), which has a size of $2m$ x $2m$ for m satellite channels being used by the estimator.

$$R = \begin{bmatrix} \sigma^2_{\rho 1} & \cdots & \sigma_{\rho 1}\sigma_{\rho m} & \sigma_{\rho 1}\sigma_{\dot{\rho} 1} & \cdots & \sigma_{\rho 1}\sigma_{\dot{\rho} m} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{\rho m}\sigma_{\rho 1} & \cdots & \sigma^2_{\rho m} & \sigma_{\rho m}\sigma_{\dot{\rho} 1} & \cdots & \sigma_{\rho m}\sigma_{\dot{\rho} m} \\ \sigma_{\dot{\rho} 1}\sigma_{\rho 1} & \cdots & \sigma_{\dot{\rho} 1}\sigma_{\rho m} & \sigma^2_{\dot{\rho} 1} & \cdots & \sigma_{\dot{\rho} 1}\sigma_{\dot{\rho} m} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{\dot{\rho} m}\sigma_{\rho 1} & \cdots & \sigma_{\dot{\rho} m}\sigma_{\rho m} & \sigma_{\dot{\rho} m}\sigma_{\dot{\rho} 1} & \cdots & \sigma^2_{\dot{\rho} m} \end{bmatrix} \tag{A.6}$$

The diagonals of R contain the variances of the pseudoranges and pseudorange rates. The off-diagonals of R contain the covariances between the satellite measurements. Measurements between different satellites are assumed to be uncorrelated, which allows the off-diagonals of R to be zero, as was shown in Equation (4.48). For GNSS, measurement variances are typically predicted using the satellite channels' C/N$_\text{O}$ ratios. Measurement variances based on C/N$_\text{O}$ ratio were shown in Equations (4.45) and (4.46).

Figure A.1 shows positioning results using RLS and WRLS. The least squares estimations provide accurate results, but with little precision; the estimations cover the majority of the interstate road. WRLS is slightly more accurate than RLS. Only a small shift in performance occurs because the estimators are using many high quality measurements from GPS and GLONASS satellite channels.

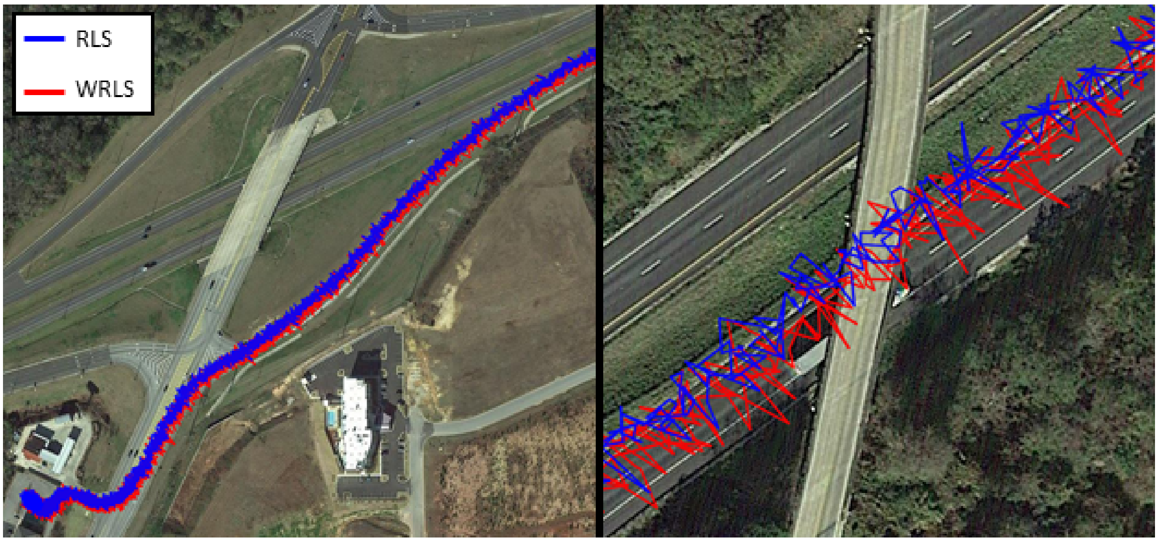

Figure A.1: Positioning Results using RLS and WRLS Estimators

A.2 Recursive Least Squares with Model and Estimation Gain

The least squares solutions only filter the measurements through geometrical fitting, the receiver PVT estimation can be improved by providing additional filtering through the use of a model. Adding a model to the estimator is the first step to a Kalman filter. In many cases, the dynamics of the receiver antenna can be predicted and applied to a model to improve the solution's precision. Equation (A.7) shows the estimation process of including a model with WRLS.

$$\begin{aligned} x_{k+1}^{-} &= Ax_k^{+} \\ x_{k+1}^{+} &= x_{k+1}^{-} + G(H^TR^{-1}H)^{-1}H^TR^{-1}Y \end{aligned} \tag{A.7}$$

$$A = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

In the estimator, (-) denotes the estimated state before the measurement update, and (+) denotes the estimated state after the measurement update. This estimator first applies a state update using an 8 x 8 state-space model A. Depending on the system, the estimator may also use an input Bu_k with the model. Then the estimator applies a measurement correction with WRLS and a scalar estimation gain G. The gain determines how much trust the estimator places onto the model relative to the measurement. When G is high, the estimator places more trust onto the measurement than the model. When G is low, the estimator places more trust onto the model than the measurement.

Note that the state-space model does not have to be complicated; this work uses a kinematic model where ECEF velocities update ECEF positions and clock drift updates clock bias. Models will improve the estimator so long as they accurately represent the dynamics of the receiver antenna. If the state-space model provides inaccurate dynamics, the estimator will degrade with the inclusion of the model.

Figure A.2 shows positioning results when the model is included with WRLS. Results are shown when the estimation gain is $G = 0.1$, $G = 0.01$, and $G = 0.001$. When $G = 0.1$, the estimated position is much less noisy than WRLS alone, but oscillatory. This estimation is on the edge of instability. When G becomes too large, the measurement correction to the state is over amplified, and the estimator becomes unstable. When $G = 0.01$, the estimated position follows the interstate very well, this solution is accurate and precise. From this, it can be said that the state-space model A is an accurate representation of interstate driving, since the estimation gain is small. This makes sense because interstate driving consists of motion at a relatively high constant velocity. When $G = 0.001$, the estimated position is precise, but inaccurate because the estimator relies too much on the model, which causes the position to go off the interstate. The measurement correction is always important to have since no model can fully represent a real dynamic system.

Figure A.2: Positioning Results using a Model and Gain with WRLS

A.3 State-Space Estimator

Instead of using a single gain to weigh the model and measurement trust, a gain matrix L can be used to improve the estimator. This matrix contains an array of gains for each state-variable, where each gain in a variable's array is associated to a measurement. In the GNSS filter, the matrix L has a size of 8 x $2m$. This form of filtering is known as state-space estimation. Note that Kalman filters are state-space estimators. Equation (A.8) describes the multi-gain filtering process.

$$\begin{aligned} x_{k+1}^{-} &= Ax_k^{+} \\ x_{k+1}^{+} &= x_{k+1}^{-} + LY \\ L &= \text{place}(A_c^T, H^T, s)^T \end{aligned} \tag{A.8}$$

The state is first updated by the discrete-time model A and then corrected by the measurement Y using the gain L. The gain can be calculated in MATLAB using the place command. The command requires inputs of the continous-time model A_c, the measurement observation H, and an array of continuous-time poles s. The poles specify the transient response of each estimated state-variable. The gain can only be calculated if the system is observable, which is the case for the GNSS model and observation.

Figure A.3 shows velocity responses in the ECEF y-direction using the state-space estimator. The responses are shown when the estimator uses continuous-time eigenvalues of $s = -0.01$, $s = -0.1$, and $s = -0.2$. The eigenvalues are all negative to ensure the estimator is stable. Initially, all the responses are precise, but two of the responses become noisy due to multipath from tree foliage. The velocity estimation when $s = -0.01$ does well filtering out the multipath effects compared to the other estimated velocities. When the eigenvalues are at $s = -0.01$, the state-space estimator behaves very closely to an integrator, or an ideal low-pass filter. The low-pass traits of the system attenuates the multipath error, but also induces time (phase) lag on the estimation as seen in the close-up figure. The estimator's eigenvalues need to be optimally chosen to ensure noise and other errors are filtered out without significant time delay.

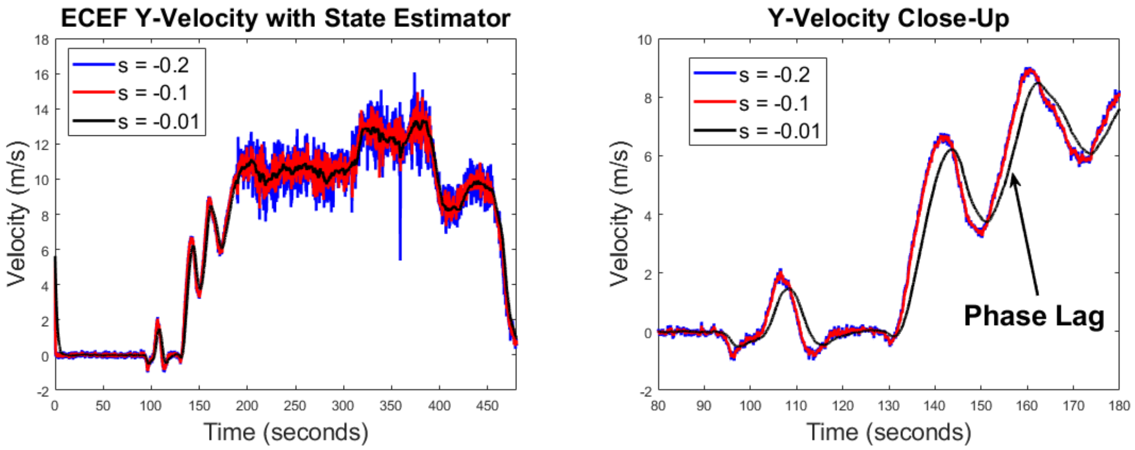

Figure A.3: ECEF Y-Velocity Responses using State-Space Estimation

Figure A.4 shows positioning results using the state-space estimators from the previous figure. The estimator with the smallest eigenvalue magnitude appears to suffer from convergence issues at transient dynamics. This issue arises from too much time delay in the response, induced by the eigenvalues used to calculate L. In other words, the low magnitude eigenvalues tell the estimator to put too much trust in the model and not enough in the measurement. Hence, when the vehicle turns or changes its dynamics, the estimator is slow to respond. Overall, none of the state-space filters perform well, as the position estimations are very noisy and oscillatory. State-space works well for linear systems, but the GNSS system is nonlinear. The calculation of L from Equation (A.8) also does not include the measurement covariance matrix R that weighs the measurements relative to each other. These issues put the state-space estimator at a disadvantage when compared to WRLS using a model.

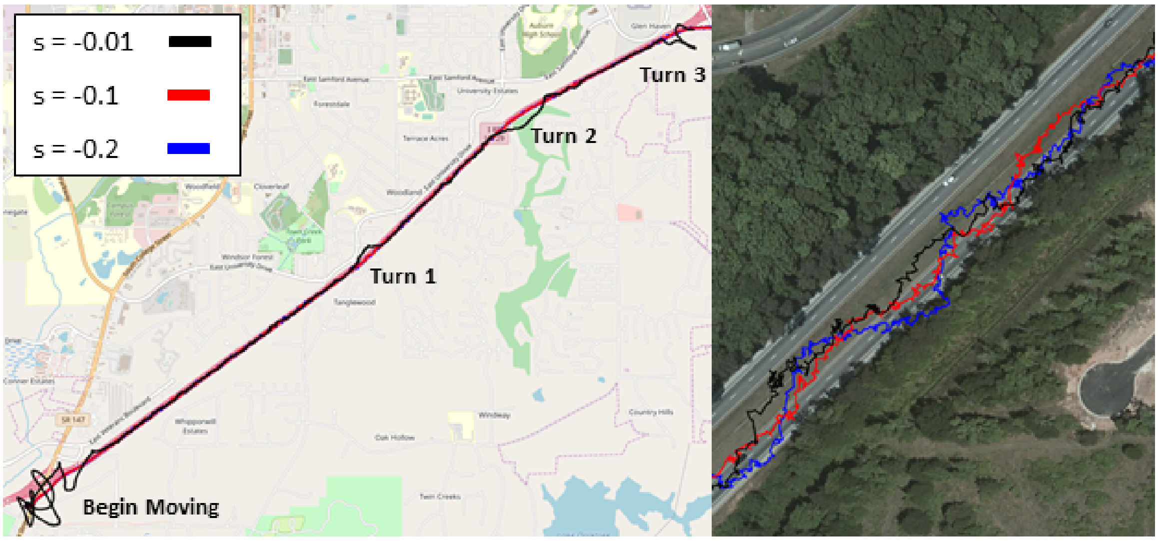

Figure A.4: Positioning Results using State-Space Estimation

A.4 Kalman Filter

The downside to the traditional state-space estimator is that its estimation gains are based on a set of eigenvalues, which indicates the estimator always uses the same trust in model and measurement. This issue also arises with the estimation gain G when using WRLS with a model. In general, systems that need to be estimated are dynamic in model and measurement quality. In GNSS, there are time periods where the estimator should put more trust in the model, and there are time periods where the estimator should put more trust in the measurement. The power of the Kalman filter comes from its ability to adaptively calculate the gain matrix to account for the cyclic change in a system's model and measurement quality.

