

DEVELOPMENT AND ASSESSMENT OF LOOSELY-COUPLED INS USING SMARTPHONE SENSORS

EDUARDO INFANTE

October 2016



**TECHNICAL REPORT
NO. 305**

DEVELOPMENT AND ASSESSMENT OF LOOSELY-COUPLED INS USING SMARTPHONE SENSORS

Eduardo Infante

Department of Geodesy and Geomatics Engineering
University of New Brunswick
P.O. Box 4400
Fredericton, N.B.
Canada
E3B 5A3

October 2016

© Eduardo Infante, 2016

PREFACE

This technical report is a reproduction of a thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Engineering in the Department of Geodesy and Geomatics Engineering, October 2016. The research was supervised by Dr. Richard Langley.

As with any copyrighted material, permission to reprint or quote extensively from this report must be received from the author. The citation to this work should appear as follows:

Infante, Eduardo (2016). *Development and Assessment of Loosely-Coupled INS using Smartphone Sensors*. M.Sc.E. thesis, Department of Geodesy and Geomatics Engineering Technical Report No. 305, University of New Brunswick, Fredericton, New Brunswick, Canada, 195 pp.

ABSTRACT

Smartphone accelerometers and gyroscopes are quite common in today's society but little work has been done on assessing how accurate and reliable they are to be used in inertial navigation systems (INS). The goal of this research is to develop a loosely-coupled INS filter that only uses sensors found inside a Moto-X Android smartphone. Micro-electro-mechanical sensors (MEMS) accelerometers and gyroscopes provide the raw motion sensor data whereas the high-sensitivity GNSS receiver in the smartphone is used to provide position and velocity updates to the filter. Magnetometers, also included in the MEMS are a potential source of heading aiding that not only aids in INS alignment but helps constrain the heading drift. A successful filter implementation could potentially open the doors of inertial navigation to the everyday smartphone user. This would allow developers of smartphone applications to focus on the creative side of their application while using the loosely-coupled INS in the background.

The loosely-coupled INS filter was developed in C++ and was run offline although the operations are exactly those that would be applied in real time. The INS filter was verified by using raw inertial measurement unit (IMU) measurements from a high-end Northrop Grumman IMU-LN200 motion sensor and single-point GNSS position/velocity updates from a high accuracy NovAtel Flexpak6 receiver. Two datasets with distinct environments were used. The first one was a relatively open-sky dataset in NW Calgary and the second was an urban canyon dataset in downtown Calgary. Once the INS was verified to work within expectations, two more datasets were collected, this time with the Moto-X Android smartphone and the NovAtel SPAN system (IMU-LN200

+ Flexpak6 running INS capable firmware). The datasets were again in open-sky and urban canyon environments. Due to the high noise of the Moto-X sensors, the high frequency noise of the raw data was removed via wavelet decomposition. This was very important as the faint sensor signal is buried under a lot of noise. Empirically derived estimates for sensor turn-on bias and scale factor errors were then found.

The easiest way to assess the validity of the filter is to compare the attitude with the truth trajectory, where the truth trajectory is that of the NovAtel SPAN solution. The reason for this is that position and velocity are directly dependent on the quality of input filter updates. It is possible to have good results in position and velocity but still have a filter that diverges in attitude. When ran with the IMU-LN200 and NovAtel Flexpak6 data, the loosely-coupled INS filter had RMS differences in pitch and roll under 0.4° in the open-sky dataset and under 0.8° in the urban canyon dataset. RMS differences in heading were below 1° in the open-sky dataset and slightly above 1° in the urban canyon dataset. When ran with the Moto-X Android smartphone sensors, the INS filter had RMS differences in pitch and roll below 4.5° in the open-sky dataset and below 16° in the urban canyon datasets respectively. The RMS differences in heading were around 13° for the open-sky dataset and large enough to make the system useless for the urban canyon dataset. The results show the Moto-X Android smartphone sensors can be used for civilian enthusiast level of navigation under open-sky environments. It is however expected for MEMS sensors to improve over time thus improving the usability of a loosely-coupled INS filter using smartphone sensors.

DEDICATION

For my wife Priscila, and my parents.

ACKNOWLEDGEMENTS

I would like to acknowledge the following people who have helped and supported me throughout the duration of this thesis:

- Dr. Richard Langley for having taken me under his wings with arms wide open ever since I first stepped into the UNB campus looking for an opportunity to enroll in a Master's degree in the Department of Geodesy and Geomatics Engineering. It has been quite an honour to work under you. Thank you very much for all your patience, guidance and wealth of knowledge.
- Ryan Dixon, Mike Bobye, and Sam Kiley-Kubik at the SPAN team at NovAtel Inc. When I started working at NovAtel I had a vague idea of what I wanted to do. It was your patience and industry-leading expertise that helped me build a solid theoretical INS foundation. Thank you very much for always being willing and available to clarify doubts. Thanks for the printed references Sam!
- NovAtel Inc. for having given me the opportunity to work with the industry leaders in navigation. This thesis would not have been possible without the GNSS and INS knowledge learned every day on the job.
- Google Inc. for Google Earth. It made analysis so much easier!
- Jeremy Nicholson at CARIS for allowing me to attend commitments related to this degree during work hours.
- Jonathan Stuart, Burns Foster, Jennifer McMullen, Jamie Parsons for the continuing support and encouragement throughout the good times but more

importantly the bad times when quitting seemed the obvious answer. You guys were always there for me.

- Sylvia Whitaker for being so helpful throughout the time it took to get through this program. I can't thank you enough for all the help regardless of whether I was in Fredericton or Calgary. You are the best!
- My parents who since day one have provided me with immeasurable support. You were always willing to listen to me whenever I had doubts and frustrations regarding the possibility of achieving this degree part-time while working full-time in the industry. There are no words that can express how thankful I am for everything you have done in my life.
- My wife Priscila for the patience and support. You were the one who had to endure all of my frustration when filters diverged, and when I kept on going in circles trying to debug code. Obrigado pelo todo teu amor e paciência!

TABLE OF CONTENTS

ABSTRACTii
DEDICATIONiv
ACKNOWLEDGEMENTSv
LIST OF TABLESx
LIST OF FIGURESxi
LIST OF SYMBOLS AND ABBREVIATIONSxiv
CHAPTER 1: INTRODUCTION	1
1.1 Background	2
1.1.1 Smartphone Sensors	2
1.1.2 INS	3
1.1.3 GNSS Outages in INS	5
1.1.4 INS Integration Methods	6
1.1.5 INS Alignment	6
1.2 Previous Research and Limitations	7
1.3 Contributions	8
1.4 Outline	9
CHAPTER 2: OVERVIEW OF GNSS AND INS	11
2.1 GNSS Measurements	13
2.1.1 Pseudorange Measurements	13
2.1.2 Carrier-Phase Measurements	15
2.1.3. Doppler Measurement	17
2.2 Error Sources in GNSS and How to Mitigate Them	18
2.2.1 Ionospheric Errors	20
2.2.2 Troposphere Errors	21
2.2.3 Orbital Errors	21
2.2.4 Satellite and Receiver Clock Errors	22
2.2.5 Multipath	22
2.2.6 Measurement Noise	23
2.3 High Sensitivity Assisted GNSS	25
2.4 Coordinate Frames Used in Inertial Navigation	27
2.5 Coordinate Transformation	32

2.6 Mechanization Equations	34
2.7 Inertial Sensor Errors	43
2.7.1 Sensor Bias Offset.....	45
2.7.2 Scale Factor Error	46
2.7.3 Non-orthogonality of the axes.....	47
2.7.4 Sensor noise	47
2.8 Classification of IMUs	48
2.9 Initial Alignment.....	51
CHAPTER 3: INTRODUCTION TO DATA PROCESSING METHODS	56
3.1 Estimation using Measurements Only	57
3.2 Estimation using Dynamic Modelling	60
3.3 Kalman Filter Algorithm.....	62
3.4 Kalman Filter in Non-Linear Systems	66
CHAPTER 4: GNSS/INS INTEGRATION	70
4.1 Integration Strategies	70
4.1.1 Tight Coupling	71
4.1.2 Loose Coupling	72
4.2 Effect of Lever Arm in INS Solution.....	74
4.3 INS Filter	75
4.3.1 System Model	75
4.3.2 Measurement Model	84
4.4 External Aided Heading.....	85
4.5 Motion Constraints.....	88
4.6 Velocity Constraints.....	89
4.7 Height Constraints	91
CHAPTER 5: FILTER VERIFICATION THROUGH TRUTH TESTING	95
5.1 IMU-LN200	95
5.2 Other Equipment Used.....	98
5.3 Procedure for Acquisition used for Truth Testing	100
5.4 Static Test.....	101
5.5 Open-sky Kinematic Truth Test.....	106
5.6 Urban Canyon Kinematic Truth Test.....	113

CHAPTER 6: PROCEDURES AND RESULTS	124
6.1 Sensors inside Moto-X smartphone	124
6.2 Procedure	129
6.2.1 Prior to Acquisition.....	129
6.2.1 During Acquisition.....	131
6.2.3. After Acquisition: Discrete Wavelet Decomposition.....	134
6.3 Magnetometer Assessment	142
6.4 Open-sky Results	146
6.5 Urban Canyon	159
CHAPTER 7: CONCLUSIONS AND RECOMMENDATIONS	172
7.1 Conclusions.....	173
7.2 Recommendations.....	177
REFERENCES	180
APPENDIX A: DYNAMICS AND MEASUREMENT MODELS FOR INS ERROR STATES IN THE E-FRAME	186
APPENDIX B: SMARTPHONE MAGNETOMETER CALIBRATION.....	192
CURRICULUM VITAE.....	1

LIST OF TABLES

Table 2-1: Characteristics and magnitudes of GNSS errors (adapted from Godha (2006), Gao (2007), O'Keefe (2005), and Langley (2016)).....	24
Table 2-2: IMU grade comparisons (adapted from NovAtel Inc, 2015e).....	49
Table 5-1: IMU-LN200 characteristics.....	96
Table 5-2: NovAtel SPAN + IMU-LN200 (NovAtel Inc, 2015e)	97
Table 5-3: Differences in attitude between author-generated INS and SPAN.....	106
Table 5-4: Differences between author-generated INS and SPAN in open-sky test	112
Table 5-5: Urban canyon sections.....	114
Table 5-6: Differences between author-generated INS and SPAN in the urban canyon test....	121
Table 5-7: RMS of differences between author-generated INS and SPAN in the urban canyon test.....	122
Table 6-1: Manufacturer-reported characteristics of Moto-X smartphone sensors	125
Table 6-2: Moto-X INS sensor characteristics used	126
Table 6-3: IMU operating rates in Moto-X Android smartphone	133
Table 6-4: Magnetometer calibration results	143
Table 6-5: Start, end epochs and cumulative time for open-sky test	150
Table 6-6: Differences between Moto-X and SPAN in open sky	157
Table 6-7: RMS of differences between Moto-X and SPAN in open sky	158
Table 6-8: Start, end epochs and cumulative time for urban canyon test	167
Table 6-9: RMS of differences between Moto-X and SPAN in urban canyon.....	171

LIST OF FIGURES

Figure 2-1: GNSS overview (NovAtel Inc, 2015d)	11
Figure 2-2: Carrier phase observation.....	16
Figure 2-3: Antennas in multipath environment (NovAtel Inc, 2015b).	23
Figure 2-4: ECEF frame (e-frame) and local level frame (l-frame).....	29
Figure 2-5: Vehicle frame (NovAtel Inc, 2015g).....	30
Figure 2-6: IMU enclosure frame	31
Figure 2-7: Visual of e-frame to l-frame conversion	33
Figure 2-8: e-frame INS mechanization (adapted from Godha, 2006)	42
Figure 2-9: Visual of bias offset, noise and scale factor	44
Figure 2-10: Bias drift.....	45
Figure 3-1: Discrete Kalman filter (adapted from El-Sheimy, 2012)	66
Figure 3-2: INS extended Kalman filter block diagram (adapted from El-Sheimy, 2012).....	68
Figure 4-1: Tightly coupled integration (adapted from El-Sheimy, 2012)	72
Figure 4-2: Loosely coupled integration (adapted from El-Sheimy, 2012)	73
Figure 4-3: Velocity constraints (adapted from El-Sheimy, 2012).....	89
Figure 4-4: Height constraints (adapted from El-Sheimy, 2012).....	91
Figure 4-5: Motion constraints (adapted from El-Sheimy, 2012).....	93
Figure 5-1: IMU-LN200 strapped to vehicle	98
Figure 5-2: Equipment used for truth testing	100
Figure 5-3: Overview of truth tests in Calgary (Google, 2015)	101
Figure 5-4: Static test environment (Google, 2015)	102
Figure 5-5: Differences in position for static test.....	103
Figure 5-6: Standard deviations of single point position	103
Figure 5-7: Pitch and roll during static test.....	104
Figure 5-8: Heading during static test.....	105
Figure 5-9: Difference in attitude with respect to truth in static test.....	105
Figure 5-10: Open-sky kinematic truth test (Google, 2015)	107
Figure 5-11: Sections in open-sky truth testing (Google, 2015)	108
Figure 5-12: Position difference between SPAN and author-generated INS (open sky)	108
Figure 5-13: Velocity difference between SPAN and author-generated INS (open sky)	109
Figure 5-14: Standard deviations of pseudorange position updates fed into INS (open sky) ...	110
Figure 5-15: Standard deviation of velocity updates fed into INS (open sky)	110
Figure 5-16: Satellites tracked (open-sky)	110
Figure 5-17: Attitude difference between SPAN and author-generated INS (open sky)	111
Figure 5-18: Height differences between raw pseudorange position and SPAN (open sky)	113
Figure 5-19: Urban canyon in downtown Calgary (Google, 2015)	114
Figure 5-20: Urban canyon downtown core (Google, 2015)	115