Equation (A.9) describes the steps of the Kalman filter.

$$\begin{aligned}
x_{k+1}^{-} &= Ax_{k}^{+} \\
P_{k+1}^{-} &= AP_{k}^{+}A^{T} + Q \\
K &= P_{k+1}^{-}H^{T}(HP_{k+1}^{-}H^{T} + R)^{-1} \\
x_{k+1}^{+} &= x_{k+1}^{-} + KY \\
P_{k+1}^{+} &= (I - KH)P_{k+1}^{-}
\end{aligned} \tag{A.9}$$

The filter contains both an estimated state x and estimated state covariance P that are updated by the system model A. The state covariance P, measurement covariance R, and measurement observation H are used to calculate the estimation gain matrix, known as the Kalman gain K. The measurement correction is performed on the state using the Kalman gain and the measurement array Y. A measurement correction is also performed on the state covariance, where I is the identity matrix.

The Kalman filter performs estimation on systems where the state-variables' errors and the measurements' noises are assumed to be unbiased and Gaussian distributed, which results in the use of the state covariance P and measurement covariance R matrices. The estimated state covariance P describes the covariances of the state-variables predicted by the Kalman filter. Equation (A.10) shows the state covariance for the GNSS estimator.

$$P = \begin{bmatrix} \sigma_x^2 & \sigma_x\sigma_{\dot{x}} & \sigma_x\sigma_y & \sigma_x\sigma_{\dot{y}} & \sigma_x\sigma_z & \sigma_x\sigma_{\dot{z}} & \sigma_x\sigma_{cb} & \sigma_x\sigma_{\dot{cb}} \\ \sigma_{\dot{x}}\sigma_x & \sigma_{\dot{x}}^2 & \sigma_{\dot{x}}\sigma_y & \sigma_{\dot{x}}\sigma_{\dot{y}} & \sigma_{\dot{x}}\sigma_z & \sigma_{\dot{x}}\sigma_{\dot{z}} & \sigma_{\dot{x}}\sigma_{cb} & \sigma_{\dot{x}}\sigma_{\dot{cb}} \\ \sigma_y\sigma_x & \sigma_y\sigma_{\dot{x}} & \sigma_y^2 & \sigma_y\sigma_{\dot{y}} & \sigma_y\sigma_z & \sigma_y\sigma_{\dot{z}} & \sigma_y\sigma_{cb} & \sigma_y\sigma_{\dot{cb}} \\ \sigma_{\dot{y}}\sigma_x & \sigma_{\dot{y}}\sigma_{\dot{x}} & \sigma_{\dot{y}}\sigma_y & \sigma_{\dot{y}}^2 & \sigma_{\dot{y}}\sigma_z & \sigma_{\dot{y}}\sigma_{\dot{z}} & \sigma_{\dot{y}}\sigma_{cb} & \sigma_{\dot{y}}\sigma_{\dot{cb}} \\ \sigma_z\sigma_x & \sigma_z\sigma_{\dot{x}} & \sigma_z\sigma_y & \sigma_z\sigma_{\dot{y}} & \sigma_z^2 & \sigma_z\sigma_{\dot{z}} & \sigma_z\sigma_{cb} & \sigma_z\sigma_{\dot{cb}} \\ \sigma_{\dot{z}}\sigma_x & \sigma_{\dot{z}}\sigma_{\dot{x}} & \sigma_{\dot{z}}\sigma_y & \sigma_{\dot{z}}\sigma_{\dot{y}} & \sigma_{\dot{z}}\sigma_z & \sigma_{\dot{z}}^2 & \sigma_{\dot{z}}\sigma_{cb} & \sigma_{\dot{z}}\sigma_{\dot{cb}} \\ \sigma_{cb}\sigma_x & \sigma_{cb}\sigma_{\dot{x}} & \sigma_{cb}\sigma_y & \sigma_{cb}\sigma_{\dot{y}} & \sigma_{cb}\sigma_z & \sigma_{cb}\sigma_{\dot{z}} & \sigma_{cb}^2 & \sigma_{cb}\sigma_{\dot{cb}} \\ \sigma_{\dot{cb}}\sigma_x & \sigma_{\dot{cb}}\sigma_{\dot{x}} & \sigma_{\dot{cb}}\sigma_y & \sigma_{\dot{cb}}\sigma_{\dot{y}} & \sigma_{\dot{cb}}\sigma_z & \sigma_{\dot{cb}}\sigma_{\dot{z}} & \sigma_{\dot{cb}}\sigma_{cb} & \sigma_{\dot{cb}}^2 \end{bmatrix} \tag{A.10}$$

Since the state vector has 8 variables, the state covariance matrix has a size of 8 x 8. Along the matrix diagonal are the estimated variances of the state-variables. The covariances between state-variables fill up the remainder of the matrix. The Kalman filter must be initialized with a state covariance P_0. Typically, P_0 is initialized with estimated variances in the matrix diagonal and zeros in the off-diagonals.

The process noise covariance matrix Q is a tuning parameter that accounts for the uncertainty of unmodeled dynamics in A. The process noise used for the GNSS estimator was expressed in Equation (5.19). Another way to think of the process noise is as a threshold for the state covariance P. The smallest possible value P can be in the calculation of the Kalman gain is Q, as seen in Equation (A.9). In general, a higher Q results in more trust to the measurement, and a lower Q results in more trust to the model. The process noise is very similar in functionality to the noise bandwidth BW_n used in the signal tracking loop filters.

Figure A.5 shows velocity results in the ECEF z-direction using the Kalman filter with a high process noise (loose) and a low process noise (tight). From the responses, it is shown that the tight Kalman filter rejects the multipath error more than the loose Kalman filter. The loose filter's process noise covariance applies more trust to the measurement through the Kalman gain, which results in the multipath entering the velocity estimation.

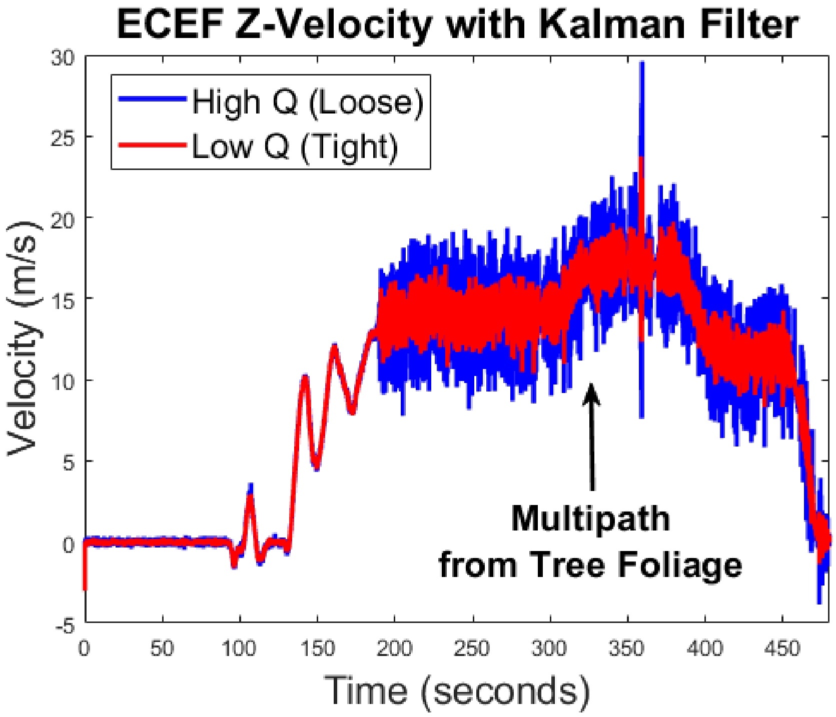

Figure A.5: ECEF Z-Velocity Responses using Kalman Filtering

Figure A.6 shows positioning results for the Kalman filters used in the previous figure. As expected, the Kalman filters perform better than the recursive least squares and state-space estimators. The tight filter performs better under the interstate bridges because it provides less correction with the distorted measurements from blocked signals. Under the bridge, the tight filter relies on the system model. The loose filter places more emphasis on the distorted measurements, which results in the position jump. Overall, both filters perform better under the interstate bridges than the vector receivers in Chapter 5. The increase in accuracy comes from GPS and GLONASS being integrated together into a central filter, instead of individual estimators for each constellation.

Figure A.6: Positioning Results using Kalman Filtering

A.4.1 Fault Detection and Exclusion

The Kalman filter assumes it is inputted with unbiased measurements, without this assumption the estimator can fail. It is common for GNSS measurements to become biased in poor signal environments, such as forests and urban canyons. In these cases, the measurement covariance matrix does not provide accurate information about the measurement error, and the filter is better off throwing out the measurement. The process of detecting and removing a biased measurement is known as Fault Detection and Exclusion (FDE). FDE is fairly simple to implement into a Kalman filter. The presence of biased measurements is estimated by normalized innovations z. A measurement's normalized innovation is a test statistic that describes the probability of the measurement being distorted (biased). The test statistic z_i for measurement i is shown in Equation (A.11).

$$\begin{aligned} z_i &= \frac{Y_i}{\sqrt{C_{ii}}} \\ C &= HPH^T + R \end{aligned} \tag{A.11}$$

The statistic is a function of the state covariance P, measurement observation H, and the measurement covariance R.

A threshold for the normalized innovation is required to determine whether the measurement should or should not be thrown out of the correction. A low threshold increases the probability that an unbiased measurement is thrown out, this is known as false alarm probability. A high threshold increases the probability that a biased measurement is not thrown out, this is known as missed alarm probability. In this work, a threshold of 3 is used, which provides a false alarm probability of 0.25%. A threshold of 3 indicates that if the calculated standard deviation of the measurement is 3 times greater than its expected standard deviation from the estimator, then the measurement is thrown out. Throwing out a measurement requires removing it from the Y, H, and R matrices prior to calculating the Kalman gain. More information on FDE can be found in [48].

Figure A.7 shows velocity results in the ECEF x-direction for a loose Kalman filter, a tight Kalman filter, and a tight Kalman filter using FDE. In this case, the loose filter performs better than the tight filter at rejecting the multipath error. The loose filter likely has better precision because its trust in good measurements outcrowds the biased measurements. Although the tight filter relies more on the model, the biased measurements still leak into the velocity estimation. When FDE is included with the tight filter, the solution's precision increases. With FDE, the estimator is no longer trying to filter the multipath error, but is simply removing it by dropping the measurements biased by multipath.

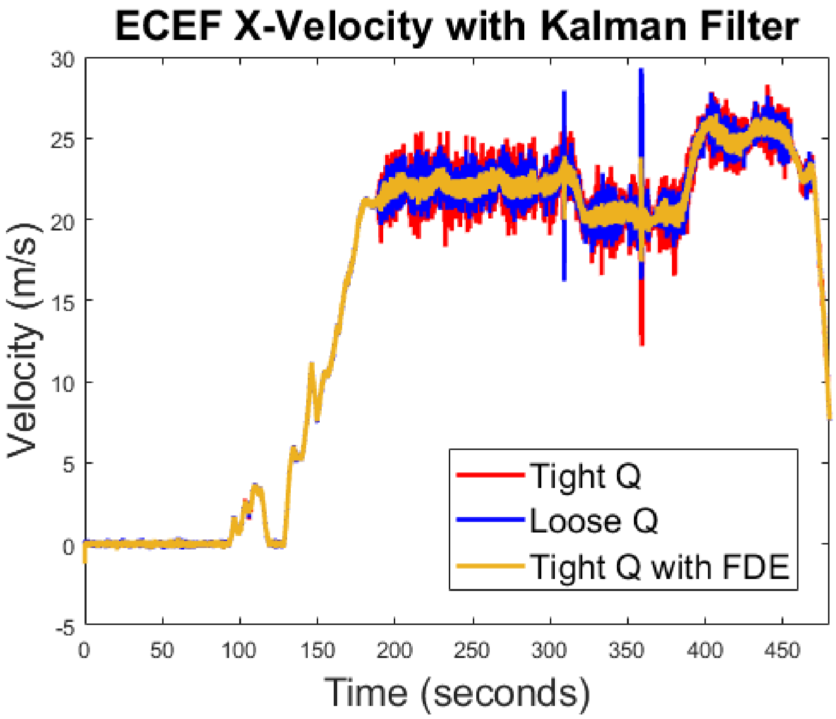

Figure A.7: ECEF X-Velocity Responses using Kalman Filtering with FDE

Figures A.8 and A.9 compare position results when the Kalman filter is and is not using FDE. The process noise covariance tuning is the same for both estimators. Figure A.8 shows positioning under the interstate bridges that induce signal blockages. Figure A.9 shows positioning near tree foliage that induces multipath. Overall, when FDE is included into the estimator, the position estimate remains more stable in the challenging signal environments.

Figure A.8: Positioning Results under Interstate Bridges using Kalman Filtering with FDE

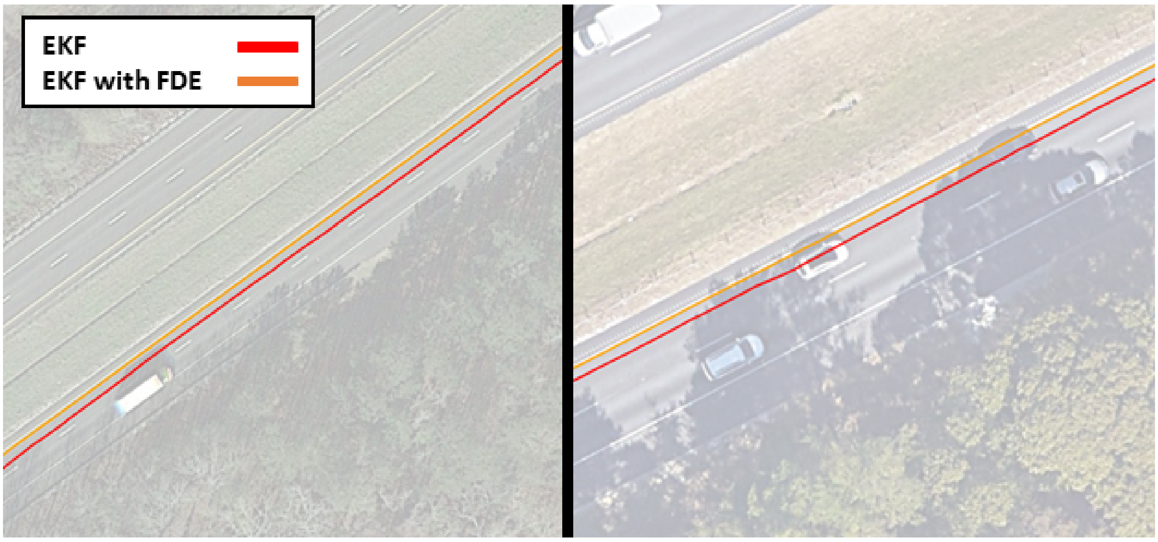

Figure A.9: Positioning Results near Tree Foliage using Kalman Filtering with FDE

FDE also mitigates noise sharing within a vector tracking loop's Kalman filter. Figure A.10 helps describe this concept. In the figure, the top plot shows a GPS satellite channel that incurs a signal blockage in the time frame between the solid black lines. The bottom plot shows another GPS satellite channel that does not contain a signal blockage. The unblocked channel in the bottom plot aids the tracking of the blocked channel in the top plot through the vector loop's estimator. The top channel is able to maintain tracking lock after the blockage ends when using the vector loop. However, during the blockage, the unblocked channel's Doppler frequency is significantly noisier, which is due to noise sharing with the blocked channel. The Doppler frequency becomes less noisy with the implementation of FDE in the vector loop's Kalman filter. FDE throws out the top channel's discriminator measurements during the blockage so they do not degrade the navigation solution. The other GPS channels maintain an accurate navigation state, which keeps replica frequencies accurate on the blocked channel. FDE is another tool that can improve vector tracking loops. FDE is especially applicable to multi-constellation vector tracking, where one secure satellite system can aid the tracking of another system that is jammed or spoofed. More information on the vector tracking effects from FDE can be found in [5].

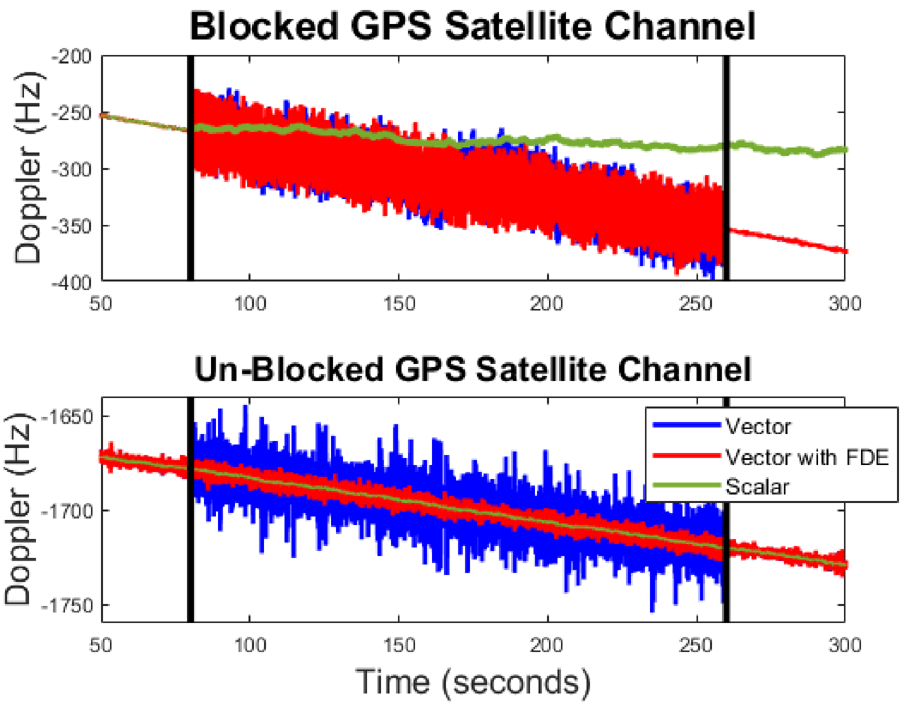

Figure A.10: Using FDE to Mitigate Noise Sharing in a Vector Tracking Loop

A.4.2 Software Receiver Kalman Filter

Figure A.11 shows a diagram of the Kalman filter implemented into the GPS and GLONASS vector tracking receiver. The filter is initalized with a state covariance P_0 and a process noise covariance Q. The filter takes advantage of FDE to remove biased measurements from degrading the navigation solution and the channels' tracking operations. Pseudorange and pseudorange rate residual measurements from both GPS and GLONASS are inputted into the estimator. The measurements are weighed using the channels' C/N$_0$ ratios. The Kalman gain K adaptively calculates the relative trust placed into the estimation between the model and the measurements. When K is near 0 (low P, high R), the estimator places more trust in the model relative to the measurements. When K is near 1 (high P, low R) the estimator places more trust in the measurements relative to the model.

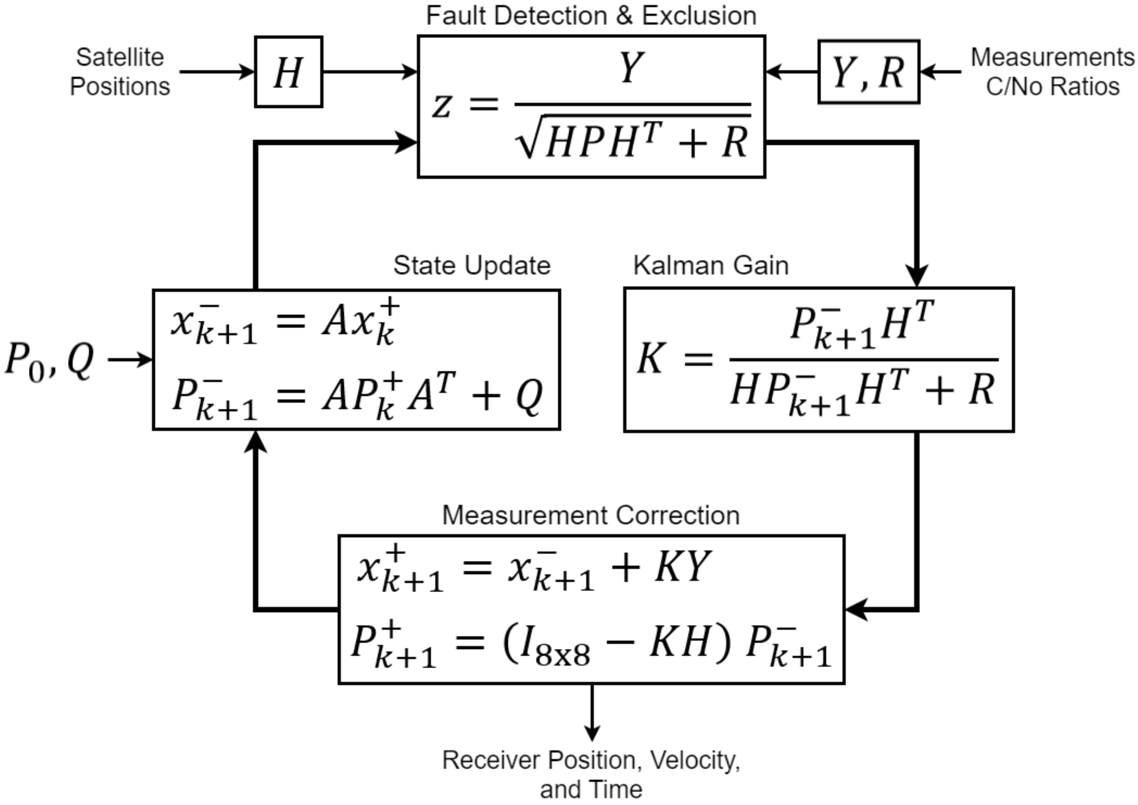

Figure A.11: Diagram of the Kalman Filter Implemented into the Software Receiver

Appendix B

Vector Tracking Measurement Synchronization Results

This appendix provides additional results of the vector tracking measurement synchronization methods discussed in Chapter 6. Specifically, positioning results are shown for the CTT batch and asynchronous (channel and scalar) methods.

Figure B.1 shows downtown Des Moines positioning results with GLONASS using the scalar correction and channel correction asynchronous methods. Positioning from the Ublox EVK-M8T receiver is also shown for reference. Both asynchronous methods destabilize at the same location. The scalar correction method becomes unstable while the channel correction method becomes marginally unstable, maintaining a biased position estimate with respect to the Ublox receiver.

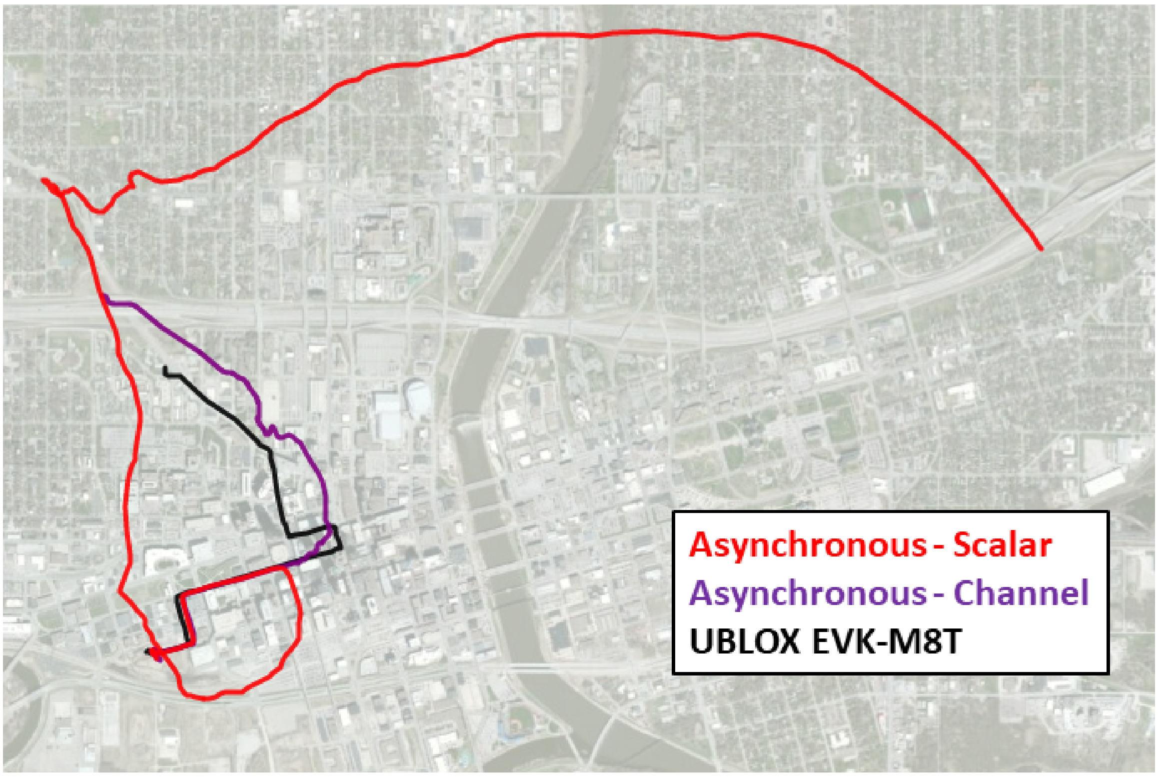

Figure B.1: Comparisons of Asynchronous Channel and Scalar Approaches with GLONASS

Des Moines position results are shown with the asynchronous estimators again in Figure B.2. The left of the figure shows the results when only GPS is used in the filter, and the right of the figure shows the results when GPS and GLONASS are used in the filter. When only GPS is used, the results are similar to the GLONASS estimations in Figure B.1, where the channel correction generates a more stable estimation than the scalar correction. However, the channel correction appears to have more position bias initially with respect to the Ublox receiver, as was seen in Figure 6.9. When GPS and GLONASS are both used in the vector tracking algorithm, the scalar correction method is more stable and accurate than the channel correction. When both constellations are used, the decoupling effect observed in the scalar correction method is not as significant, allowing the algorithm to maintain a stable position estimation.