Figure 5-21: Loosely-coupled INS (top) vs. SPAN tightly-coupled INS (bottom) (Google, 2015)	116
Figure 5-22: Position difference between SPAN and author-generated INS (urban canyon)....	117
Figure 5-23: Velocity difference between SPAN and author-generated INS (urban canyon)...	118
Figure 5-24: Attitude difference between SPAN and author-generated INS (urban canyon)....	119
Figure 5-25: Standard deviation of pseudorange position updates fed into INS (urban canyon)	120
Figure 5-26: Standard deviation of velocity updates fed into INS (urban canyon)	120
Figure 5-27: Satellites tracked (urban canyon).....	120
Figure 6-1: Moto-X smartphone location	132
Figure 6-2: Location of Flexpak6 and IMU-LN200	132
Figure 6-3: Wavelet multiple level of decomposition (adapted from Nassar, 2003)	135
Figure 6-4: Amplitude spectrum for raw x-axis accelerometer (open sky)	136
Figure 6-5: First 5 LOD for raw x-axis accelerometer (open sky)	139
Figure 6-6: Original and filtered x-axis accelerometer (open sky)	140
Figure 6-7: Standard deviations of approximations for each LOD	141
Figure 6-8: Comparison of original and filtered Δt signal	142
Figure 6-9: Heading from magnetometer compared to SPAN (open sky).....	144
Figure 6-10: Heading from magnetometer compared to SPAN (Urban Canyon).	145
Figure 6-11: Open-sky dataset with Moto-X (Google, 2015).....	147
Figure 6-12: Section 1 & Section 2 of open-sky dataset (Google, 2015)	148
Figure 6-13: Section three of open-sky dataset (Google, 2015)	148
Figure 6-14: Section four of open-sky dataset (Google, 2015).....	149
Figure 6-15: Section five of open-sky test (Google, 2015).....	150
Figure 6-16: Difference in position with respect to SPAN (open-sky).....	151
Figure 6-17: GNSS position update standard deviation (open sky).....	152
Figure 6-18: Difference in velocity with respect to SPAN (open sky)	153
Figure 6-19: GNSS velocity update standard deviation (open sky).....	154
Figure 6-20: Difference in attitude with respect to SPAN (open sky)	154
Figure 6-21: Heading comparison (open-sky)	156
Figure 6-22: Urban canyon dataset overview with Moto-X sensors (Google, 2015)	160
Figure 6-23: First section of urban canyon dataset (Google, 2015).....	161
Figure 6-24: Section two of urban canyon dataset (Google, 2015)	161
Figure 6-25: Section two looking west (Google, 2015).....	162
Figure 6-26: Section two looking southeast (Google, 2015)	163
Figure 6-27: Section three of urban canyon dataset (Google, 2015)	163
Figure 6-28: Start of section three looking southeast (Google, 2015)	164
Figure 6-29: Plan view of start of section three (Google, 2015).....	164
Figure 6-30: Last part of section three looking westward (Google, 2015)	165
Figure 6-31: Effect of the Bow (Google, 2015).....	166
Figure 6-32: Section four of urban canyon dataset (Google, 2015).....	166
Figure 6-33: Difference in position with respect to SPAN (urban canyon).....	167

Figure 6-34: GNSS position update standard deviation (urban canyon)	168
Figure 6-35: Difference in velocity with respect to SPAN (urban canyon).....	168
Figure 6-36: GNSS velocity update standard deviation (urban canyon)	169
Figure 6-37: Difference in attitude with respect to SPAN (urban canyon).....	169

LIST OF SYMBOLS AND ABBREVIATIONS

Symbols

<i>bar</i>	...Averaged quantity
<i>dots</i>	...Time derivative
<i>hat</i>	...Estimated quantity
\sim	...Measured value by the sensor
*	...Nominal state vector value
()(<i>t</i>)	...Quantity as a function of time
<i>k</i> ()	...Quantity at <i>k</i> th epoch
() ⁻	...Predicted quantity
() ⁺	...Updated quantity
() _{<i>q</i>}	...Quantity in <i>q</i> -frame,
() _{<i>a</i>}	...Accelerometer specific quantity
() _{<i>g</i>}	...Gyro specific quantity
$\frac{\partial}{\partial X}$...Partial derivative with respect to <i>X</i>
<i>b-frame</i>	...INS Computation frame
<i>e-frame</i>	...ECEF frame
<i>i-frame</i>	...Inertial frame
<i>l-frame</i>	...Local level frame
<i>v-frame</i>	...Vehicle frame
<i>E</i> ^{<i>e</i>}	...Skew-symmetric matrix of misalignment error states
<i>F</i>	...System dynamic matrix
<i>G</i>	...Shaping matrix
<i>H</i>	...Design matrix

I	...Identity matrix
K	...Kalman gain matrix
N	...Prime vertical radius
N	...Tensor of gravity gradients
P	...Error covariance matrix associated with state vector
$Q(t)$...Spectral density matrix
Q	...Process noise covariance matrix
R	...Measurement noise covariance matrix
R_p^q	...Rotation matrix from the p -frame to the q -frame
S	...Scale-factor errors of sensor X
X_b -turn-on	...Turn-on bias of sensor X
c	...Speed of light in vacuum (2.99792458×10^8 m/s)
d_{ion}	...Ionospheric error
d_{orb}	...Satellite orbital error
d_{trop}	...Tropospheric error
dt	...Satellite clock error
dT	...Receiver clock error
e	...Eccentricity
f	...Specific force
	...System model non-linear function
h	...Ellipsoidal height
	...Measurement model non-linear function
p	...Pseudorange
q_i	...Quaternions $i = 1 \dots 4$
	...Spectral density of i

r	...Position vector (x_e, y_e, z_e)
v	...Velocity vector (v_x, v_y, v_z)
w	...Angular rate
w	...Process Noise
x	...State vector
z	...Observation vector
∂b_x	...Bias drift of sensor X
ψ	...Heading
φ	...Latitude
λ	...Longitude
ε	...Misalignment error state vector ($x, y, z, \varepsilon_x, \varepsilon_y, \varepsilon_z$)
δ	...Perturbation from the nominal value
η	...Pitch
η_X	...Random noise of system X
ζ	...Roll
ω_{pq}^r	...Rotation rate of the q -frame, relative to the p -frame, expressed ...in the r -frame.
Ω_{pq}^r	...Skew symmetric matrix of the rotation rate r
σ_X	...Standard deviation of X
ρ	...True range between satellite and receiver antenna (m)
Φ	...Transition matrix
Δt	...Time increment ($t_{k+1} - t_k$)
$\Delta\theta_{pq}^r$...Vector of angular increments of q frame, relative to p frame, ...expressed in r frame
Δv_f^q	...Velocity increments expressed in q frame

Abbreviations

ADR	...Accumulated Doppler Range
AMR	...Anisotropic Magnetoresistance
C/A	...Coarse/Acquisition
CDMA	...Code Division Multiplier Access
DGPS	...Differential GPS
DOP	...Dilution of Precision
ECEF	...Earth-Centred Earth-Fixed
ENU	...Easting Northing Up
EKF	...Extended Kalman Filter
FDMA	...Frequency Division Multiple Access
HSGNSS	...High Sensitivity GNSS
GNSS	...Global Navigation Satellite Systems
GPS	...Global Positioning System
IMU	...Inertial Measurement Unit
INS	...Inertial Navigation System
ITAR	...International Traffic in Arms Regulations
LLF	...Local Level Frame
LKF	...Linearized Kalman Filter
LOD	...Level of Decomposition
MEMS	...Micro-Electro-Mechanical Sensors
PPM	...Parts Per Million
PRN	...Pseudo-Random Noise
POE	...Point of Expansion
PVA	...Position Velocity Acceleration
RF	...Radio-Frequency
RFID	...Radio Frequency Identification

RMS	...Root Mean Square
RTK	...Real Time Kinematic
TEC	...Total Electron Content
UERE	...User Equivalent Ranging Error
ZUPT	...Zero Velocity Update

CHAPTER 1: INTRODUCTION

The world is changing rapidly before our very own eyes in the realm of automated navigation. Some current services inside some car manufacturers such as BMW's Parking Assistant allows for automated parallel parking, while others such as Dodge use the automated safety braking feature as an important selling point consumers are interested in. It is also common knowledge that Google has been successfully testing self-driven cars in California. In other words, automation is a selling point to consumers.

This need for automation and technology can be seen in our everyday life in the smartphone: a technological device that has become as essential as carrying a wallet. Smartphones are able to do a lot of advanced tasks because of the different sensors it has become standard for them to have. Smartphones have accelerometers, gyroscopes, magnetometers, pressure sensors, temperature sensors, and high sensitivity GNSS receivers. It is these sensors that rotate the smartphone screen when oriented one way or another, count the steps taken per day, and allow the owner to share their location with their friends in social media.

The purpose of this thesis is to put the two concepts mentioned above together. Accelerometers and gyroscopes inside smartphones are meant to be used for identifying obvious changes in orientation, but how good are they to be used for automated navigation? The sensors required in the Google self-driven car are orders of magnitude better, larger and more expensive. This thesis will explore the suitability of currently available smartphone sensors for navigation through the creation of a loosely-coupled

inertial navigation system (INS) that blends the smartphone accelerometer, gyroscope and GNSS data.

1.1 Background

The following section introduces some of the core components used in this thesis research such as smartphones, GNSS, and INS.

1.1.1 Smartphone Sensors

Smartphones are able to fit so many sensors inside a small, constrained space through the use of micro-electro-mechanical sensors (MEMS) that are quite inexpensive to make and have a small form factor. The three smartphone MEMS that were used in this thesis were accelerometers, gyroscopes, and magnetometers. Accelerometers measure the specific forces acting on the body, gyroscopes measure the rate of rotations, and magnetometers measure the magnetic field around the body. When put together, accelerometers and gyroscopes form an inertial measurement unit (IMU). The downside of being so small and cheap is that these sensors are quite noisy and have errors that are orders of magnitude bigger than those of their traditional (and considerably more accurate) counterparts. This does not mean all MEMS are obsolete as far as high-grade navigation is concerned. In fact, Honeywell has a number of MEMS IMUs that are export-controlled due to their ability at measuring high rates of rotations and accelerations that could easily be put in a guided military weapon (NovAtel, 2015e).

High-quality MEMS are used every day in the industry but they are in a whole other league compared to those used herein.

As stated earlier, smartphones also contain high sensitivity GNSS receivers. These receivers are able to compute a position in places were traditional high-accuracy receivers aren't able to. The reason for this is smartphone developers aim to provide solution availability over accuracy so that users are able to provide a position inside buildings. Having said that, the use of very low quality antennae in smartphones means the high sensitivity aspect is not exploited to its fullest.

1.1.2 INS

Using the data coming out of the smartphone accelerometers, gyroscopes, and high-sensitivity receiver, we have the building blocks for an INS. These same instruments (albeit of better quality) are integrated in the same way in the self-driven cars mentioned earlier. The difference being that the type of integration is more complex as it involves added sensors to check for proximity to other objects, among others. The algorithm part of an INS can be divided into two components: the mechanization equations, and the extended Kalman filter (EKF) (El-Sheimy, 2012).

The mechanization equations are the expressions used to integrate the raw data from the accelerometers and gyroscopes in order to get position, velocity, and attitude estimates. This is traditionally referred to as dead-reckoning and was the method used by the early navigators of the Renaissance. The navigators of the 15th and 16th centuries would sail vast oceans by using stars for positioning (could only be done at night. Sun

observations were used for latitude determination), the chip log for measuring velocity, and clocks to determine the time elapsed (Misra and Enge, 2011). These measurements weren't constantly available so they had to extrapolate the measurements in order to estimate where they were in between. There are many flaws with this and is the fact that many of them were able to discover as many lands as they did is truly astonishing. This is in essence what the mechanization equations are doing with the raw IMU data. Integrating raw accelerometer data twice gives us the change in position and integrating raw gyroscope data once gives us the change in rotation; both related to when the previous measurements were taken. This makes it possible to estimate the epoch-to-epoch change in position, velocity, and attitude (El-Sheimy, 2012). However, there is no such thing as a perfect instrument and accelerometers and gyroscopes aren't the exception; even the high-grade sensors proven in automated navigation have sources of error. This is where the EKF come in.

The EKF uses measurement and dynamic models recursively to provide an estimate that is optimal in the least-squares sense (Gao and Sideris, 2007). The dynamic model describes the system's behaviour and is kept in check via the measurement model. In terms of the INS, the dynamic model is that which describes how the position, velocity, and attitude relate to one another in the IMU whereas the measurement model is made up of the GNSS observations that are fed into the filter. The dynamic model requires estimates of the position, velocity, and attitude errors (known as the states) as well as their respective accuracies (*ibid*). Similarly, the measurement model requires GNSS observations and their respective accuracies. The EKF runs in two stages. The first stage is known as the prediction and consists in the filter using the characteristics of

the system to come up with uncertainties of the position, velocity, and attitude estimates found in the mechanization equations. The second stage is called the update and takes place when GNSS inputs are present. The observed GNSS measurements are compared to what the INS predicted at the exact same epoch. The INS then uses knowledge of both system and measurement uncertainties to blend them in the mathematically optimal sense. Incoming GNSS measurements that aren't accurate compared to what the system is estimating will be weighted accordingly and vice-versa. The important concept with the INS is that the system quality will degrade quickly in the absence of GNSS updates. The reason being it is the GNSS measurements that are keeping the INS sensor errors in check. Without these updates, the INS solution will drift at a rate that corresponds to the grade of sensors being used (Schwarz and Wei, 2000).

1.1.3 GNSS Outages in INS

INSs are not required to be used only when GNSS measurements are available. In fact, one of the purposes of an INS is to bridge areas where GNSS outages are present. This is only possible with high-grade accelerometers and gyroscopes. The idea is that high grade motion sensors have very accurate sensors whose errors are well understood, thus allowing an INS to accurately run in the prediction mode for a certain amount of time when a GNSS outage is present (El-Sheimy, 2012). However, when in the absence of GNSS input the position, velocity, and attitude will drift so it really becomes a ticking bomb as far as at what point the INS prediction can produce estimates accurate enough for the level of navigation required by the user.

1.1.4 INS Integration Methods

There are two main ways in which the INS can be designed: loosely-coupling and tightly-coupling (Petovello, 2003). Loosely-coupled integrations use position and velocity estimates as measurement updates in the filter. These could technically come from any source but GNSS is the obvious choice for providing absolute position and velocity. Tightly-coupled integrations make use of raw GNSS observations as inputs in the filter. That is, it uses GNSS pseudorange, Doppler, and carrier-phase measurements (*ibid*). In other words, loosely-coupled integrations have the INS estimating position, velocity, and attitude, whereas in tightly-coupled integrations the INS estimates the raw GNSS observations in addition to position, velocity, and attitude (El-Sheimy, 2012). The loosely-coupled integration was selected for this project due to the inability of the smartphone to provide the necessary raw measurements required in the tightly-coupled integration.

1.1.5 INS Alignment

As was stated earlier, an IMU provides relative navigation from epoch to epoch if the accelerometer and gyroscope raw data are integrated. This means a starting position, velocity, and attitude is required. This is what is referred to as the INS alignment. There are different methods for aligning depending on the application, equipment, and sensor quality (NovAtel Inc, 2015g). An initial position and velocity can be fed through GNSS, whereas pitch and roll can be estimated with enough accuracy through gravity (Schwarz and Wei, 2000); the real problem with INS alignment is with

the initial heading estimate. High-end IMUs are sensitive enough to the Earth's rotation rate and can estimate heading statically. Lower quality IMUs require vehicle movement to estimate an initial heading. The best possible way is an aided alignment which uses an initial heading estimate from an external source such as GNSS (NovAtel Inc, 2015b). This type of alignment can take place whether the vehicle is static or kinematic, and best of all is instantaneous. The INS alignment is a very important part of the INS. Wrong initial estimates will lead to solution divergence (Godha, 2006).

1.2 Previous Research and Limitations

There has been extensive research done in the field of INS that has assessed the validity entry-level inertial measurement units such as that done by Kong (2000), Gao (2007), Godha (2006), among others. However, the entry-level sensors in those publications are orders of magnitude better than the smartphone sensors used herein. That is, those sensors are entry-level as far as industry standards are concerned and cost thousands of dollars compared to the smartphone sensors that are in the single digit dollar range (Dixon, 2014). A research paper by Guo et al. (2015) performed a loosely-coupled integration using sensors inside the Apple iPhone but limited results to position and velocity which are largely dependent on the position and velocities fed into the filter from the GNSS receiver. A complete analysis of smartphone-level sensors requires an analysis of attitude and especially of heading as it is the attitude component that is hardest to estimate due to it not being directly observable in the absence of external heading-aiding sensors (Bobye, 2015).

1.3 Contributions

The main contributions of this work can be summarized as follows:

1. Development of a loosely-coupled INS filter that makes use of smartphone accelerometers, gyroscopes, and GNSS observations. The filter can potentially be provided to an entire community of developers of smartphone applications looking for highest grade of navigation possible with smartphone sensors.
2. Development of the methodology to assess GNSS velocities and height accuracies as they are not reported by the smartphone.
3. Assessment of Hall effect magnetometers inside smartphones as heading aiding sources in the INS for alignment and bias drift constraint purposes.
4. Assessment of the relative difference in performance of industry leading tightly-coupled INS filter with user-generated loosely-coupled INS filter under open-sky and urban canyon environments using a high accuracy GNSS receiver and high-end IMU.
5. Assessment of position, velocity, and attitude using smartphone sensors in an author-generated loosely-coupled INS under open-sky and urban canyon environments.

1.4 Outline

Chapter 1 presents motivation, objectives, background, and contributions of the work done in the creation of a loosely-coupled INS filter that uses MEMS accelerometers and gyroscopes from the Moto-X Android smartphone.

Chapter 2 provides an overview of both GNSS and INS technologies used in this thesis. GNSS measurements and error sources are presented in the first part. INS frames, errors, mechanization equations, classification and alignment methods are presented in the second part.

Chapter 3 is an introduction to processing methods. Specifically, the least-squares adjustment and extended Kalman filter.

Chapter 4 provides an in-depth look at the extended Kalman filter taking place behind the scenes of the INS. The integration strategies are described as well as INS-specific dynamic and measurement models, vehicle motion constraints, and sensor heading aiding.

Chapter 5 assesses the author-generated loosely-coupled INS with that of the tightly-coupled SPAN filter in open-sky and urban canyon environments. The NovAtel Flexpak6 with the IMU-LN200 was used throughout. Position, velocity, and attitude differences are shown.

Chapter 6 assesses the author-generated loosely-coupled INS using the Moto-X Android smartphone sensors in open-sky and urban canyon environments. The results

were compared to the truth trajectory from the NovAtel SPAN system (NovAtel Flexpak6 and IMU-LN200). Position, velocity, and attitude differences are shown.

Chapter 7 concludes with the results and findings obtained in this research. It provides recommendations for future work and states the limitations from the work in this thesis.

CHAPTER 2: OVERVIEW OF GNSS AND INS

Global navigation satellite systems (GNSS) allow users to determine their real time absolute position anywhere in the world through means of multilateration provided they are able to track at least four GNSS satellites. A GNSS is made up of three components or segments: space segment, control segment, user segment.

The Space segment consists of the satellites orbiting the Earth at around 20,000 km altitude (Misra and Enge, 2011). The control segment consists of data uploading stations, master control stations, and base stations which are used to monitor the health and status as well as to control the satellites. The user segment consists of GNSS receivers able to decode and use the incoming satellite signals to multilaterate a user position. The three segments are shown below in Figure 2-1.

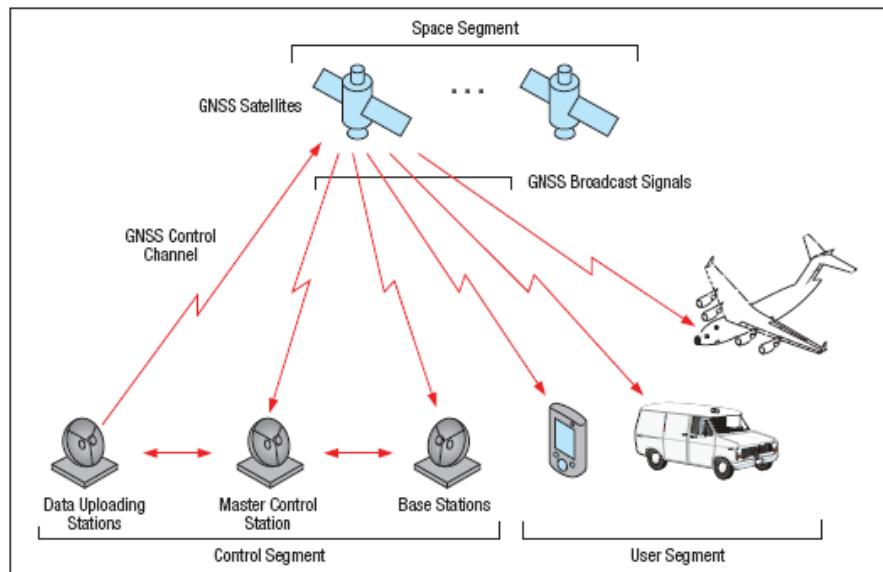


Figure 2-1: GNSS overview (NovAtel Inc, 2015d)

There are currently 4 GNSS constellations in use (*ibid*). GPS (the original GNSS) and GLONASS are maintained by the United States and Russia respectively and they provide global coverage with GPS typically having 30 operational space vehicles (SVs) and GLONASS having 24 operational SVs. It is worth mentioning both of these constellations also have in-orbit spares. The European-maintained Galileo constellation has only deployed 10 satellites that are currently still being used for development purposes. However, the plan is to have Galileo fully running with 30 SVs by 2020 (NovAtel Inc, 2015b). The fourth system in use is called BeiDou and is maintained by the People's Republic of China. BeiDou currently has 14 SVs covering Asia and the Middle East but a full, global constellation is expected to have 35 SVs by 2020.

The introduction of additional constellations to GPS is very good for users as it provides increased position availability in adverse environments such as urban canyons. The higher satellite availability also means improved satellite geometry (especially at high and low latitudes) which translates to improved position accuracy (Misra and Enge, 2011).

GPS, Galileo, and BeiDou work through code division multiple Access (CDMA) in which all satellites in a given constellation operate at the same reserved frequency. Each satellite's raw signal (sent on two or more frequencies) is encoded with a navigation message (e.g. 50 bps for GPS) and by either an open or encrypted code that when known on the receiver end allows differentiating each satellite in the constellation (*ibid*). GLONASS on the other hand, operates through frequency division multiplier access (FDMA) in which satellites are distinguished from one another by operating at different frequencies. The encrypted frequencies (such as the legacy GPS L2P) are exploited in

civilian applications through the use of semi-codeless tracking inside the receiver. All constellations have plans of providing a secondary open code which will negate the need for semi-codeless technology in receivers and thus increase accuracy for entry level GNSS receivers.

2.1 GNSS Measurements

GNSS provide users with three types of measurements which serve as updates in an INS filter: pseudorange measurements, carrier-phase measurements, and Doppler measurements.

2.1.1 Pseudorange Measurements

Pseudorange measurements consist of the time it takes for a signal to travel from the satellite to the receiver multiplied by the speed of light. As the name implies, these ranges contain errors which have to be taken into account in the positioning algorithm (*ibid*). The ranging errors are caused by a number of factors such as the satellite and receiver clocks not being completely in sync with the constellation time base. Let's focus on GPS for the next example.

Recall that the raw satellite signal is sent over two or more predefined frequencies. These signals are modulated by the navigation message (50 bps) and both the C/A (transmitted at 1.023 MHz on L1) and P(Y) codes (transmitted at 10.23 MHz on both L1 and L2). The C/A codes are modulated on the quadrature part of the signal and the P(Y)

codes are modulated on the in-phase part of the signal. The receiver knows what the unique code (known as pseudo-random noise, PRN code) for each satellite is. When the raw signal reaches the processing part of the receiver, the received PRN code is shifted until it lines up with the internal PRN code (Langley, 2016). The time shift required to line up both PRN codes is the time of travel of the signal. The pseudorange equation is as follows:

$$P_{(m)} = \rho + c(\delta T - \delta t) + \delta_{ion} + \delta_{trop} + \delta_{orb} + \varepsilon_{code} \quad (2.1)$$

where:

$P_{(m)}$ is the measured pseudorange (metres)

ρ is the true range between the satellite and receiver antennas (metres)

c is the speed of light in vacuum (m/s)

δT is the receiver clock error (seconds)

δt is the satellite clock error (seconds)

δ_{ion} is the ionosphere induced error (metres)

δ_{trop} is the troposphere induced error (metres)

δ_{orb} is the orbital error (metres)

ε_{code} is the code level multipath plus noise (metres)

An important point worth mentioning is that the chips that make up the PRN code have a period of 1 μ s meaning each chip is 293 metres in length (ibid). This low resolution from which C/A based pseudorange positioning is computed from translates to a positioning accuracy of a few metres depending on the type of receiver correlators. The pseudorange measurement itself is accurate to tens of cm in high-accuracy receivers (Langley, 2016).

2.1.2 Carrier-Phase Measurements

Also referred to as accumulated-Doppler-range (ADR), carrier-phase measurements have a much higher resolution than pseudorange measurements and as such when used in a base-rover setup can provide positioning accuracies of 1-2 cm, sometimes even better (Langley, 2016). That is because these types of observations make use of the fact that at the specific GNSS frequency, the carrier wave has a small wavelength (e.g. 19 cm for GPS L1, 24 cm at GPS L2). Thus, the distance from the antenna to the satellite can be thought of as a sum of wavelengths (cycles) through the propagation space. The number usually has an integer and fractional component so an analogy of a pseudorange measurement (in cycles) plus an initial integer offset can be made. The fractional component and the change in integer offsets can be measured easily by the tracking loops but the same cannot be said about the initial integer offset (NovAtel Inc, 2015d). This value is referred to as the integer ambiguity. Thus, the total carrier-phase (Φ_{TOTAL}) is made up of the carrier phase measurement ($\Phi_{OBSERVED}$) (which includes integer and fractional changes) plus the integer ambiguity term (N).