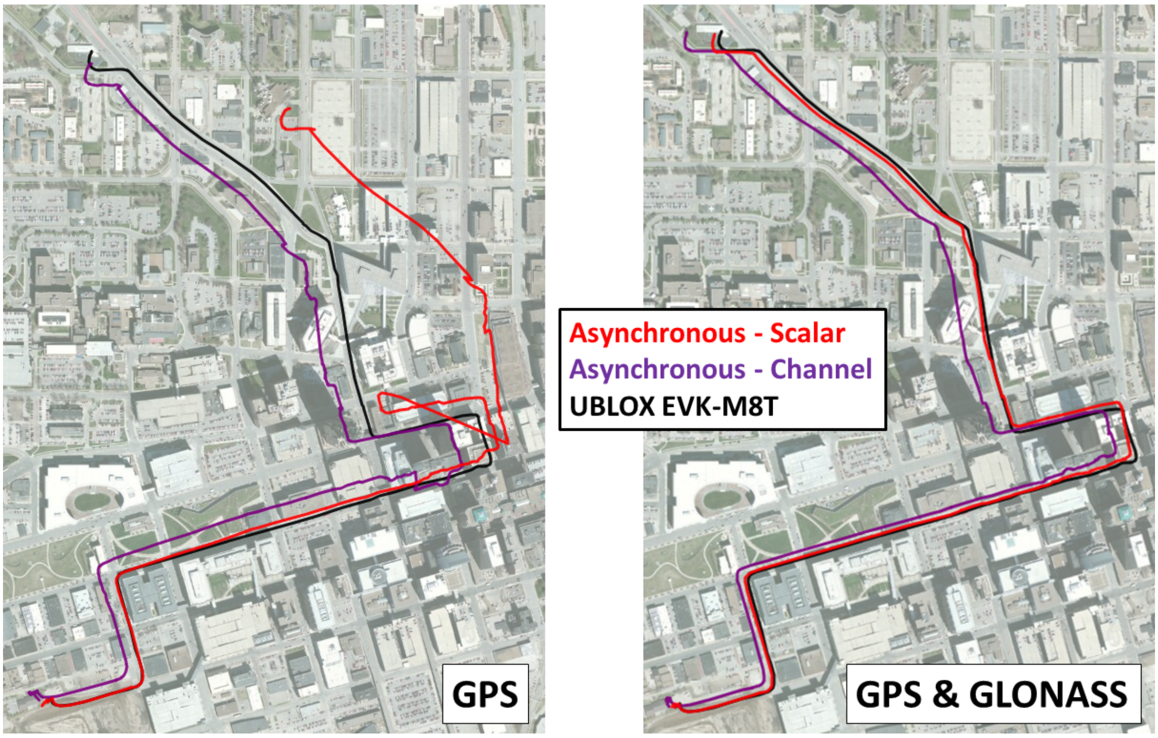

Figure B.2: Comparisons of Asynchronous Channel and Scalar Correction Approaches with GPS and GPS & GLONASS

The asynchronous measurement approach with scalar discriminator corrections is chosen to be analyzed further against the CTT batch approach in the next section. Although less stable than the channel correction method, the scalar method is simpler to implement and provides better position accuracy. The instability effect of the scalar method is also reduced when GPS and GLONASS are combined, due to the increase in satellite channel measurements. However, the asynchronous method needs to be compared to the CTT batch approach because of its instability issues.

B.1 CTT Batch and Asynchronous Scalar Comparisons

To ensure the software receiver in this work uses the optimal vector tracking algorithm, results using the CTT batch and the asynchronous scalar methods are compared below. Each of the results are shown for when the receiver performs vector tracking with both GPS and GLONASS.

Figure B.3 shows static position comparisions of the methods when using high and low process noise covariance tunings in the centralized Kalman filter. When the process noise tuning is high, the CTT batch method is more accurate than the asynchronous scalar method. However, the asynchronous scalar method is more precise than the CTT batch method.

When the process noise tuning is low, both methods have approximately the same accuracy. The asynchronous scalar method is slightly more precise than the CTT batch method. The asynchronous scalar method is more precise because it positions on receive time instead of transmission time as the CTT batch method does. However, when the process noise covariance tuning in the Kalman filter is low, the gain in precision is not as significant. Meaning, a well-tuned Kalman filter can get approximately the same precision out of positioning with measurements on transmission time or receive time.

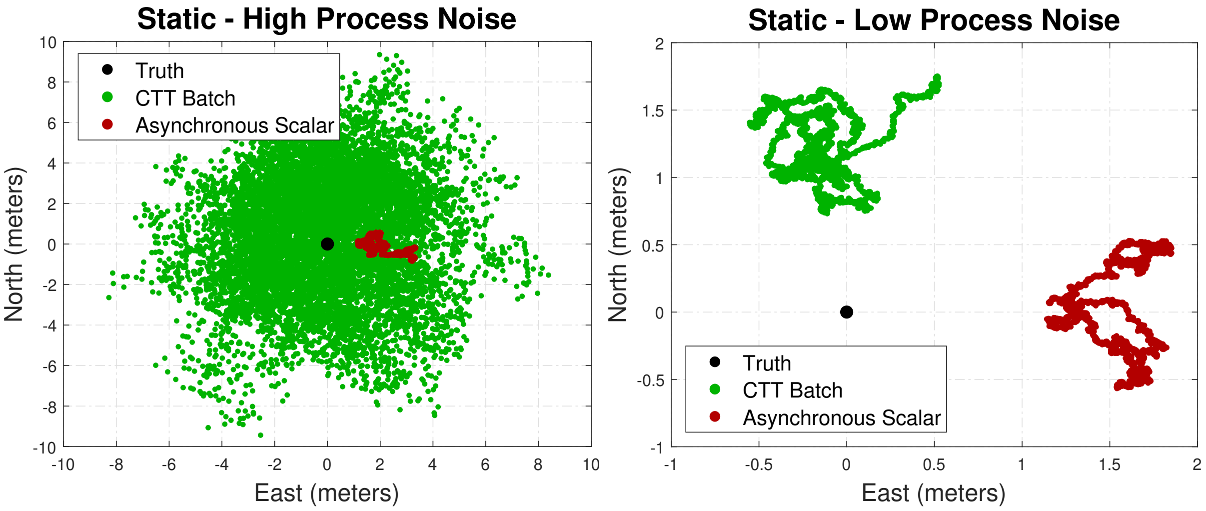

Figure B.3: GPS & GLONASS Vector Tracking Static Position Comparisons with Difference Process Noise Tunings

Figures B.4 through B.11 show estimated positioning and variance results with GPS & GLONASS vector tracking using the CTT batch and asynchronous scalar methods. The results are shown for the experiments performed in Chapter 6. Both methods maintain stable estimations in all the experiments. However, in each case, the CTT batch method outperforms the asynchronous scalar method with respect to the GPS & GLONASS Ublox receiver reference position. From these results, the software receiver's vector algorithm operates with the CTT batch measurement synchronization method. All of the vector tracking results shown in Chapter 6 use the CTT batch method.

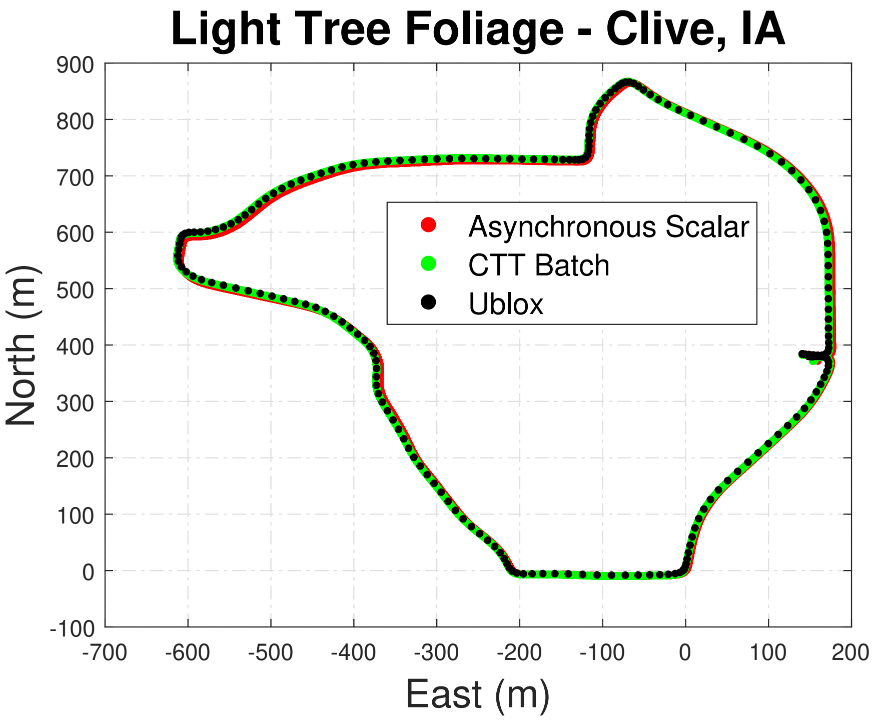

Figure B.4: GPS & GLONASS Vector Tracking Position Comparisons in the Light Tree Foliage Experiment

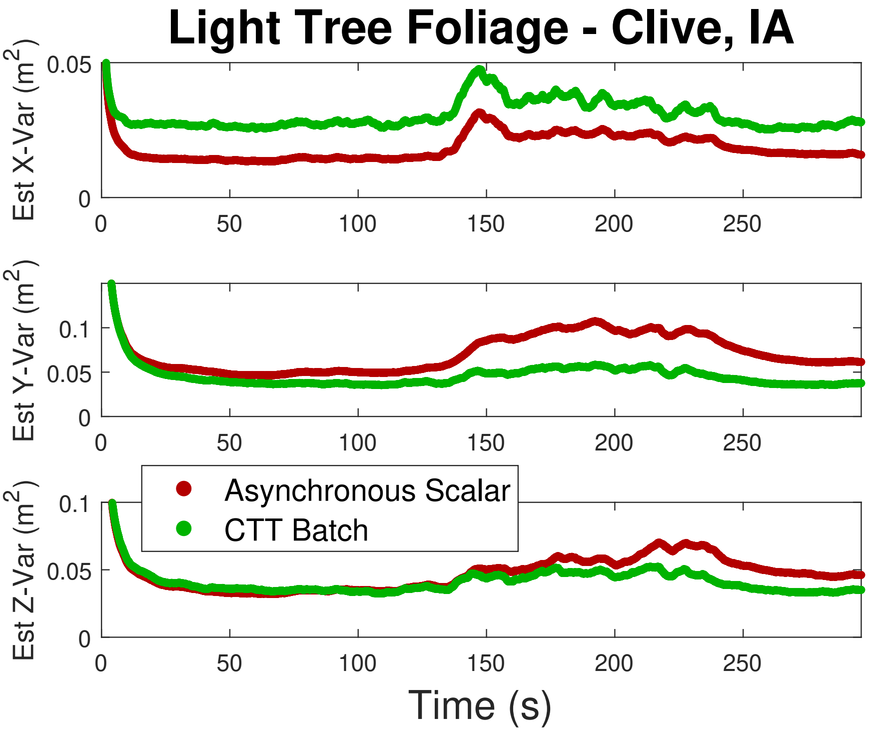

Figure B.5: GPS & GLONASS Vector Tracking Variance Comparisons in the Light Tree Foliage Experiment

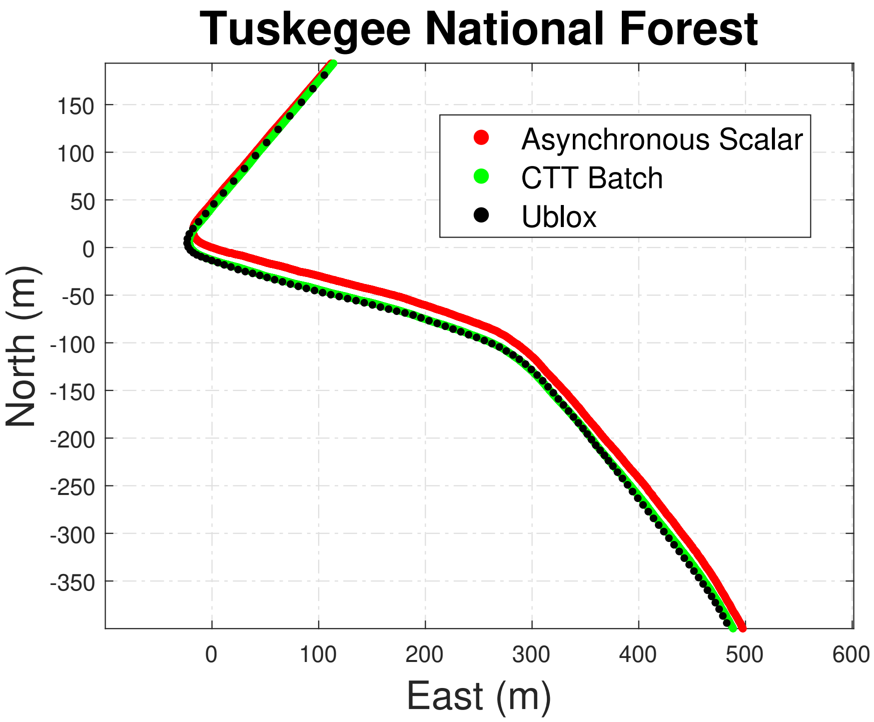

Figure B.6: GPS & GLONASS Vector Tracking Position Comparisons in the Heavy Tree Foliage Experiment