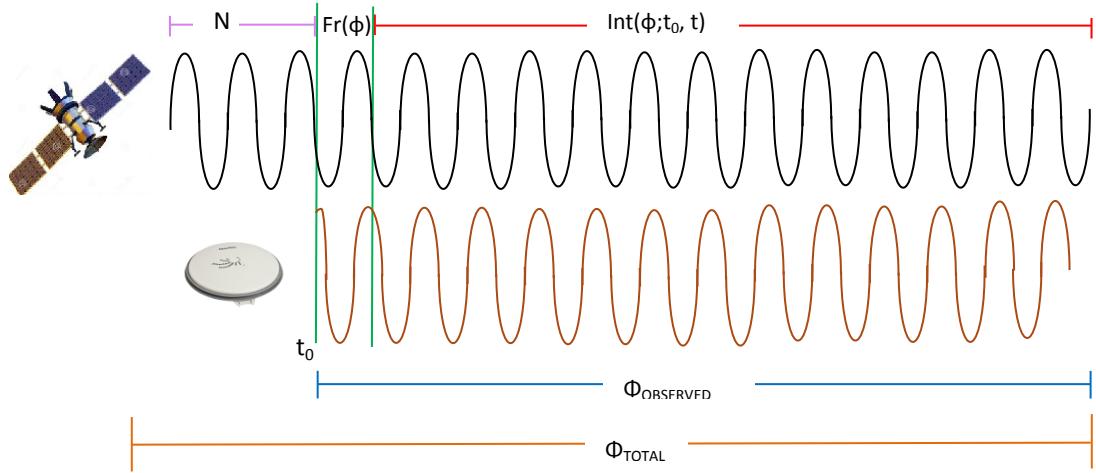


Figure 2-2: Carrier phase observation

$$\Phi_{TOTAL} = \Phi_{OBSERVED} + N \quad (2.2)$$

$$\Phi_{OBSERVED} = \Phi_{TOTAL} - N \quad (2.3)$$

which after following the derivation in Wells et al., (1999), gives us:

$$\Phi_{(m)} = \rho + c(\delta T - \delta t) - \delta_{ion} + \delta_{trop} + \delta_{orb} + \varepsilon_{carrier} + \lambda N \quad (2.4)$$

where the differences with respect to Equation 2.1 are:

$\varepsilon_{carrier}$ is the cm level multipath plus noise from the carrier (metres) (Langley, 2016)

λN is the integer ambiguity multiplied by the carrier wavelength (metres)

As can be seen in Equation 2.4, the carrier-phase observable contains the same types of errors as the pseudorange measurement plus the integer ambiguity component. There are two more differences that stand out. The first one is that the ionospheric effect is

subtracted rather than added because of the fact the ionosphere causes an advance in carrier-phase observations whereas it causes a delay in the pseudorange measurements. The second difference is that the multipath and noise in the carrier phase measurements are much smaller (cm-level) instead of the metre-level that would be caused from the code measurements.

2.1.3. Doppler Measurement

The Doppler frequency is the instantaneous rate of change of the carrier-phase measurement and is useful for providing velocity. This observation is not impacted by the integer ambiguities (derivative of constant is zero) meaning the derivative of Equation 2.4 becomes:

$$\dot{P}_{(m/s)} = \dot{\rho} + c(\dot{\delta T} - \dot{\delta t}) - \dot{\delta_{ion}} + \dot{\delta_{trop}} + \dot{\delta_{orb}} + \dot{\varepsilon} \quad (2.5)$$

where,

$\dot{P}_{(m/s)}$ is the observed rate of range (m/s)

$\dot{\rho}$ is the true range rate between the satellite and the receiver (m/s)

$\dot{\delta T}$ is the receiver clock error drift (m/s)

$\dot{\delta t}$ is the satellite clock error drift (m/s)

$\dot{\delta_{orb}}$ is the orbital error drift (m/s)

$\dot{\varepsilon}$ is the drift due to multipath and noise (m/s)

2.2 Error Sources in GNSS and How to Mitigate Them

It is now time to take a look at the individual GNSS error sources mentioned in the previous section as well as how to mitigate them. Knowing what causes the errors is important in our understanding as to how to minimize or cancel out their effect. First, it is important to mention that the positional accuracy is a function of the sum of error sources as well as satellite geometry (Misra and Enge, 2011). The distribution of satellites in the sky is quantified through the unitless dilution of precision (DOP). The sum of errors in the line of sight from the antenna to the receiver is referred to as the user equivalent ranging error (UERE):

$$\text{Position accuracy} = \text{UERE} * \text{DOP} \quad (2.6)$$

Hence, the more satellites in view, the better the geometry, and the better the position accuracy will be.

The errors making up the UERE can be dealt with by either modelling or cancelling them out as will be further explained below. The first method called differential GNSS consists of having a static GNSS receiver (called the base) transmit corrections or raw observations to a secondary stationary (or kinematic) GNSS receiver, referred to as the rover (Novatel Inc, 2015d). There are two types of differential GNSS methods used: code-based and carrier-phase-based (referred to as RTK). RTK stands out for real time kinematic but doesn't necessarily have to be performed in real time (Langley, 2016). Both methods require the base to be static and placed over a previously surveyed point.

In code-based differential GNSS, the known position of the base is compared with the computed position and the difference in known vs. computed range for each satellite is sent as corrections to the rover. Code-base differential GNSS will reduce or eliminate orbital, atmospheric, and satellite errors (O’Keefe, 2007). These corrections On the other hand, RTK sends the raw observations to the rover which then computes a double differencing of observations in order to reduce or cancel out common errors experienced by both receivers. Readers are invited to read Petovello (2003), Godha (2006) and Wells et al. (1999) among others for further details on double differencing. RTK requires the use of carrier-phase measurements which complicates matters because the integer ambiguity needs to be known. However, it is these carrier-phase measurements that allow 1 cm + 1 PPM accuracy in RTK compared to half-metre accuracy in code based differential GNSS (NovAtel Inc, 2015b).

The second method for correcting a receiver’s position is through modelling of errors. This can be done either in post-processing through software, or in real-time through satellite delivered corrections. The latter requires the antenna to be able to track a separate set of geostationary satellites that broadcast corrections as well as a subscription to the service. The most common satellite correction systems are OmniSTAR, TerraStar and StarFire. These satellite-based correction services are able to offer decimetre positioning accuracy throughout the world (NovAtel Inc, 2015b) but do require about 20 minutes of convergence. This long start-up time is due to noise in ionosphere-free measurement combination, tropospheric wet delay estimation and carrier-phase ambiguity resolution (Maron, 2014). It is worth mentioning this type of satellite derived corrections (also known as precise point positioning, PPP) use wide-

lane ambiguity fixing which means that it converges to a stable but not unique solution, unlike RTK (*ibid*). Wide-lane and narrow-lane combinations are used for integer ambiguity determination. Readers are invited to read Misra and Enge (20011) for further information on these topics.

2.2.1 Ionospheric Errors

The ionosphere extends from about 50 to 1000 km, or higher, and consists of ionized air caused by the Sun's radiation. This means the ionosphere behaviour is dependent on the time of day (Misra and Enge, 2011). The Sun also has an 11 year cycle during which there are variations in the amount of sunspots and solar flares. The peak of the solar cycle is when there is a maximum in solar flares which generate a very active ionosphere and subsequently large ionospheric errors in GNSS (O'Keefe, 2007).

The total electron content (TEC) in the ionospheric line of sight changes the refractive index and affects RF signals at GNSS frequencies by advancing the carrier phase measurements and delaying the code measurements (Misra and Enge, 2011). The ionosphere is also dispersive in nature, which means it affects different GNSS frequencies differently. This means that in the absence of differential methods, the ionospheric errors can be cancelled through the use of a dual frequency receiver (*ibid*). Single frequency receivers not receiving differential corrections model the ionosphere through the Klobuchar model that is broadcast by the satellites (NovAtel Inc, 2015b). On the other hand, receivers using differential positioning methods effectively cancel

out the ionospheric effect provided the atmospheric behaviour is the same at both ends of the baseline.

2.2.2 Troposphere Errors

The troposphere extends from sea level to the stratosphere (~9 km – 17 km) and it contains a dry part and a wet part (water vapour) (O’Keefe, 2007). Unlike the ionosphere, it is not dispersive in nature meaning its effect is the same over the multiple frequencies broadcast by a specific GNSS satellite. An interesting fact about the troposphere error is that 90% of it comes from the dry component which can be modelled with an accuracy of 1% whereas the remaining 10% comes from the wet component. This wet component is difficult to predict due to variation in water vapour density meaning the models are only 20% accurate (Misra and Enge, 2011). This type of error is reduced through modelling and differential techniques leaving a 10-15 cm error residual for a 100 km baseline (*ibid*). This error can actually be smaller if residual tropospheric delay is estimated from the data itself (Langley, 2016).

2.2.3 Orbital Errors

These types of errors arise from imperfections in the satellite orbit with respect to the planned trajectory as defined by the ephemerides. The satellite health and status are monitored continuously through the ground control stations and the orbits are typically corrected once a day (Misra and Enge 2011). The uploaded corrected orbital parameters are then transmitted to receivers via the broadcast ephemerides. Differential GNSS

techniques minimize the effect of the orbital errors. Usually, a 100 km baseline will contain a remaining orbital error baseline of around 5 cm (*ibid*).

2.2.4 Satellite and Receiver Clock Errors

Satellite and receiver clock errors are biases due to the fact that receivers and satellites are not completely in sync with the GNSS set time base (e.g. GPS Time for GPS). Both of these clock biases cancel out during RTK due to the nature of double differencing. The receiver clock error can be expected to be from a few metres to a few thousand kilometers (Misra and Enge, 2011). If running in single point mode (i.e.: without double differencing), the receiver clock error is computed in the filter in addition to the three-dimensional positional coordinates whereas the estimated error of the satellite clock is computed from satellite clock coefficients broadcast in the navigation message.

2.2.5 Multipath

Multipath is originated by reflections in the nearby environment that cause the GNSS signal delays to be longer than they should be (O'Keefe, 2007). Since the receiver expects the signals to be a direct line-of-sight, a bias is introduced to the ranging measurements. Multipath is hard to model due to the fact that it is dependent on the environment but there are methods to mitigate it (Park et al., 2004). The most obvious one is not to set up the antenna in a challenging environment. Another way is to choose

a high quality antenna that rejects signals coming from below the horizon as well as those that are left handed polarized (Weill, 1997). The reason for this is that GNSS signals are right handed polarized and become primarily left-handed if reflected. Only nearby objects will induce troublesome multipath because the receiver tracking loops are designed to reject indirect ranges longer than 1.5 chips (~ 440 m). The type of correlator in the receiver also plays a very important role in multipath determination. Narrow correlators are better at computing the PRN autocorrelation peak meaning they are also better at identifying and rejecting multipathed signals (O'Keefe, 2007). Figure 2-3, shown below illustrates the effect of multipath on the ranged signals.

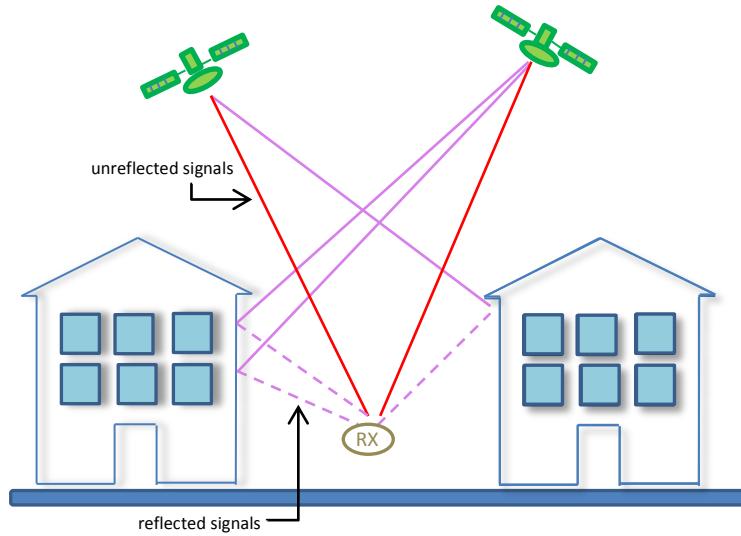


Figure 2-3: Antennas in multipath environment (NovAtel Inc, 2015b)

2.2.6 Measurement Noise

The amount of noise in the receiver tracking loops plays a very important role in how accurately a receiver can measure pseudoranges and carrier phase (Langley, 1997). The

most basic kind of electrical noise is produced by random movement of electrons in a conductor (such as those in a GNSS receiver) and is referred to as thermal noise (ibid). This random movement of electrons in turns creates an electromagnetic radiation. The antenna ends up picking up electromagnetic radiation from the receiver, the sky, the ground, and objects in the vicinity of the receiver antenna (ibid).

Noise experienced by the receiver in the form of cable loss and receiver temperature also has to be taken into account (ibid). Putting all of these noise sources together, C/A code measurements can be observed with a precision of 4 cm whereas carrier phase measurements can be observed with a precision of around 0.5 mm (NovAtel Inc, 2015h). Table 2-1 below summarizes the characteristics and approximate magnitudes of GNSS errors in single point positioning.

Table 2-1: Characteristics and magnitudes of GNSS errors (adapted from Godha (2006), Gao (2007), O'Keefe (2005), and Langley (2016))

GNSS Errors	Characteristics	Magnitude in Single Point
Ionosphere	Spatially correlated Frequency dependent Varies with location and solar activity	2 – 50 m
Troposphere	Spatially correlated	2 – 30 m

	Frequency independent	
Orbital	Spatially correlated	< 1 m
Satellite clock	Estimated through coefficients broadcast in ephemerides	<1 m
Receiver clock	Estimated in filter along with position	200 ns – several ms
Code Multipath	Depends on environment, antennas and elevation angle	0.2 –500 m
Code Measurement Noise	Depends on noise experienced by antenna and receiver	~ 0.4 m

2.3 High Sensitivity Assisted GNSS

The differential correction methods mentioned in the previous section are not commonly used with low-end receivers such as those found in the Moto-X Android smartphone, used in this thesis. However, these low-end receivers do have a few differences in design that give them an advantage over high accuracy receivers. GNSS receivers such as those found in smartphones have a higher sensitivity compared to their high accuracy counterparts due to the fact the incoming signals are integrated for a longer amount of time (Driscoll et al., 2011). The code-correlation computation in

GNSS receivers is what is able to raise the buried signal from the strong ambient noise (Langley, 1997). High accuracy GPS receivers integrate the received signal for 1 ms, which is the duration of one C/A cycle and results in the ability to acquire and track signals at around -160 dBW. Every 20 ms there is a new bit of navigation data transmitted (recall the navigation message is sent at 50 bps) and this bit change limits the coherent integration in the receiver unless the navigation bits are known a priori (MacGougan, 2003). High sensitivity receivers make use of assisted GNSS to remove the navigation message via the cellular provider. This, along with a large increase in number of correlators compared to those in high accuracy receivers, and limitation of residual frequency errors, enable high sensitivity receivers to track much weaker GNSS signals. By removing the navigation message, the 1 ms integration can now be lengthened about 1000 times which in turn is able to raise weaker signals out of ambient noise (Zhang et al., 2011). This means high sensitivity GNSS receivers are able to acquire and track signals down to the -190 dBW level, which is a 30 dB increase with respect to high accuracy receivers. Thus the advantage of high sensitivity GNSS receivers is they can track GNSS signals indoors, albeit at a very low accuracy. They were not meant for providing survey grade accuracies. The high sensitivity aspect of the GNSS receiver inside the smartphone did not provide any advantages due to the nature of this project being related to getting the highest accuracy possible rather than the greatest availability.