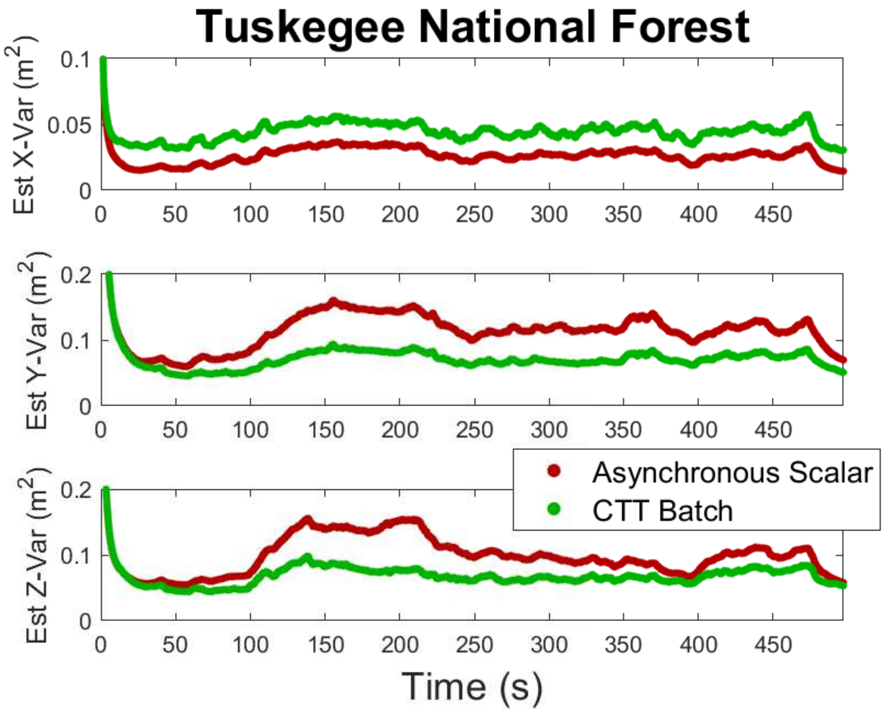

Figure B.7: GPS & GLONASS Vector Tracking Variance Comparisons in the Heavy Tree Foliage Experiment

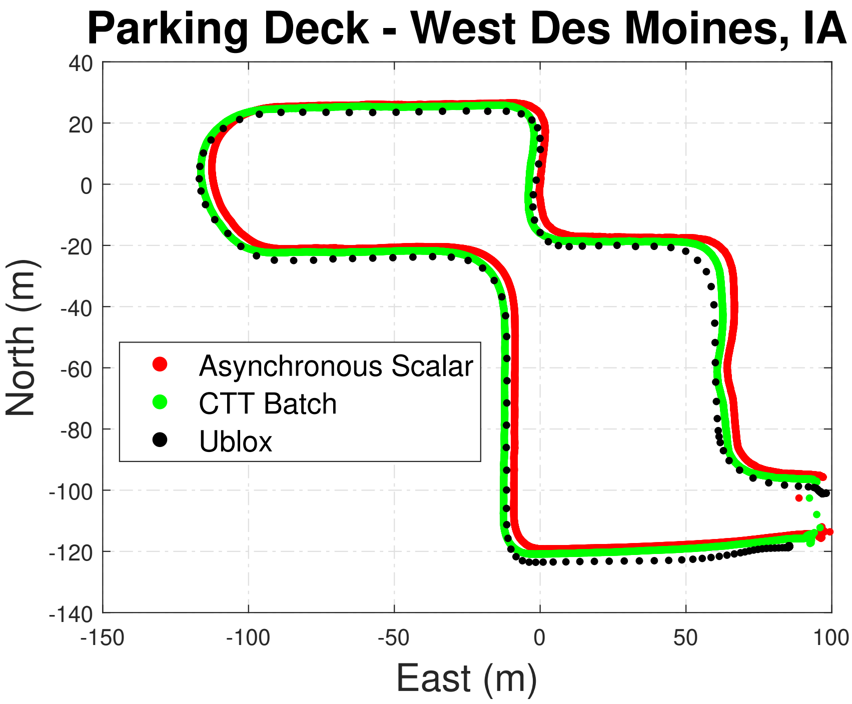

Figure B.8: GPS & GLONASS Vector Tracking Position Comparisons in the Parking Deck Experiment

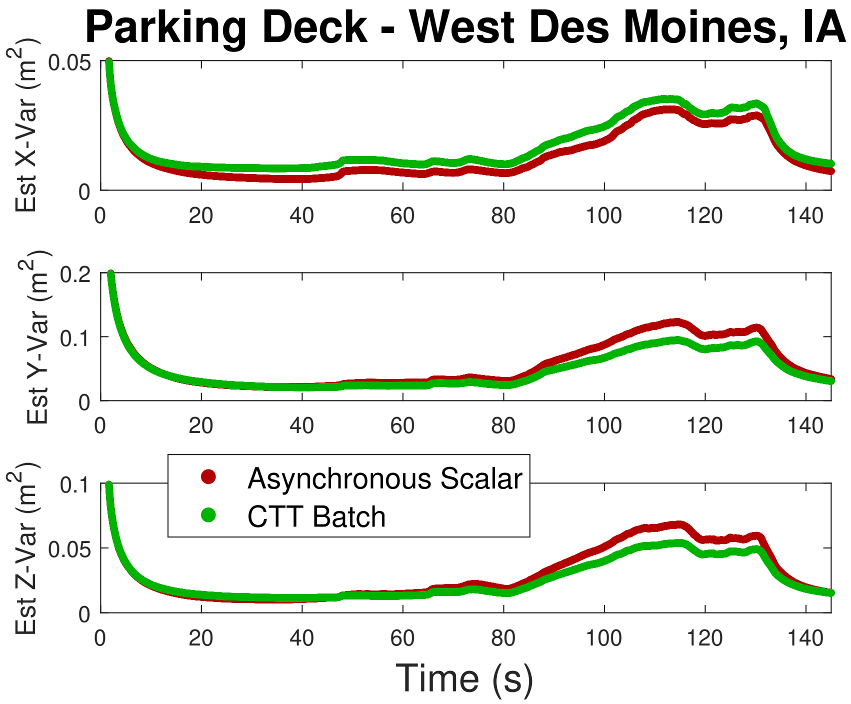

Figure B.9: GPS & GLONASS Vector Tracking Variance Comparisons in the Parking Deck Experiment

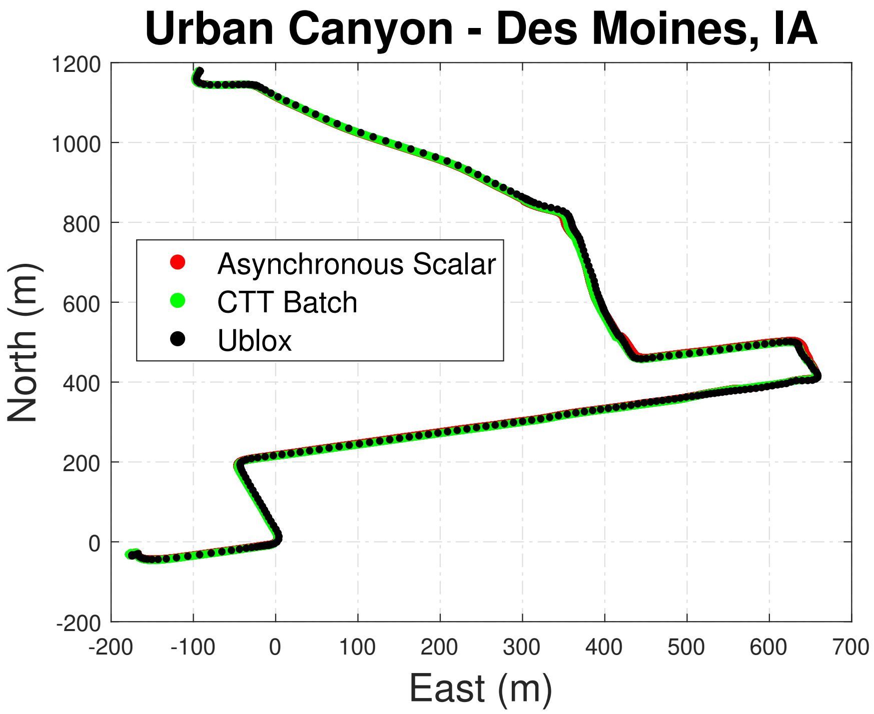

Figure B.10: GPS & GLONASS Vector Tracking Position Comparisons in the Urban Canyon Experiment

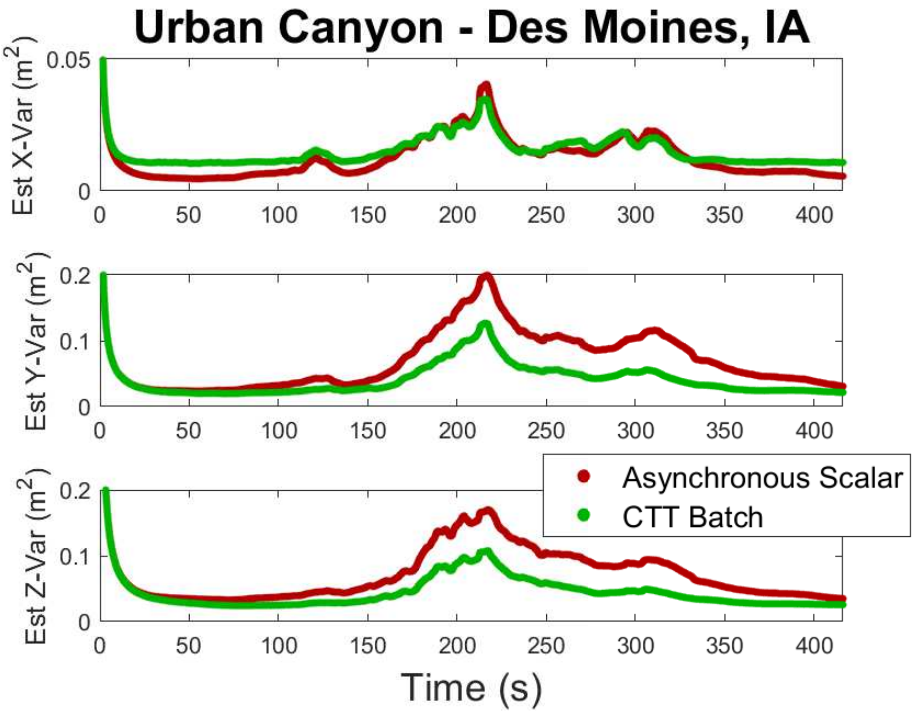

Figure B.11: GPS & GLONASS Vector Tracking Variance Comparisons in the Urban Canyon Experiment

Appendix C

Receiver Clock Bias Estimation Results

The GPS and GLONASS L1 vector tracking software-defined receiver operates with the IFEN SX3 hardware front-end in which both constellations' signal data sets are sampled by the same clock. This allows the receiver's navigation processor to estimate a single clock drift for GPS and GLONASS, but not necessarily a single clock bias. The GPS and GLONASS pseudorange measurements are generated on their own constellations' time systems, which will induce unique clock biases for each constellation. However, it was assumed that the clock bias difference between GPS and GLONASS is constant and small. In Chapter 6, the positioning results were generated with estimation of a single clock bias that was associated to both constellations. This appendix analyzes receiver clock bias and position results when the navigation processor estimates one clock bias for both constellations and when the navigation processor estimates two clock biases for each constellation.

The IFEN SX3 also has the ability to operate with external oscillators. The appendix shows experimental receiver clock bias and position results for different types of external clocks. In each each experiment, the software receiver uses GPS & GLONASS CTT batch vector tracking to perform the navigation. The same process noise covariance tuning is used when the Kalman filter estimates one or two clock bias states. Clock tuning in the process noise covariance matrix Q for each type of oscillator quality used the Allan variance coefficients in [54]. For additional information on clock error tuning, refer to Equations (5.20) and (5.21).

C.1 TCXO Results

Experimental signal data was collected with the internal TCXO clock in the IFEX SX3 front-end. The data set is associated with the light tree foliage experiment discussed in Chapter 6. Figure C.1 shows clock bias estimation results with the TCXO when the software receiver estimates two clock biases, one for each satellite system. The top plot of the figure shows the difference in clock bias between the GPS and GLONASS estimations. The bottom plot of the figure shows the EKF estimated clock bias variances. For the TCXO, the difference between the GPS and GLONASS clock biases is small (approximately a meter difference) and constant (slowly varying).