2.4 Coordinate Frames Used in Inertial Navigation

There are different frames that have to be considered when dealing with inertial navigation systems. The mechanization equations are computed in either the Earth-Centred-Earth-Fixed frame (e-frame) or the Local-Level frame (l-frame) although the position and velocity are usually output in the latter due to it being much easier to physically relate to.

Inertial Frame (i-frame)

Origin = Earth's centre of mass

Z^i -axis = Parallel to Earth's spin axis

X^i -axis = Towards mean vernal equinox

Y^i -axis = Orthogonal to X and Z axes forming a right-handed system

The inertial frame is a fixed non-rotating frame that is used in the mechanization equations to distinguish observables induced by the Earth's rotation (Coriolis effect, centrifugal acceleration) from those that don't (raw gyroscope measurements).

Earth Centred Earth Fixed (ECEF or e-frame)

Origin = Earth's centre of mass

Z^e -axis = Parallel to Earth's spin axis

X^e -axis = Towards mean Greenwich meridian

Y^e -axis = Orthogonal to X and Z axes forming a right-handed coordinate frame

Local-Level (LLF or ENU or l-frame)

Origin = IMU centre of navigation

Z^l -axis = Upwards (parallel to normal gravity)

X^l -axis = Orthogonal to normal gravity, pointing towards East

Y^l -axis = Orthogonal to normal gravity, pointing towards North

As per Schwarz and Wei (2000), mechanizing in the e-frame is mathematically simpler than the l-frame because the normal radius of curvature of the ellipsoid does not have to be taken into account which greatly simplifies most computations. However, the normal gravity computation in this frame is more complicated than in the l-frame. Note that the Local-level frame is defined here as being right-handed. However some applications such as hydrography define the l-frame as a left-handed. In such cases the Z axis is positive downwards, in line with normal gravity. Both e-frame and l-frame are shown below in Figure 2-4.

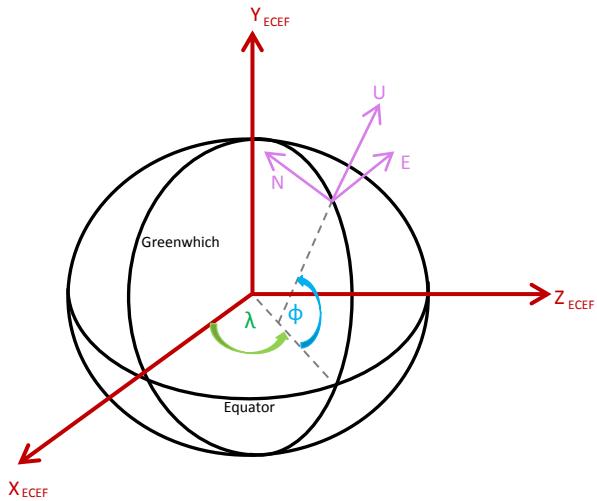


Figure 2-4: ECEF frame (e-frame) and local level frame (l-frame)

Vehicle Frame (v-frame)

Origin = IMU centre of navigation

Z^v = Upwards

X^v = Towards right side of vehicle

Y^v = Towards front of vehicle

The vehicle frame is used for attitude purposes. During the use of the mechanization equations, the attitude estimate in either ECEF or LLF will be rotated to the computation frame and then to this frame. The reason for this is that most applications require the attitude of a moving vehicle. To be able to output in this frame it is necessary to take into account the difference between the IMU enclosure frame, computation frame (see below), and the vehicle frame.

In all of the tests carried out in for the research reported, the smartphone was mounted in such way that the IMU enclosure frame (see below) matched the vehicle frame. This was done for ease of setup although there are no mathematical constraints to how the IMU is mounted provided the appropriate rotations are taken into account. The v-frame is shown below in Figure 2-5.

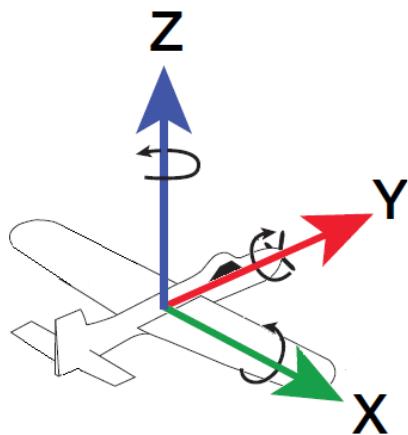


Figure 2-5: Vehicle frame (NovAtel Inc, 2015g)

IMU Enclosure Frame

This is the frame of the physical sensors in the IMU. It is suggested to mount the IMU in the default way where the enclosure's axes match that of the vehicle frame. Otherwise, further rotations would be required to get the solution output in the vehicle frame. This frame is shown below in Figure 2-6 for the sensors used herein.

Origin = IMU centre of navigation

Z^b = Upwards, away from the smartphone

X^b = Towards right side of smartphone

Y^b = Towards top of smartphone

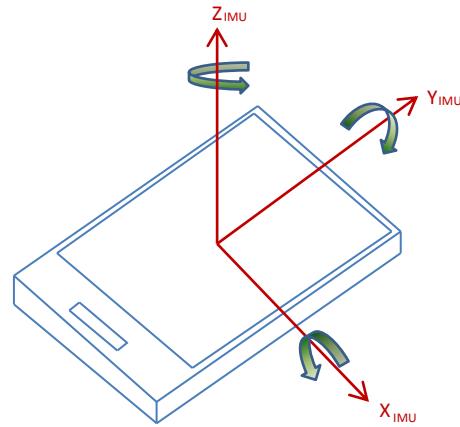


Figure 2-6: IMU enclosure frame

Computation Frame (b-frame)

Origin = IMU centre of navigation

Z^c -axis = Upwards in line with gravity

Y^c -axis = Dependent on mapping

X^c -axis = Dependent on mapping

Depending on the vehicle size constraints, the IMU might not be mounted in such a way that the enclosure frame matches the vehicle frame. For example, it might be mounted with the enclosure frame's X-axis in line with gravity instead of the Z-axis. This frame is used to map the axes of the enclosure frame to a computation frame where

the Z-axis is always in line with normal gravity. The computation frame will match the enclosure frame if the enclosure Z-axis points upwards.

2.5 Coordinate Transformation

Throughout this thesis it will be necessary to transform vectors back and forth from the e-frame to the l-frame as well as the b-frame. Rotating from one frame to another can be done by sequentially rotating each axis in the “from” frame until the “to” frame is matched. The elementary rotations around the three axes are defined as follows:

$$\mathbf{R}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\xi) & \sin(\xi) \\ 0 & -\sin(\xi) & \cos(\xi) \end{pmatrix} \quad (2.7)$$

$$\mathbf{R}_2 = \begin{pmatrix} \cos(\eta) & 0 & -\sin(\eta) \\ 0 & 1 & 0 \\ \sin(\eta) & 0 & \cos(\eta) \end{pmatrix} \quad (2.8)$$

$$\mathbf{R}_3 = \begin{pmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (2.9)$$

The product of elementary rotations forms a rotation matrix \mathbf{R}_j^p where the subscript indicates the “from” frame and the overscript indicates the “to” frame. The basis vectors that make up a rotation matrix are of unit length and their dot product defines the cosine of the angles between the vector pairs. This is why the rotation matrices are sometimes also referred to as direction cosine matrices (Mohamed and Mamatas, 2012). This

allows us to convert vectors from one frame to another as in the example below where the arbitrary vector \mathbf{B} is to be converted from the e-frame to the l-frame:

$$\mathbf{B}^l = \mathbf{R}_e^l \mathbf{B}^e \quad (2.10)$$

A very important property that facilitates transforming vectors from one frame to another is:

$$(\mathbf{R}_j^p)^{-1} = (\mathbf{R}_j^p)^T = \mathbf{R}_p^j \quad (2.11)$$

The rotation between the e-frame and l-frame consists of two consecutive rotations around the X^e and Z^e axis. The conversion between e-frame and l-frame is shown below in Figure 2-7.

$$\mathbf{R}_e^l = (\mathbf{R}_e^l)^T = \mathbf{R}_1(90^\circ - \varphi) \mathbf{R}_3(\lambda + 90^\circ) \quad (2.12)$$

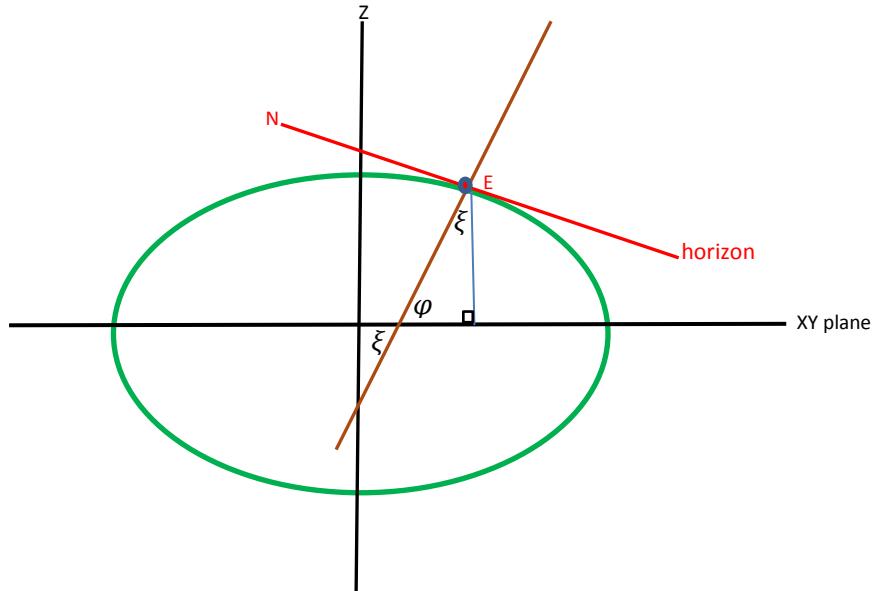


Figure 2-7: Visual of e-frame to l-frame conversion

The other type of rotation that is done often is the conversion from either the l-frame or the e-frame to the b-frame. Note that unlike with the previous transformation, the order of rotations does matter when working with the b-frame because of the ENU convention behind it (Mohamed and Mamatas, 2012). When defining a rotation from a mechanization frame (either e-frame or l-frame) to the body frame a sequence of y, x, z must be employed. Thus, a rotation from the b-frame to the mechanization frame would consist of the opposite order of z, x, y as follows:

$$\mathbf{R}_b^n = (\mathbf{R}_n^b)^T = \mathbf{R}_z(-\psi)\mathbf{R}_x(-\xi)\mathbf{R}_y(-\eta) \quad (2.13)$$

where the n overscript stands for either the e-frame or l-frame. Note that if $n = l$ the rotations around the x, y, z axes are called pitch, roll and yaw respectively. Since navigation applications deal with heading instead of yaw, Equations 2.13 would have to be modified accordingly.

$$\mathbf{R}_b^l = (\mathbf{R}_l^b)^T = \mathbf{R}_z(\psi)\mathbf{R}_x(-\xi)\mathbf{R}_y(-\eta) \quad (2.14)$$

Recall that throughout this thesis the b-frame matches the v-frame and therefore no further rotations would be required to output the vehicle pitch, roll and, heading.

2.6 Mechanization Equations

As was mentioned at the start of the chapter, INS is essentially a dead reckoning system providing position, velocity, and attitude from the measurements inside an IMU (El-Sheimy, 2012). The first step in the INS solution is to integrate the raw

measurements from the accelerometers and gyroscopes inside the IMU. Because the phone was mounted in such way that the enclosure frame already matches the vehicle frame, the raw measurements are already in the b-frame. Therefore, the mechanization equations will rotate the raw observations from the b-frame to the e-frame (or l-frame). Note that just running the mechanization equation won't provide an accurate INS as the solution will drift due to sensor biases explained in the INS Error section of this thesis. That is, in order to provide a proper INS solution, the sensor errors have to be accurately modelled and removed. The first order differential equations representing vehicle motion can be mathematically modelled as follows (El-Sheimy, 2012).

$$\begin{pmatrix} \dot{\mathbf{r}}^e \\ \dot{\mathbf{v}}^e \\ \dot{\mathbf{R}}_b^e \end{pmatrix} = \begin{pmatrix} \mathbf{v}^e \\ \mathbf{R}_b^e \mathbf{f}^b - 2\boldsymbol{\Omega}_{ie}^e \mathbf{v}^e + \boldsymbol{\gamma}^e \\ \mathbf{R}_b^e (\boldsymbol{\Omega}_{ei}^b + \boldsymbol{\Omega}_{ib}^b) \end{pmatrix} \quad (2.15)$$

Where the dots on top of the variable represents the time derivative, e, b represent the e-frame and b-frame respectively, and the bold values represent the input raw data from the IMU. The remaining symbols are:

\mathbf{r} is the position vector (x^e, y^e, z^e)

\mathbf{v} is the velocity vector (v_x^e, v_y^e, v_z^e)

$\boldsymbol{\gamma}$ is the normal gravity vector

\mathbf{R}_b^e is the rotation matrix from the b-frame to the e-frame

$\boldsymbol{\Omega}_{ie}^e$ is the skew-symmetric matrix of the rotation rate, ω_{ie}^e

$\boldsymbol{\Omega}_{ei}^b$ is the skew-symmetric matrix representing the Earth's rotation rate

\mathbf{f} is the specific force vector (raw accelerometers observations)

Ω_{ib}^b is the skew-symmetric matrix of the rotation rate, ω_{ib}^b , representing the raw gyroscope observations.