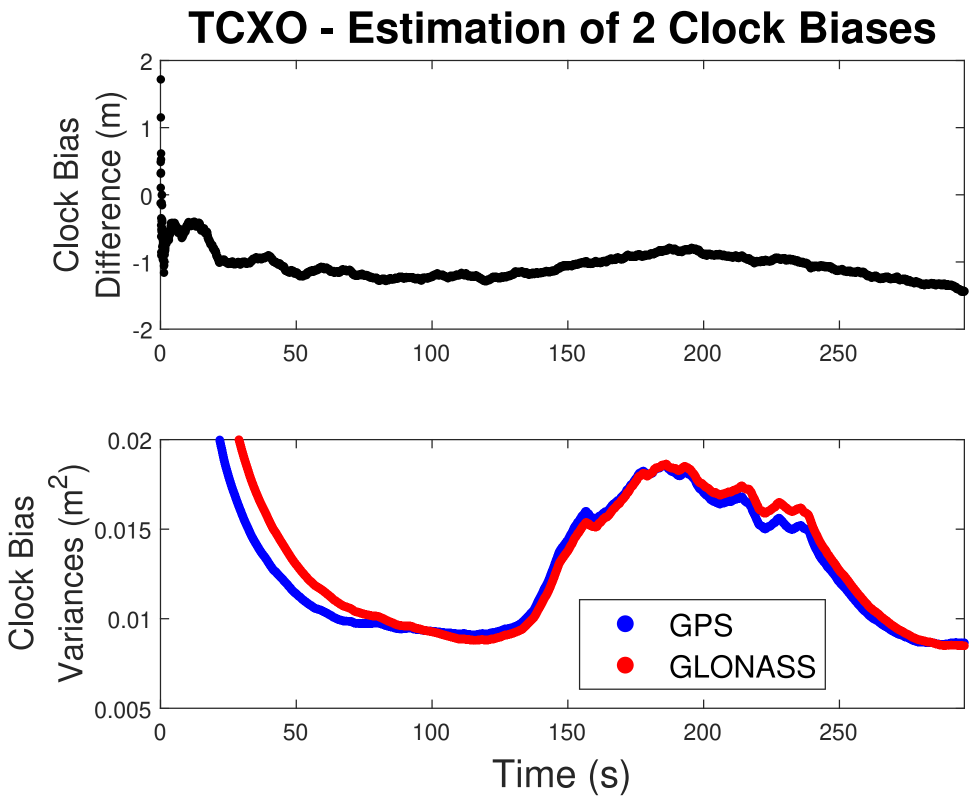

Figure C.1: TCXO Results when the Software Receiver Estimates Two Clock Biases

Figure C.2 shows clock bias estimation results with the TCXO when the software receiver estimates one clock bias for both satellite systems. The top plot of the figure shows the clock bias errors between the single (one bias for both constellations) and double (one bias for each constellation) estimations. The bottom plot of the figure shows the EKF estimated clock bias variance. The errors between the single and double clock bias estimations are small, but appear to be growing slowly. It is uncertain whether the clock bias errors will settle to a constant value.

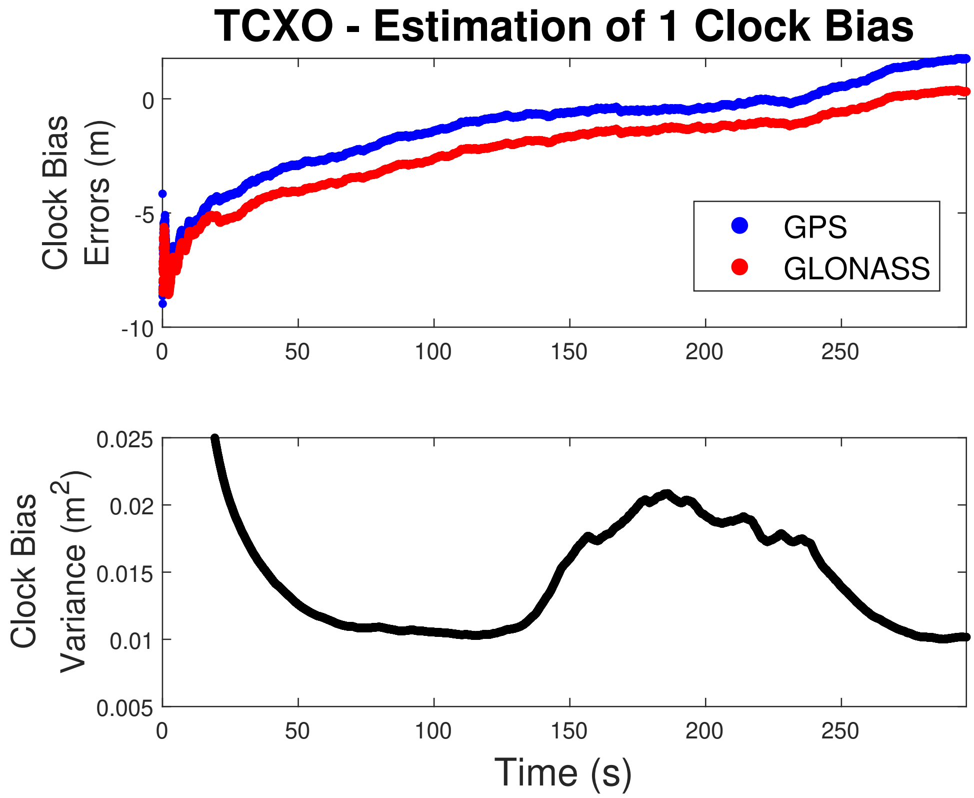

Figure C.2: TCXO Results when the Software Receiver Estimates One Clock Bias

Figure C.3 shows GPS & GLONASS positioning results for the receiver with the TCXO, when estimating one clock bias and two clock biases. The Ublox trajectory is shown for reference. The positioning results have approximately the same accuracy with respect to the Ublox when estimating one or two clock bias states. The positioning accuracy may be slightly better when estimating one clock bias state for both constellations. Because the clock bias difference between the constellations was small in Figure C.1, the position results do not change significantly when transitioning from one clock bias estimation to the other.

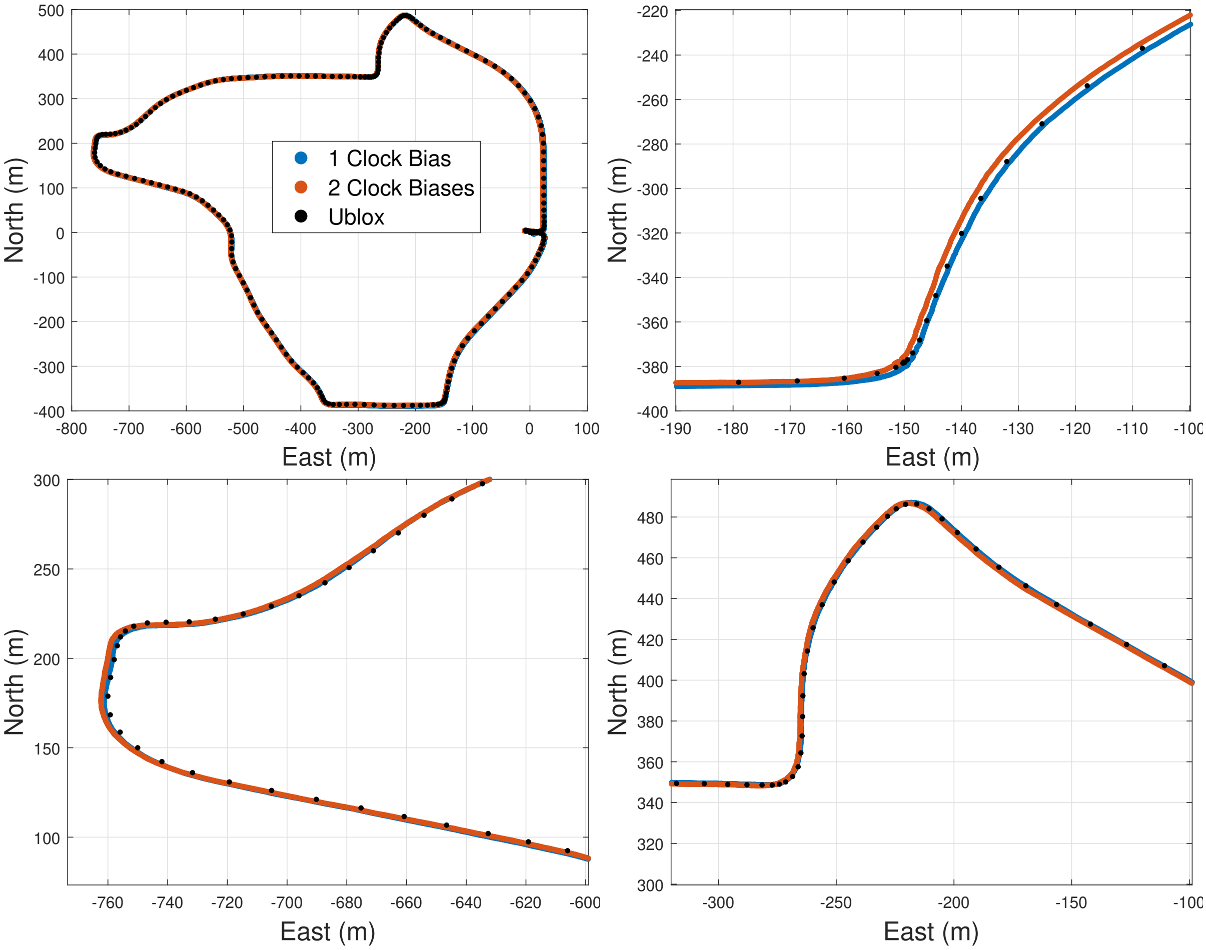

Figure C.3: Positioning Results when using the TCXO

C.2 OCXO Results

Experimental signal data was collected with a static antenna using an external OCXO aiding the IFEN SX3. Figure C.4 shows a picture of the OCXO, built by Ettus Research.

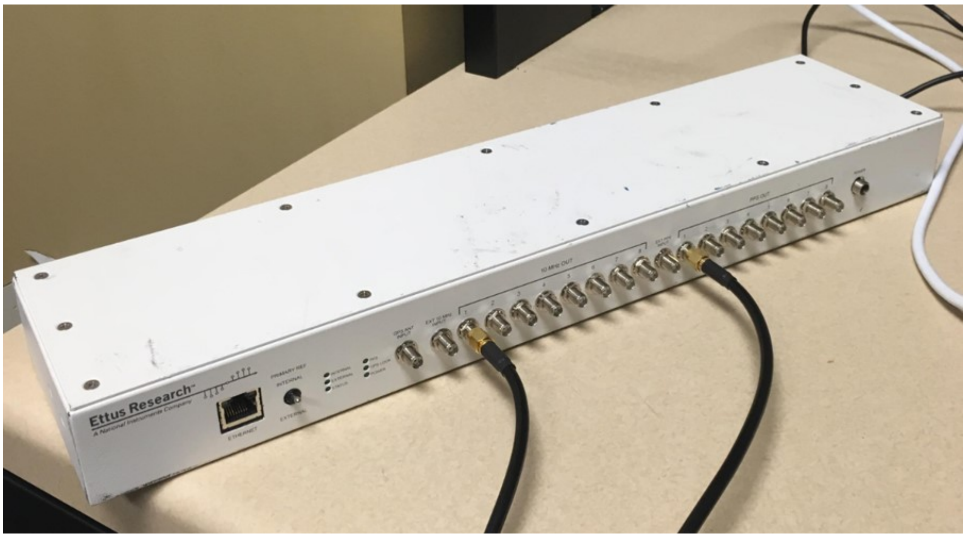

Figure C.4: Picture of the OCXO Tested in the Experiment

Figure C.5 shows clock bias estimation results with the OCXO, when the software receiver estimates two clock biases, one for each satellite system. The top plot of the figure shows the difference in clock bias between the GPS and GLONASS estimations. The bottom plot of the figure shows the EKF estimated clock bias variances. Like the TCXO, the OCXO clock bias difference between GPS and GLONASS is constant, but the magnitude of the difference is significantly larger.

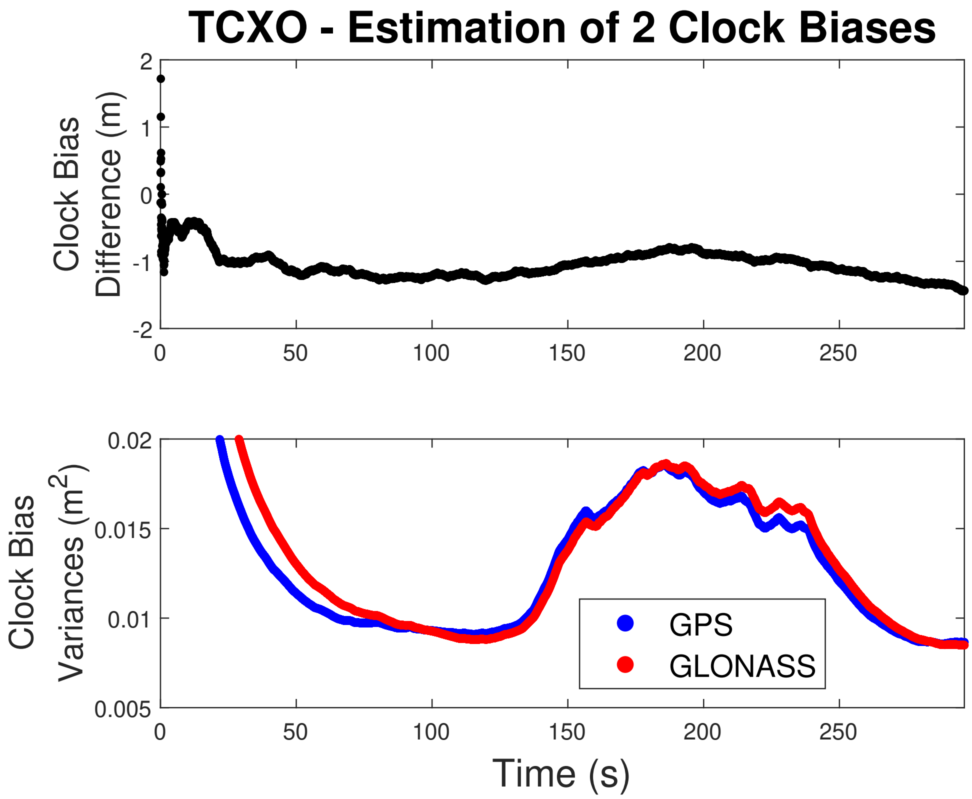

Figure C.5: OCXO Results when the Software Receiver Estimates Two Clock Biases

Figure C.6 shows clock bias estimation results with the OCXO, when the software receiver estimates one clock bias for both satellite systems. The top plot of the figure shows the clock bias errors between the single (one bias for both constellations) and double (one bias for each constellation) estimations. The bottom plot of the figure shows the EKF estimated clock bias variance. The errors between the single and double clock bias estimations are larger for the OCXO than the TCXO. This likely occurs because the difference between the GPS and GLONASS clock biases was larger for the OCXO in Figure C.5 than the TCXO in Figure C.1.