The mechanization equations are the solution to the first order differential equation shown above. The following is the solution derivation of the mechanization equations in the e-frame as shown in Schwarz and Wei (2000).

1. Remove bias from raw accelerometer and gyroscope measurements:

$$\theta_{ib}^b = (\tilde{\theta}_{ib}^b - \mathbf{b}_g \Delta t) / (1 + S_g) \quad (2.16)$$

$$v_f^b = (\Delta \tilde{v}_f^b - \mathbf{b}_a \Delta t) / (1 + S_a) \quad (2.17)$$

where $\tilde{\theta}_{ib}^b$ corresponds to the gyroscope measured angular increments. The subscript indicates the rotation is from the i-frame to the b-frame and the overscript indicates the rotation is measured in the b-frame. $\Delta \tilde{v}_f^b$ corresponds to the accelerometer measured velocity changes. The subscript indicates the measurement is done in the body frame. b_g and b_a correspond to the gyroscope and accelerometer biases respectively. S_g and S_a correspond to the gyroscope and accelerometer scale factors respectively.

Note the biases are multiplied by the IMU operating time increment (inverse IMU operating frequency). The reason for this is that IMUs usually output the raw observations as velocity increments (for accelerometers) and angular increments (for

gyroscopes). Thus, the bias estimates have to be transformed to increments prior to being removed from the raw measurements. Something worth stating is the Moto-X Android phone used in this thesis outputs raw measurements as acceleration and angular velocity. Thus, raw measurements were also multiplied by the IMU operating time increment in order to match the equations above.

2. Remove Earth's rotation from raw gyroscope measurements.

$$\theta_{ie}^b = (R_e^b \omega_{ie}^e) \Delta t \quad (2.18)$$

$$\theta_{eb}^b = \theta_{ib}^b - \theta_{ie}^b \quad (2.19)$$

Equation 2.18 transforms the Earth's mean rotation rate from the e-frame to the b-frame. It then multiplies it by the IMU time increment in order for the units to match that of 2.16. Equation 2.19 represents the raw gyroscope measurements after being corrected for the Earth's rotation rate.

Note that only IMUs with a gyroscope bias of less than 15 deg/hour are capable of sensing the Earth's rotation (NovAtel Inc, 2015e). This is an important concept that will be brought up again in the INS alignment section. Very low-end gyroscopes such as the ones inside the Moto-X Android phone are unable to sense the Earth's rotation and therefore removing the Earth's rotation is not necessary. The procedure nevertheless was added here for the sake of completeness.

3. Quaternion update

$$\boldsymbol{\theta}_{eb}^b = \begin{pmatrix} \theta_x^b \\ \theta_y^b \\ \theta_z^b \end{pmatrix} \quad (2.20)$$

$$\theta = \sqrt{\theta_x^{b^2} + \theta_y^{b^2} + \theta_z^{b^2}} \quad (2.21)$$

$$s = \frac{2}{\theta} \sin\left(\frac{\theta}{2}\right) \quad (2.22)$$

$$c = 2 \left(\cos \frac{\theta}{2} - 1 \right) \quad (2.23)$$

$$\begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix}_{k+1} = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix}_k + 0.5 \begin{pmatrix} c & s\theta_z^b & -s\theta_y^b & s\theta_x^b \\ -s\theta_z^b & c & s\theta_x^b & s\theta_y^b \\ s\theta_y^b & -s\theta_x^b & c & s\theta_z^b \\ -s\theta_x^b & -s\theta_y^b & -s\theta_z^b & c \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix}_k \quad (2.24)$$

Equation 2.24 uses a quaternion approach for updating the attitude. A Quaternion is a four dimensional complex matrix that provides a convenient mathematical representation of orientations and rotations of objects in three dimensions (Mohamed and Mamatas, 2012). The quaternion contains four differential equations whereas a rotation matrix requires six differential equations. The quaternion also avoids the singularities that might be encountered with rotation matrices when dealing with pitch and roll values that are close to 90 degrees (El-Sheimy, 2012).

Please note that an initial rotation matrix with respect to the b-frame, expressed in the e-frame is required during the INS alignment procedure which will be explained in a

later chapter. This rotation matrix is then converted into the quaternion $(q_1, q_2, q_3, q_4)^T_k$ used in Equation 2.24.

4. Convert quaternion to rotation matrix in e-frame

$$\mathbf{R}_b^e = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} = \begin{pmatrix} (q_1^2 - q_2^2 - q_3^2 + q_4^2) & 2(q_1q_2 - q_3q_4) & 2(q_1q_3 + q_2q_4) \\ 2(q_1q_2 + q_3q_4) & (q_2^2 - q_1^2 - q_3^2 + q_4^2) & 2(q_2q_3 - q_1q_4) \\ 2(q_1q_3 - q_2q_4) & 2(q_2q_3 + q_1q_4) & (q_3^2 - q_1^2 - q_2^2 + q_4^2) \end{pmatrix} \quad (2.24)$$

5. Transform specific force from b-frame to e-frame

$$\mathbf{S}^b = \begin{pmatrix} 0 & -\theta_z^b & \theta_y^b \\ \theta_z^b & 0 & -\theta_x^b \\ -\theta_y^b & \theta_x^b & 0 \end{pmatrix} \quad (2.25)$$

$$\Delta v_f^e = \mathbf{R}_{b_k}^e \left(I + \frac{1}{2} \mathbf{S}^b \right) \Delta v_f^b \quad (2.26)$$

Equation 2.26 uses the rotation matrix from the previous epoch, $\mathbf{R}_{b_k}^e$, from Equation 2.24 to convert the measured velocity changes to the e-frame. It also takes into account the corrected angular increments from the current epoch.

6. Coriolis effect correction

$$\mathbf{a}^e = 2\boldsymbol{\Omega}_{ie}^e \mathbf{v}_e = \begin{pmatrix} 0 & -2\omega_e & 0 \\ 2\omega_e & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} = \begin{pmatrix} -2\omega_e v_y \\ 2\omega_e v_x \\ 0 \end{pmatrix} \quad (2.27)$$

Because the IMU is located on the surface of the Earth, it experiences so called pseudo-forces from the Earth's rotation which need to be taken into account. Equation 2.27 shows the computation for the Coriolis Effect.

7. Gravity correction

$$\boldsymbol{\gamma}^e = \frac{a_1}{r} \begin{pmatrix} (c_1 + c_2 t^2 + c_3 t^4 + c_4 t^6) x^e \\ (c_1 + c_2 t^2 + c_3 t^4 + c_4 t^6) y^e \\ (d_1 + d_2 t^2 + d_3 t^4 + d_4 t^6) z^e \end{pmatrix} + \begin{pmatrix} \omega_e^2 x_e \\ \omega_e^2 y_e \\ 0 \end{pmatrix} \quad (2.28)$$

The left-hand side of Equation 2.28 corresponds to the normal gravity correction where r is the radius to the centre of mass of the Earth (origin of e-frame) using e-frame coordinates $(x^e, y^e, z^e)^T$.

The right-side of Equation 2.28 corresponds to the Earth's centrifugal force. The centrifugal force is a function of the Earth's spin rate and the separation from the spin axis. That is, equatorial latitudes are under a higher centrifugal acceleration than higher/lower latitudes (Braun, 2006).

8. Apply Coriolis and gravity corrections to measured specific forces

$$\Delta \mathbf{v} = \Delta \mathbf{v}_f^e - \mathbf{a}^e \Delta t + \boldsymbol{\gamma}^e \Delta t \quad (2.29)$$

Note that the Coriolis (\boldsymbol{a}^e) and gravity ($\boldsymbol{\gamma}^e$) corrections are multiplied by the IMU time increment due to the fact specific forces are measured in velocity increments. The output of Equation 2.29 is the corrected velocity increments in the e-frame.

9. Compute velocity at current epoch

$$\boldsymbol{v}^e_{k+1} = \boldsymbol{v}^e_k + 0.5(\Delta\boldsymbol{v}_k + \Delta\boldsymbol{v}_{k+1}) \quad (2.30)$$

Velocity at the current epoch is computed by adding the average velocity estimates at current and previous epoch with the previous computed velocity.

10. Position computation at current epoch

$$\boldsymbol{r}^e_{k+1} = \boldsymbol{r}^e_k + 0.5(\boldsymbol{v}^e_{k+1} + \boldsymbol{v}^e_{k+1})\Delta t \quad (2.31)$$

The position is computed in a similar manner to velocity in Equation 2.30. Note the change in time which is integrating velocity to position.

11. Convert coordinates from ECEF to geodetic

As was already mentioned earlier, it is more intuitive to deal with geodetic coordinates than with the e-frame coordinates. The geodetic coordinates also need to be used in the attitude computation.

12. Compute attitude matrix in l-frame, \boldsymbol{R}_b^l

$$\boldsymbol{R}_b^l = \boldsymbol{R}_e^l \boldsymbol{R}_b^e \quad (2.32)$$

The current attitude estimate from the mechanization equations, \mathbf{R}_b^e is rotated into the local level frame for attitude computation purposes. The rotation matrix, \mathbf{R}_e^l , computed from geographic coordinates is used for the transformation.

13. Compute attitude

$$\xi = \tan^{-1}[-\mathbf{R}_b^l(3,1)/\mathbf{R}_b^l(3,3)] \quad (2.33)$$

$$\eta = \sin^{-1}[\mathbf{R}_b^l(3,2)] \quad (2.34)$$

$$\psi = \tan^{-1}[-\mathbf{R}_b^l(1,2)/\mathbf{R}_b^l(2,2)] \quad (2.35)$$

The complete mechanization equation procedure is shown below in Figure 2-8.

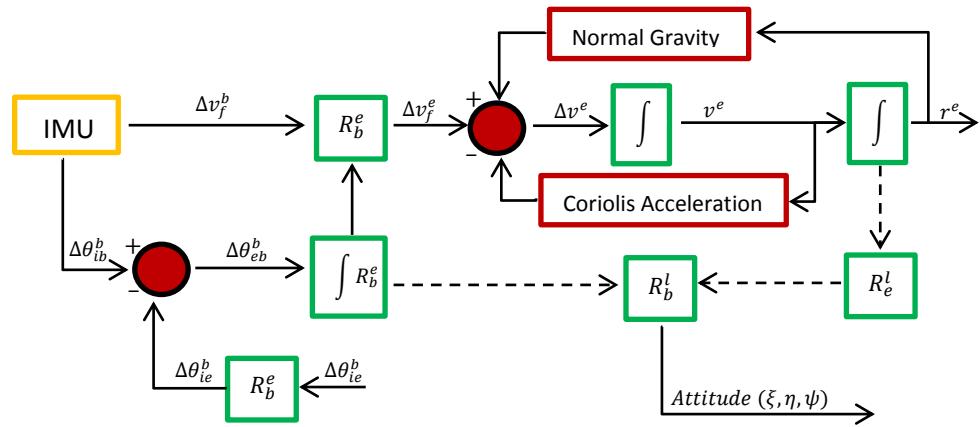


Figure 2-8: e-frame INS mechanization (adapted from Godha, 2006)

2.7 Inertial Sensor Errors

The mechanization equation process from the previous section is the dead-reckoning part of the INS system integrated with an IMU. If the accelerometers and gyroscopes were free of errors then we would be pretty much done. However that is not the case. As per El-Sheimy (2012), all sensors are subject to errors which limit the accuracy with which the observable can be measured. Recall that accelerometer and gyroscope bias were removed in Equations 2.16 and 2.17. These are but a couple of the errors affecting accelerometers and gyroscopes that will be further discussed in this section. As can be expected, very low-level MEMS such as the ones inside the Moto-X Android smartphone are going to have much higher errors than MEMS used in the industry such as the ADIS16488, STIM300, HG1930, let alone non-MEM IMUs such as the IMU-CPT, KVH1750, or LN200 (NovAtel Inc, 2015e).

The performance of accelerometers and gyroscopes inside an IMU can be quantified through bias offsets, scale factors, bias drift, non-orthogonality of the axes and noise. The following equations show the accelerometer and gyroscope measurement models:

$$\tilde{\mathbf{w}} = \mathbf{w} + \mathbf{b}_g + \mathbf{w} * \mathbf{S}_g + \mathbf{w} * \mathbf{N}_g + \boldsymbol{\varepsilon}_g \quad (2.36)$$

$$\tilde{\mathbf{f}} = \mathbf{f} + \mathbf{b}_a + \mathbf{w} * \mathbf{S}_a + \mathbf{w} * \mathbf{N}_a + \boldsymbol{\varepsilon}_a \quad (2.37)$$

Where the ‘~’ overscript represents the measured readings and the ‘a’ and ‘g’ subscript represent accelerometer and gyroscope respectively. The ‘w’ and ‘f’ represent the angular rate and specific force respectively. The remaining symbols are:

b = sensor bias offset,

S = matrix representing scale factor error

N = matrix representing axes non-orthogonality

ε = Sensor noise

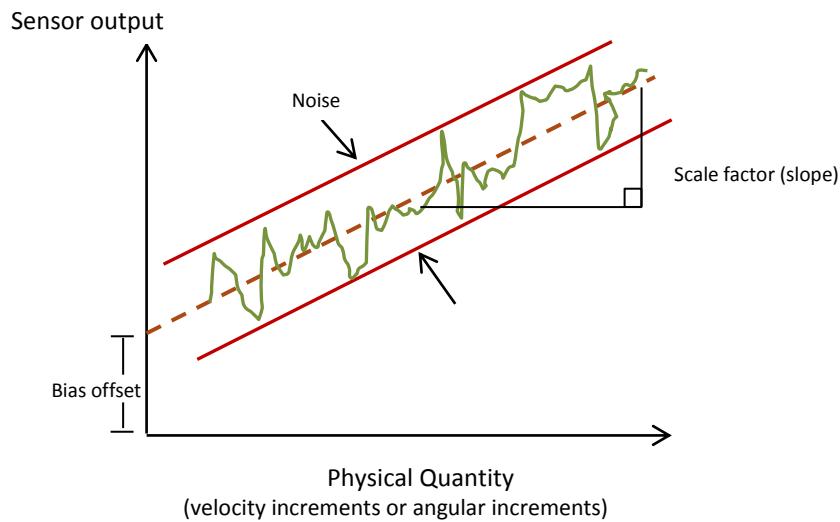


Figure 2-9: Visual of bias offset, noise and scale factor

Figure 2-9 above is a very good visual for explaining what the bias offset, noise and scale factors are. The X-axis represents the true, error-free quantity being measured (either velocity increments for accelerometers or angular increments for gyroscopes). The Y-axis represents the actual sensor measurement. Ideally, if the sensors were errorless we would expect the ratio slope of the line to be 1 (dashed line in diagram), provided the sensor output has the same units as the physical quantity being measured (Langley, 2016). The fact the slope does not start at the origin of the plot indicates the measurement is in the presence of a bias. Also, the deviation from the ideal slope

indicates the noise in the measurements and sensor drift. Changing this figure a bit will help explain what a bias drift is as shown below in Figure 2-10 .