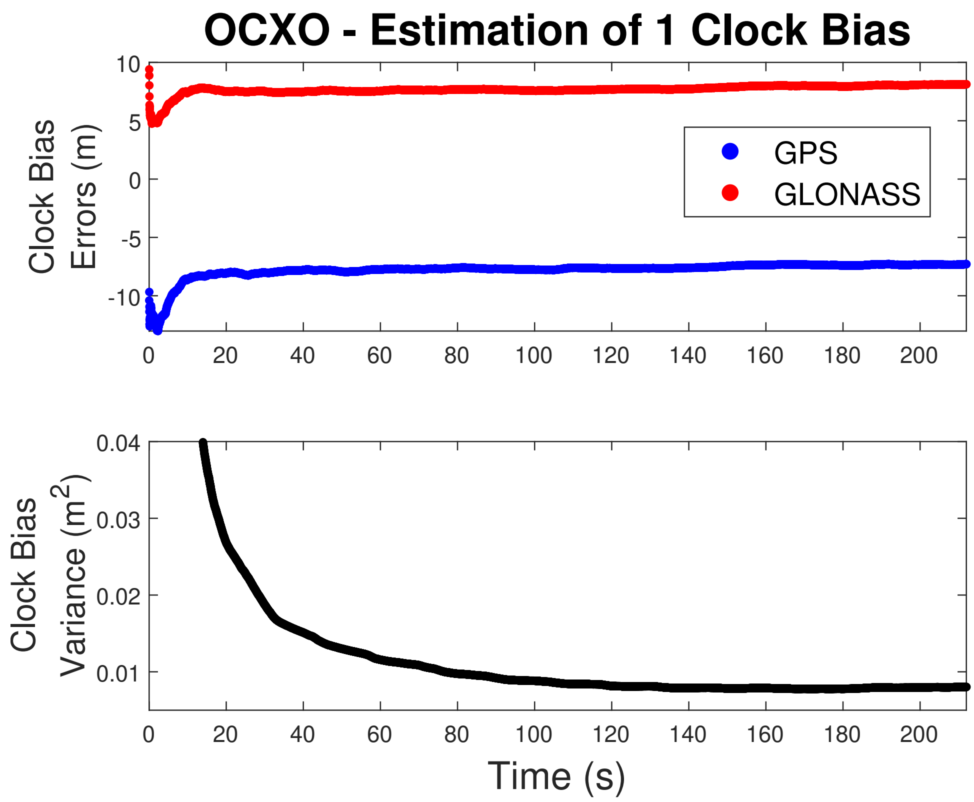

Figure C.6: OCXO Results when the Software Receiver Estimates One Clock Bias

Figure C.7 shows GPS & GLONASS positioning results for the receiver with the rubidium clock, when estimating one clock bias and two clock biases. The positioning results have approximately the same precision and accuracy. There appears to be a translational bias between the estimations, which likely comes from the clock bias difference between the GPS and GLONASS constellations seen in Figure C.5.

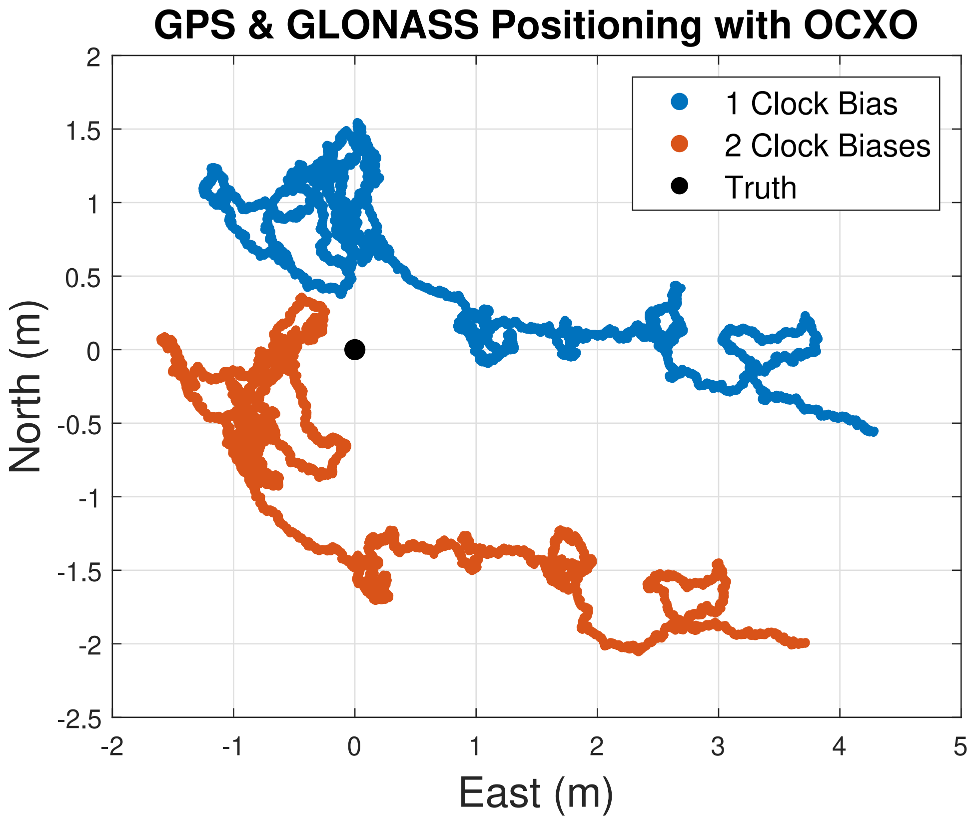

Figure C.7: Positioning Results when using the OCXO

C.3 Rubidium Results

Experimental signal data was collected with a static antenna using an external rubidium atomic clock aiding the IFEN SX3. Figure C.8 shows a picture of the rubidium atomic clock, built by Stanford Research Systems.

Figure C.8: Picture of the Rubidium Atomic Clock Tested in the Experiment

Figure C.9 shows clock bias estimation results with the rubidium atomic oscillator when the software receiver estimates two clock biases, one for each satellite system. The top plot of the figure shows the difference in clock bias between the GPS and GLONASS estimations. The bottom plot of the figure shows the EKF estimated clock bias variances. The clock bias difference for the rubidium oscillator is similar to the difference shown for the OCXO in Figure C.5.

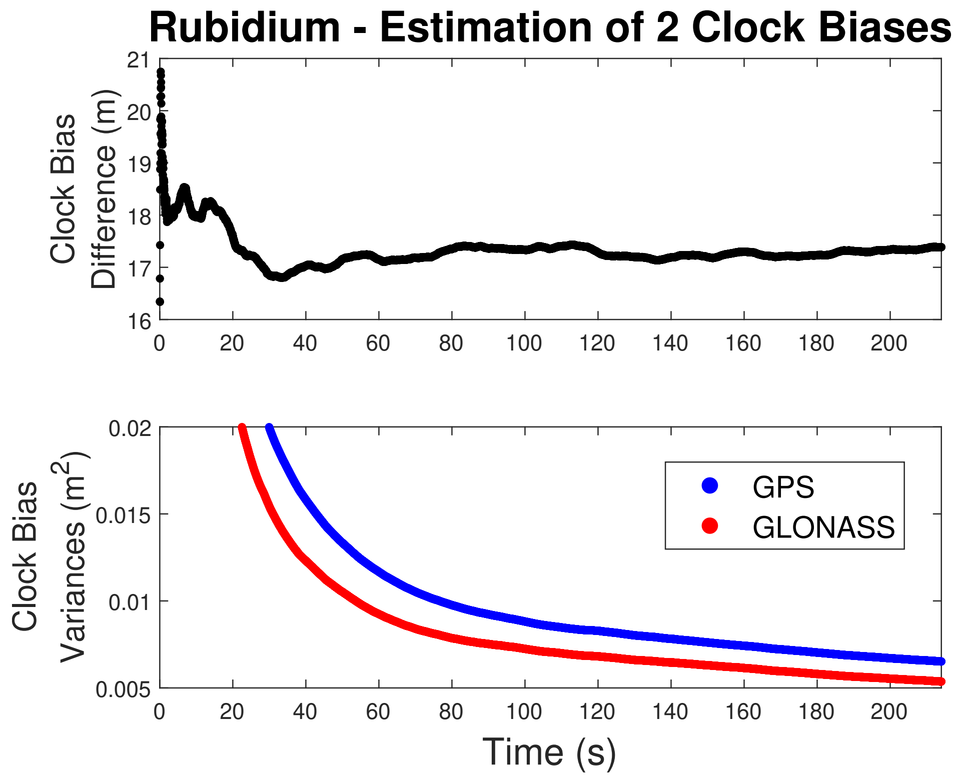

Figure C.9: Rubidium Results when the Software Receiver Estimates Two Clock Biases

Figure C.10 shows clock bias estimation results with the rubidium atomic oscillator when the software receiver estimates one clock bias for both satellite systems. The top plot of the figure shows the clock bias errors between the single (one bias for both constellations) and double (one bias for each constellation) estimations. The bottom plot of the figure shows the EKF estimated clock bias variance. The errors between the single and double clock bias estimations are similar to those for the OCXO in Figure C.6 for the same purposes.

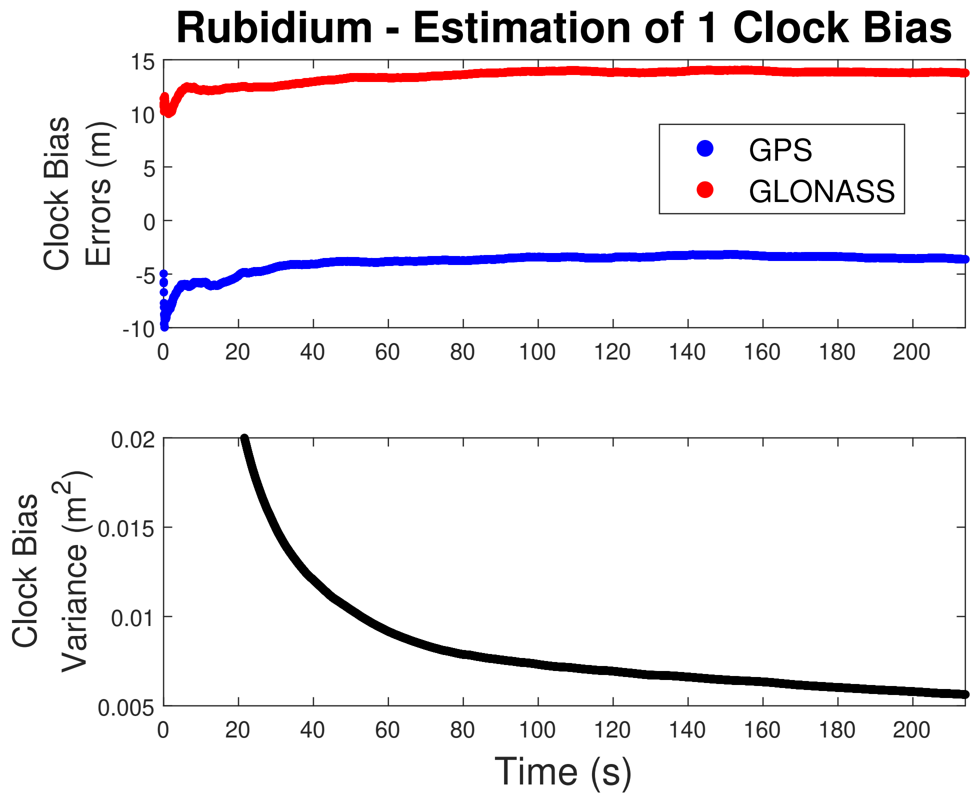

Figure C.10: Rubidium Results when the Software Receiver Estimates One Clock Bias

Figure C.11 shows GPS & GLONASS positioning results for the receiver with the rubidium clock, when estimating one clock bias and when estimating two clock biases. The positioning results have approximately the same precision. The single clock bias estimation is slightly more accurate than the double clock bias method. Like the OCXO position estimates in Figure C.7, a translational bias is present between the single and double receiver clock bias methods for the rubidium clock

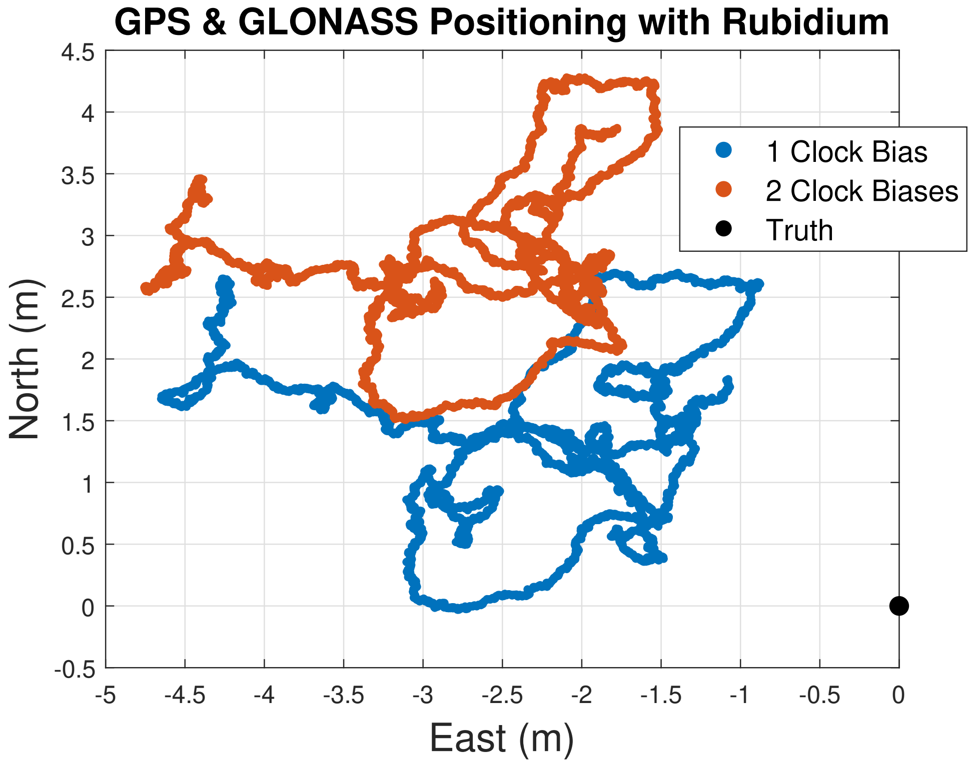

Figure C.11: Positioning Results when using the Rubidium Atomic Clock

C.4 Cesium Results

Experimental signal data was collected with a static antenna using an external cesium atomic clock aiding the IFEN SX3. Figure C.12 shows a picture of the cesium atomic clock. This cesium oscillator is a Chip Scale Atomic Clock (CSAC).

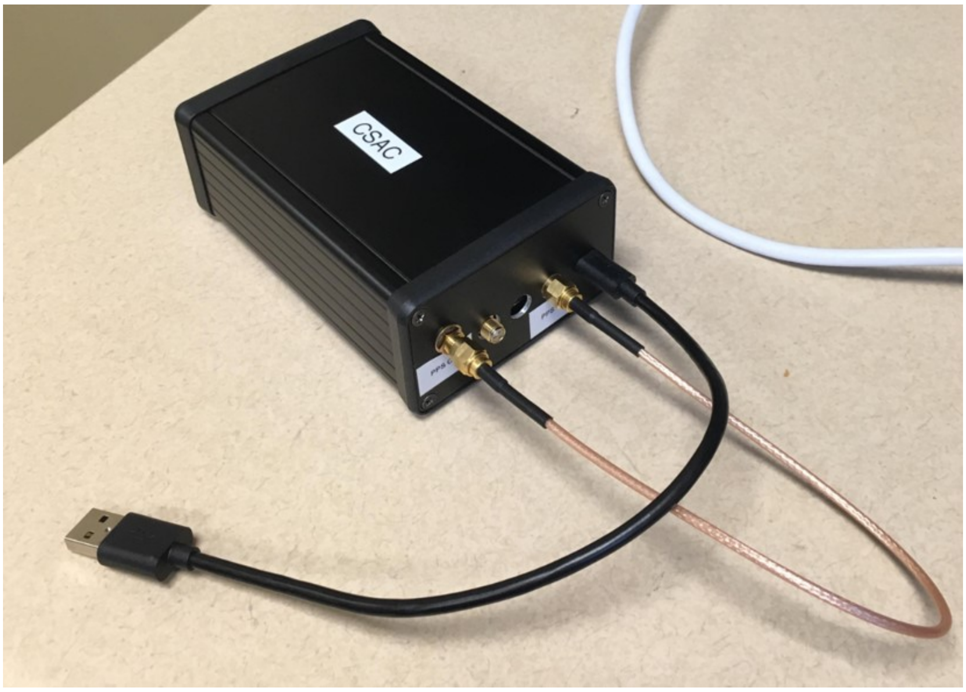

Figure C.12: Picture of the Cesium Atomic Clock Tested in the Experiment

Figure C.13 shows clock bias estimation results with the CSAC when the software receiver estimates two clock biases, one for each satellite system. The top plot of the figure shows the difference in clock bias between the GPS and GLONASS estimations. The bottom plot of the figure shows the EKF estimated clock bias variances. The clock bias difference between GPS and GLONASS becomes even larger for the CSAC than the rubidium clock or OCXO.

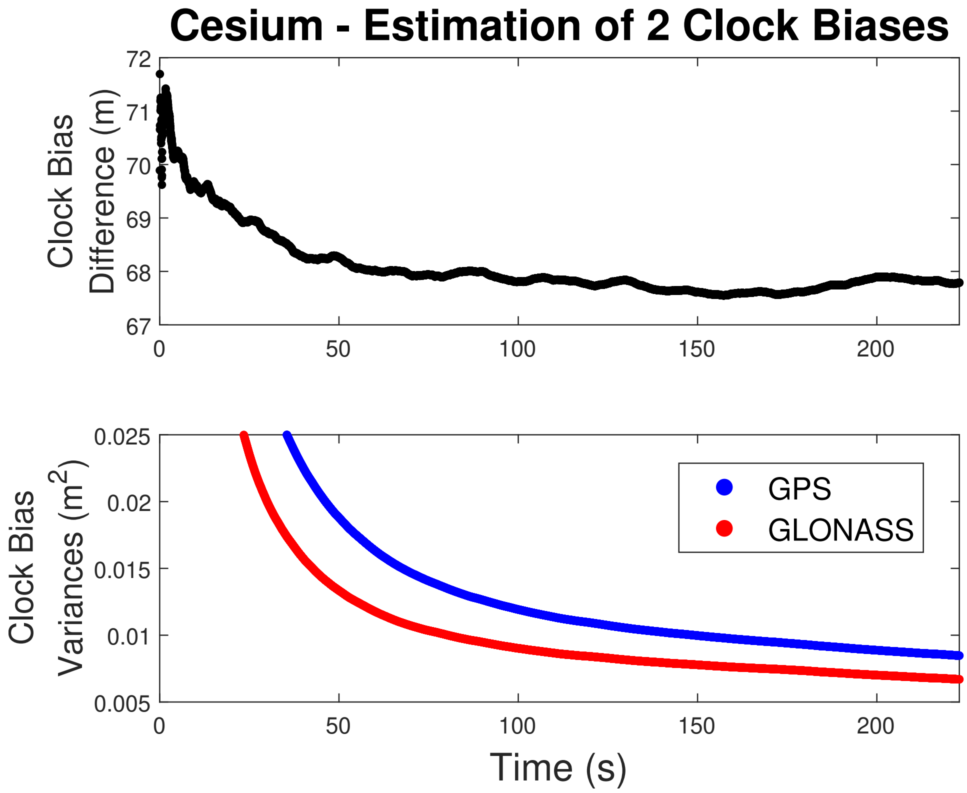

Figure C.13: CSAC Results when the Software Receiver Estimates Two Clock Biases

Figure C.14 shows clock bias estimation results with the CSAC when the software receiver estimates one clock bias for both satellite systems. The top plot of the figure shows the clock bias errors between the single (one bias for both constellations) and double (one bias for each constellation) estimations. The bottom plot of the figure shows the EKF estimated clock bias variance. Because the clock bias difference between GPS and GLONASS is large in the CSAC, this results in substantial errors between the single and double estimations.

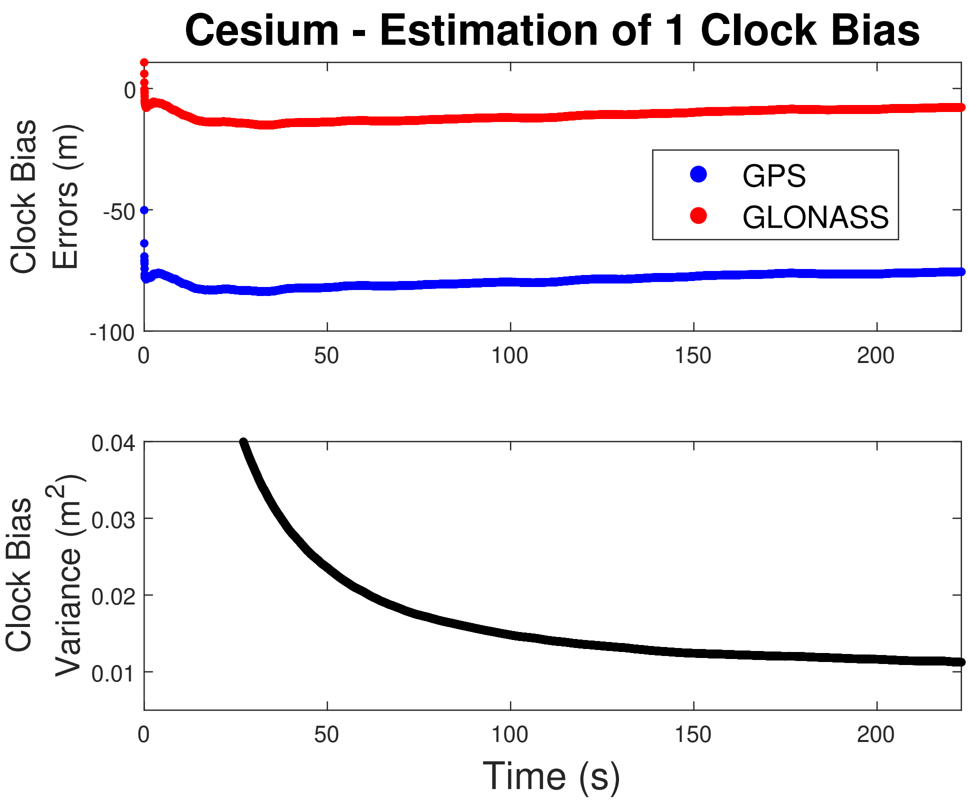

Figure C.14: CSAC Results when the Software Receiver Estimates One Clock Bias

Figure C.15 shows GPS & GLONASS positioning results for the receiver with the CSAC, when estimating one clock bias and when estimating two clock biases. The positioning results are more accurate when the software receiver estimates separate clock biases for GPS and GLONASS. Because the clock bias difference between the constellations was large, shown in Figure C.13, the positioning estimation becomes significantly biased when only one receiver clock bias is estimated for both constellations.

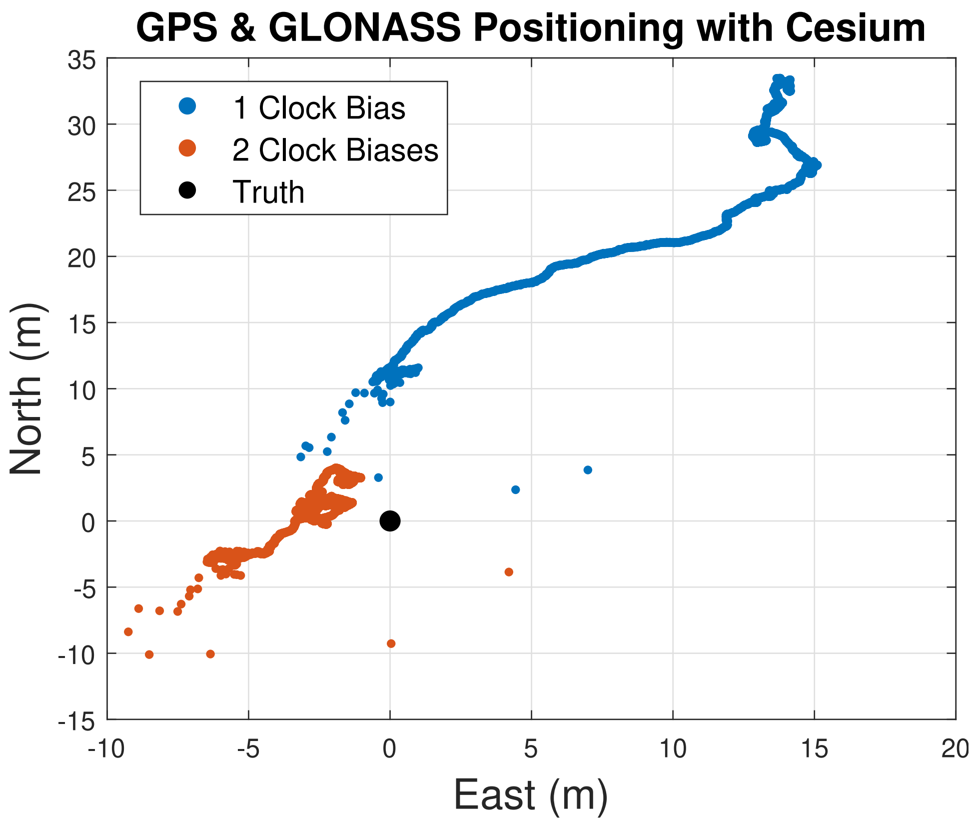

Figure C.15: Positioning Results when using the CSAC

C.5 Summary of Results

The difference in clock bias between GPS and GLONASS varied for the different types of oscillators. The clock bias difference was not significant for the TCXO clock, which allowed for accurate positioning results in the software receiver when estimating only one clock bias for both constellations. The clock bias difference became more significant for the OCXO and rubidium clocks. The clock bias difference between satellite systems was larger in the OCXO and rubidium than the TCXO, but small enough that acceptable positioning could be performed with estimation of one clock bias. The clock bias difference between satellite systems was larger than the OCXO and rubidium in the CSAC, and large enough that positioning became significantly degraded when only one clock bias was estimated.