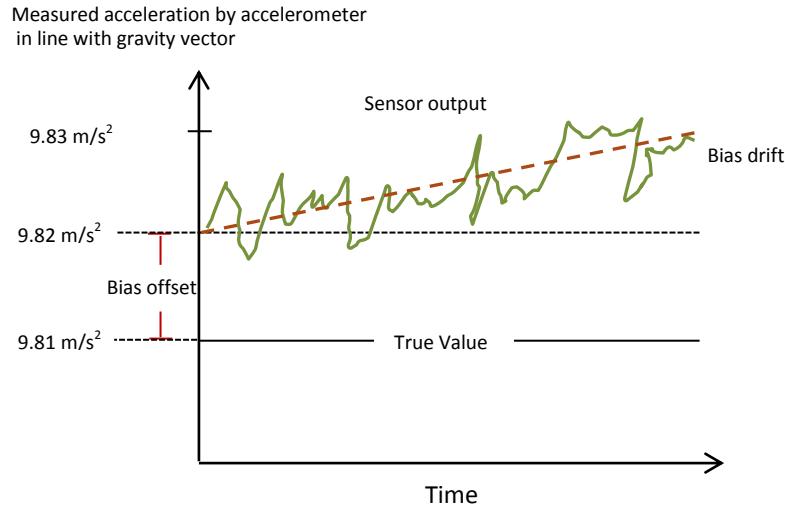


Figure 2-10: Bias drift

The x-axis in the figure above now represents time whereas the y-axis represents the measured acceleration by an accelerometer in line with the gravity vector. The example shows the accelerometer has a bias offset of 0.01 m/s^2 at the very start and a bias offset of 0.02 m/s^2 by the end of the plot. The difference in the bias at the end of the plot with respect to the start represents the bias drift. Let us now take a deeper look at each error.

2.7.1 Sensor Bias Offset

In literature, it is sometimes referred to as turn-on bias because of the fact it represents the bias in the sensor output when the unit is powered on. It is expressed in

$^{\circ}/\text{h}$ (or rad/s) for gyroscopes and in m/s^2 (or mg) for accelerometers. The sensor bias offset is a deterministic error that can be removed through a calibration procedure such as the six position test or the turntable calibration. The former consists of measuring raw data in six different positions to cancel out other errors and the latter consists on putting the IMU on a multi-axis turntable whose axes are oriented precisely to the l-frame. The repeatability of the sensor bias offset from turn-on to turn-on is very accurately determined for industry-used IMUs (both low and high grade). These errors are so small in tactical and navigation grade IMUs that their effects are negligible (Nayak, 2000). However, this is not the case with industry-used MEMS and therefore must be taken into account. As will be mentioned in the Results section of this thesis this wasn't the case with the ultra-low-end sensors inside the Moto-X smartphone. This deterministic bias offset was not repeatable from power cycle to power cycle and its effect can easily break the solution if not properly removed.

2.7.2 Scale Factor Error

As was stated earlier in the explanation for Figure 2-9, the scale factor represents the ratio of sensor output to true quantity were 1 is the ratio of an ideal sensor. The scale factor is sometimes divided into linear and a non-linear component due to the fact the inertial sensor's response is not exactly linear (El-Sheimy, 2012). The scale factor errors are measured in PPM and are negligible in tactical and navigation grade IMUs but not MEMS. As was the case with the sensor bias offset, the scale factor errors are deterministic and can be removed through the six position test or the turntable

calibration. The scale factor errors can change with time in MEMS in which case they are modelled stochastically. The scale factor errors proved to be quite important in the ultra-low-end sensors inside the Moto-X smartphone and had to be found empirically.

2.7.3 Non-orthogonality of the axes

This error results from imperfection of the mounting of the sensors and is sometimes referred to as the axes misalignment error. The result is that the axes are not completely independent of one another in the body frame (El-Sheimy, 2012). This error can be removed through the six position test or the turntable calibration.

2.7.4 Sensor noise

Noise present in the raw signal of accelerometers and gyroscopes adds an additional signal to the output signals being measured (El-Sheimy, 2012). Unlike the bias and scale factor, noise cannot be determined deterministically and instead must be modelled stochastically. Noise can be modelled in different ways depending on the stochastic characteristics it presents. For example, it can be modelled as white, random constant (bias), random walk, periodic random process, or Gauss Markov. Accelerometers and gyroscopes inside an IMU are usually modelled as a Gauss Markov process due to its simple mathematical description and ease of implementation. There are actually two types of noise that have to be modelled: sensor noise (high frequency component) and

bias noise (low frequency component). The high frequency component can be easily found by measuring the standard deviation of each sensor axis over a small amount of time (usually one or two seconds). The reason for this is that industry-used IMUs operate at a frequency >100 Hz thus a couple of seconds give enough samples to come with an estimate but it's also relatively a small time sample where the longer frequency noise is not taken into account. The low frequency noise component is attributed to the bias drift noise (and scale factor in MEMS) and is a bit more complicated to model. It is usually separated from the high frequency noise through wavelet decomposition where the level of decomposition is determined by the level at which the standard deviation of the remaining low frequency signal component reaches a minimum (Nassar, 2003). However, as was shown by Nassar (2003), and as will be explained in Chapter 6 of this thesis, INS sensor noise cannot be accurately modelled as a Gauss Markov process and instead ends up being empirically found.

2.8 Classification of IMUs

IMUs can be classified into different grades according to the magnitude of errors shown in the previous section. These sensors, in increasing order of quality, are referred to in the industry as entry level, mid performance level, and high performance level (NovAtel Inc, 2015e). Given the advances in MEMS technology over the last few years, industry-used MEMS have been grouped into the same grade as FOG technology IMUs such as the IMU-CPT from KVH. Recall that this thesis deals with a MEMS of much more lower quality that is not currently used for inertial navigation. As such, a new

grade henceforth referred to as recreation grade MEMS will be used to refer to MEMS such as those found inside smartphones.

Table 2-2: IMU grade comparisons (adapted from NovAtel Inc, 2015e)

IMU Grades	Recreation	Entry	Mid Performance	High Performance
Gyro Bias (deg/hr)	> 720	720 – 20	2 – 5	<= 1
Angular Random Walk (degrees/rt-hour)	> 0.3	0.0667 - 0.3	0.012 - 0.125	0.07 – 0.1
Accelerometer Bias (mg)	>50	>50	1- 2	<= 1
Measurement Rate (Hz)	<50	100- 200	100- 200	>=200
Technology	MEMS	MEMS, FOG	Ring Laser Gyro, FOG	FOG
Examples	Smartphone	IMU-CPT, ADIS16488	KVH-1750, HG1700-AG62	LN200, FSAS

Table 2-2 above shows some of the metrics used to differentiate amongst the different grades of IMU, the most important one being gyroscope quality. Although varying quality of accelerometers do exist, gravity is a very strong signal easy to pick up even with recreation-level accelerometers. As will be seen in the next chapter, accelerometers are used to measure the pitch and roll of the vehicle. Granted, high performance IMUs will be able to measure gravity more accurately than entry level

IMUs but the accelerometer quality wouldn't be the differentiator in choosing the IMU needed for a specific application.

Gyrosopes however, are the main differentiator for each IMU grade level. Gyrosopes that have a sensor bias offset greater than 15 deg/hour are not able to sense the Earth rotation rate which is a very important limitation as far as how to initialize the INS as will be seen in the next section. Gyroscope quality directly translates to how accurately heading can be measured. Barring the use of aided-heading in the system, a low quality IMU won't be able to determine its heading statically. Instead, the heading will depend on the kinematics the system experiences. If changes in heading are not observed over a long period of time, the system will lose knowledge of heading and the estimate will drift according to the quality of the gyroscope.

The angular random walk shown in Table 1 represents the noise level created by the gyroscope. The higher the quality of the IMU, the easier it is to estimate its angular random walk and any other noise level/errors due to the sensors being more stable. On the other hand, as will be seen in the Results section of this thesis, the recreation-level sensors inside the smartphone proved to have very unstable sensors from turn-on to turn-on making it very tough to accurately model its stochastic properties. The changing gyroscope bias offsets and angular random walk from turn-on to turn-on the reasons why the recreation sensor was only able to observe heading accurately to 15 degrees at best.

2.9 Initial Alignment

As was briefly alluded to when the mechanization equations were presented, one of the most important points in INS is the initial alignment. This can be considered an INS initialization process. Recall that INS sensors provide very accurate relative estimates due to the epoch-to-epoch integration taking place in the mechanization equations. However, they need an initial estimate to know what to add the accurate angular and velocity increments to. The initial alignment consists of providing an initial position, velocity and attitude, along with corresponding uncertainties for each estimate. Providing erroneous uncertainties to the estimates could potentially break the filter. Let us now talk about the different methods used in the industry to align an INS.

The initial absolute position and velocities are fed through GNSS. Lately, there have been more and more indoor navigation tests that require use of RFID technology in order to determine the offset with respect to a well-known point (whose position is accurately known by GNSS). Although mathematically possible, these indoor navigation projects haven't hit the mainstream market as outdoor projects have.

Pitch and roll are easily determined by accelerometers (from ultra-low-end to high-end IMUs) because they are solely dependent on the gravity vector (El-Sheimy 2012) as follows. Suppose an orthogonal triad of accelerometers is sitting on a parked car on a slope. The specific forces on the x-axis and y-axis can be related to pitch and roll as follows:

$$f_x = g * \sin(\xi) \quad (2.38)$$

$$f_y = g * \sin(\eta) \quad (2.39)$$

Where f_x and f_y correspond to the specific forces in the x-axis and y-axis respectively, g is the gravity scalar, and ξ, η are the rotations around the x-axis (pitch) and y-axis (roll) respectively. The formulation is a bit more complicated when the IMU is in motion because the specific forces will now measure vehicle motion as well as gravity but the principle is in essence the same. By modelling what gravity is at each point we are able to isolate the specific force components that are not related to vehicle dynamics and can therefore estimate pitch and roll.

The accuracy of the pitch and roll estimates is dependent on the accelerometer biases as follows (Nayak, 2000):

$$\sigma\xi = \frac{b_{ax}}{\gamma} \quad (2.40)$$

$$\sigma\eta = \frac{b_{ay}}{\gamma} \quad (2.41)$$

The heading (or azimuth) is the hardest component to estimate because of a couple of facts. The first one is that heading is not directly observed (unlike gravity). Instead, heading is indirectly observed through vehicle kinematics (more on this later). The second reason is that only tactical and navigation grade IMUs containing a gyroscope bias $< 15^{\circ}/hr$ are able to sense the Earth's rotation. That is, a navigation grade IMU such as an IMU-LN200 is sensitive enough to Earth's rotation and is therefore able to statically tell where its axes are pointing with respect to North. However, MEMS such as the Sensoonor STIM300 (or the ultra-low entry sensors inside the Moto-X smartphone for that matter), have gyroscope biases greater than the $15^{\circ}/hr$ meaning they are not able to

discern where their axes are pointing with respect to North statically. Thus the methods for coming up with the initial heading estimates in INS are:

1. Kinematic Alignment: Can be applied to any IMU but is usually used for the lower end sensors unable to sense Earth's rotation. The kinematic alignment is a fairly simple process. The course made good from the vehicle's change in position over consecutive epochs is injected as an initial INS estimate.

$$\psi_{GNSS} = \tan^{-1} \left[\frac{E_k - E_{k-1}}{N_k - N_{k-1}} \right] \quad (2.42)$$

Or, since GNSS already provides velocity through Doppler or carrier-phase measurements, we can use a velocity estimate directly:

$$\psi_{GNSS} = \tan^{-1} \left[\frac{v_E}{v_N} \right] \quad (2.43)$$

By doing covariance propagation on Equation 2.43 we can come up with the accuracy of the heading estimate:

$$\sigma^2 \psi_{GNSS} = \frac{v_E^2}{(v_E^2 + v_N^2)^2} \sigma^2 v_N + \frac{v_N^2}{(v_E^2 + v_N^2)^2} \sigma^2 v_E = \frac{\sigma^2 v}{v^2} \quad (2.44)$$

Where $\sigma^2 v$ is the variance in the horizontal velocity vector and v^2 is the horizontal velocity. As per Equation 2.44, the greater the horizontal velocity the smaller the heading estimate.

2. Static Alignment

For IMUs that do have a gyroscope bias of less than 15 °/hr, it is possible to align the INS statically. In the static alignment, pitch and roll are averaged from Equations 2.40, 2.41. Heading is found by taking the inverse tangent of the average of the raw angular increments after having been rotated from the body to the horizontal. This is shown in Equations 2.45, 2.46 below (Petovello, 2003):

$$\Delta\bar{\theta}_{ib}^h = R1(-\eta)R2(-\xi)\Delta\bar{\theta}_{ib}^b \quad (2.45)$$

$$\psi = -\tan^{-1}\left(\frac{(\Delta\bar{\theta}_{ib}^h)_x}{(\Delta\bar{\theta}_{ib}^h)_y}\right) \quad (2.46)$$

Where the bar corresponds to the average values over a specified amount of time (usually one minute).

3. Aided Heading

This is the optimum method for providing the initial INS heading estimate due to the fact it is instantaneous and can be used with low-end or high-end INS sensors. The idea is to feed the initial heading from an external source. Usually, this aided heading comes from a secondary GNSS receiver-antenna pair that provides a heading estimate with respect to the master GNSS receiver-antenna pair (NovAtel Inc, 2015d). The dual antenna GNSS heading also helps constrain the heading drift (more on that on Chapter 4). Lately, there has also been research on using magnetometers for providing an aided heading estimate. Tang (2014) showed a filtered static magnetometer-derived solution form the

commercial product 3DM-GX1 had an accuracy of 0.7° when in the absence of magnetic anomalies and 4.7° when in the presence of magnetic disturbances. As will be further explored in this thesis, the magnetometer inside Moto-X smartphone proved to be very unreliable and as such was not used for heading even after calibrating for soft-iron and hard-iron effects.

CHAPTER 3: INTRODUCTION TO DATA PROCESSING METHODS

This chapter provides an introduction to data processing methods used in this thesis, focusing with the Kalman Filter. First, let us start with the most basic of estimation processes. Growing up we are taught how to solve for unknowns using mathematical equations that relate known information with quantities trying to be solved. We are also taught that in order to find a unique solution, we need the number of equations (observables) to be equal to the number of unknowns. If we are dealing with a linear system of equations we can then represent the equations in the following matrix form:

$$\mathbf{z} = \mathbf{H}\mathbf{x} \quad (3.1)$$

where,

\mathbf{z} is a vector representing the measurements

\mathbf{H} is the design matrix (Jacobian) which represents the derivatives (geometry) of the functions with respect to the unknowns

\mathbf{x} is the vector of unknowns

This works really well mathematically except that in real life we usually cannot directly measure the quantity of interest and instead must compute it from other known information; thus measurement and hence the variable of interest isn't perfectly determined. Measurements contain blunders, systematic biases due to equipment and/or measuring techniques, and random errors (El-Sheemy, 2008). Systematic errors and blunders can be removed if known, but the measurements will still have random errors.

These errors cannot be computed deterministically and instead must be modelled statistically. Equation 3.1 would then become:

$$\mathbf{z} = \mathbf{Hx} + \boldsymbol{\nu} \quad (3.2)$$

where $\boldsymbol{\nu}$ is the random error in the observations caused by measurement noise.

3.1 Estimation using Measurements Only

Let us consider an example in which the length of a table is desired (thus only one unknown). If the length of the table is measured 10 times with a measuring tape (giving us nine redundant observations), most of the measurements are more than likely going to differ from one another due to the errors mentioned earlier. Some observations will be very different from others (due to blunders), whereas a systematic bias may be inherent in all measurements due to the observation technique as well as material expansion/contraction. The best possible estimate in this example (table length) is the arithmetic mean of the measurements (sans blunders and systematic biases). The reason for this is that the mean is the least squares estimate for a group of measurements of a certain parameter. The least squares estimate is that in which in addition to satisfying the mathematical model, minimizes the weighted sum of squares of the residuals. This condition satisfies the properties of the best estimate which is most probable (maximum likelihood), most precise (minimum variance), and most accurate (unbiased) (El-Sheimy, 2008). The residuals are the differences between the model and the measurements ($\boldsymbol{\nu}$ in Eq. 3.2). Thus, the least square principle is to minimize the

difference between observations and the estimates of the state vector. Going back to the purely mathematical example posted earlier, if we had four equations and three unknowns there is more than one solution that meets the model, but there is only one in which the sum of squares of the residuals is a minimum. On the other hand, a system of equations with the same number of equations (knowns, m) as unknowns (n) would only have one unique solution and thus no adjusting of observations is necessary.

It should be noted that in the real world we usually deal with multivariate ($n > 1$) instead of univariate ($n = 1$) problems. Also, ideally we want as many redundant observations as possible as it means having a better estimate of the parameter(s) of interest. Thus, the idealistic arithmetic mean procedure of above does not apply to over-determined mathematical models in which $m > n$ and $n > 1$.

The general least squares expression can be derived as follows. Recall that the least squares principle minimizes the weighted sum of squares of residuals. From Eq. 3.2 ,

$$\mathbf{v} = \mathbf{z} - \mathbf{H}\hat{\mathbf{x}} \quad (3.3)$$

where $\hat{\mathbf{x}}$ is the estimate of the parameters. Thus the cost function, J , which is the function to be minimized is:

$$J = (\mathbf{z} - \mathbf{H}\hat{\mathbf{x}})^T \mathbf{W}^{-1} (\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}) \quad (3.4)$$

where \mathbf{W} is the weighting function used. Recall that a function is minimized by taking the derivative and setting it to zero. Thus, if we take the derivative of Equation 3.4, set it to zero and solve for $\hat{\mathbf{x}}$ we end up with:

$$\hat{\mathbf{x}} = (\mathbf{H}^T \mathbf{W}^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W}^{-1} \mathbf{z} \quad (3.5)$$

Applying the law of covariance propagation, the covariance of the unknown (state) vector, $\mathbf{C}_{\hat{\mathbf{x}}}$ is:

$$\mathbf{C}_{\hat{\mathbf{x}}} = (\mathbf{H}^T \mathbf{W}^{-1} \mathbf{H})^{-1} (\mathbf{H}^T \mathbf{W}^{-1} \mathbf{C}_z \mathbf{W}^{-1} \mathbf{H}) (\mathbf{H}^T \mathbf{W}^{-1} \mathbf{H})^{-1} \quad (3.6)$$

where \mathbf{C}_z is the covariance of the observation vector. Setting \mathbf{C}_z as the weighting function \mathbf{W} simplifies Eq. 3.5 and 3.6 to:

$$\hat{\mathbf{x}} = (\mathbf{H}^T \mathbf{C}_z^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{C}_z^{-1} \mathbf{z} \quad (3.7)$$

$$\mathbf{C}_{\hat{\mathbf{x}}} = (\mathbf{H}^T \mathbf{C}_z^{-1} \mathbf{H})^{-1} \quad (3.8)$$

The least squares estimation shown above is used when dealing with time invariant parameters which can be fully modelled through measurements (e.g. geodetic networks). An important concept to keep in mind is that least squares estimation can be used for linear and non-linear mathematical models. The linearization takes place during the computation of the design matrix, which as explained earlier is computed through taking the derivatives of the functions with respect to the unknowns. Non-linear models require multiple iterations to converge, where the number of iterations depends on the closeness of the initial estimate of the states with respect to the true value of the states. If the initial estimate (point of expansion, POE) is not chosen wisely, the adjustment might converge to a different answer. Thus, the design matrix is constantly changing from iteration to iteration. It can also be expected to see the residuals get smaller and smaller with every iteration until the solution converges. On the other hand, linear models consist of constant design matrices meaning the solution will converge in the first iteration regardless of the POE chosen (El-Sheimy, 2008).

3.2 Estimation using Dynamic Modelling

The least square adjustment mentioned above is used in estimation processes that can be fully modelled by measurements. That is, the state vectors are time invariant. However, many real world processes and mathematical models aren't. One such example is GNSS/INS as it consists in estimating the position, velocity and attitude of a moving platform. Time variant models then need a dynamics model that describes how the system behaves over time. A continuous time-variant model is modelled through:

$$\text{Dynamics model} \quad \dot{\mathbf{x}} = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{G}(t)\mathbf{w}(t) \quad (3.9)$$

$$\text{Measurement model} \quad \mathbf{z}(t) = \mathbf{H}(t)\mathbf{x}(t) + \mathbf{v}(t) \quad (3.10)$$

where,

$\mathbf{F}(t)$ is the dynamics matrix describing the kinematics of the system at time t .

$\mathbf{x}(t)$ is the state vector at time t .

$\dot{\mathbf{x}}(t)$ is the time derivative of the state vector at time t

$\mathbf{G}(t)$ is the shaping matrix of system noise input at time t

All other quantities are the same as Eq. 3.2 at time t .

The discrete equivalent of Equations 3.9 and 3.10, which is what will be used throughout this thesis, is:

Dynamics model $\boldsymbol{x}_{k+1} = \boldsymbol{\Phi}_{k+1,k} \boldsymbol{x}_k + \boldsymbol{G}_k \boldsymbol{w}_k$ (3.11)

Measurement model $\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k$ (3.12)

where

$\boldsymbol{\Phi}_{k+1,k}$ is the discrete state transition matrix from epoch k to $k+1$.

\boldsymbol{x}_k is the state vector at epoch k

\boldsymbol{x}_{k+1} is the state vector at the epoch $k+1$

\boldsymbol{G}_k is the shaping matrix of the process noise at epoch k

\boldsymbol{w}_k is the process noise at the epoch k with a covariance matrix, \boldsymbol{Q}_k

\boldsymbol{z}_k is the measurement vector at epoch k

\boldsymbol{H}_k is the design matrix relating the state vector to the measurements at epoch k

\boldsymbol{v}_k is the zero-mean measurement noise at epoch k with a covariance matrix, \boldsymbol{R}_k

The discrete transition matrix, $\boldsymbol{\Phi}_{k+1,k}$, can be computed from the continuous dynamics matrix, $\boldsymbol{F}(t)$, provided the assumption the system does not change over the time transition interval. That is, $\boldsymbol{\Phi}$ is fixed from epoch k to $k+1$. The computation is through a Taylor series expansion as follows (Brown and Hwang 1997).

$$\boldsymbol{\Phi} \approx e^{\boldsymbol{F}\Delta t} = \boldsymbol{I} + \boldsymbol{F}\Delta t + \frac{(\boldsymbol{F}\Delta t)^2}{2!} + \frac{(\boldsymbol{F}\Delta t)^3}{3!} + \dots \quad (3.13)$$

where \boldsymbol{I} is the identity matrix and Δt is the time interval. Should the assumption above not apply, the error can be mitigated by shortening Δt (Petovello, 2003).

The discrete process noise \mathbf{w}_k is uncorrelated zero mean with Gaussian distribution:

$$\begin{cases} E[\mathbf{w}_k] = 0 \\ E[(\mathbf{w}_k)_i(\mathbf{w}_k)^T_j] = \begin{cases} \mathbf{Q}_k, i = j \\ 0, i \neq j \end{cases} \end{cases} \quad (3.14)$$

A numerical algorithm presented by Grewal and Andrews. (2001) for computing \mathbf{Q}_k is

$$\mathbf{Q}_k = E[\mathbf{w}_k \mathbf{w}_k^T] = 0.5 * \Delta t (\Phi_{k+1,k} \mathbf{G}_k \mathbf{Q} \mathbf{G}_k^T \Phi_{k+1,k}^T + \mathbf{G}_k \mathbf{Q} \mathbf{G}_k^T) \quad (3.15)$$

where \mathbf{Q} is the matrix describing the spectral densities of the noise in the system.

The covariance matrix of the measurements \mathbf{R}_k is defined similarly to the covariance matrix of the process noise:

$$\begin{cases} E[\mathbf{v}_k] = 0 \\ E[(\mathbf{v}_k)_i(\mathbf{v}_k)^T_j] = \begin{cases} \mathbf{R}_k, i = j \\ 0, i \neq j \end{cases} \end{cases} \quad (3.16)$$

3.3 Kalman Filter Algorithm

The Kalman filter makes use of the dynamics and measurement models above recursively to optimize the estimate of the state vector by minimizing the variance (Gao and Sideris, 2007). The Kalman filter is optimal with respect to virtually any criterion that makes sense due to the fact it processes all available measurements regardless of their precision to estimate the current values of the state vector (ibid). This is accomplished through knowledge of system and measurement dynamics, stochastic description of uncertainty of dynamics, measurement errors and system noises, and

initial information about the variables being estimated. The recursive behaviour of the Kalman filter means it does not need knowledge of all previous data. This is possible because the system dynamics are predicted and corrected at each epoch, with each computation already containing previous information about the filter's behaviour and accuracy. This is very similar to the sequential least squares formulation, which considers a new batch of observations in conjunction with a previously computed solution. In fact, the sequential least squares formulation is closely related to the formulation presented in the Kalman filter (O'Keefe, 2008).

The first requirement in the Kalman filter is to have an initial estimate of the states being assessed as well as a corresponding covariance. The covariance is very important because it will determine by how much these initial estimates are trusted. If the covariance is high then the filter won't trust the initial conditions much and will instead rely on the quality of subsequent measurements as well as the estimation of the kinematic model to drive itself.

After the initial conditions have been selected, the filter runs through a two stage processes consisting of prediction and updates. The prediction stage is run at epochs where measurements are not present, whereas the update stage takes place when the system has measurement knowledge. The measurement variances also have to be known in order for the filter to decide to what degree the observations are to be trusted. High accuracy measurements keep the filter in check by indirectly providing knowledge of actual system dynamics. Low accuracy measurements are weighted much less and the filter's predicted kinematics are trusted more. Let us take a closer look at the expressions used in the Kalman filter.

The prediction stage first consists of predicting the error state using knowledge of the previous epoch:

$$\hat{\mathbf{x}}_k(-) = \Phi_{k,k-1} \hat{\mathbf{x}}_{k-1}(+) \quad (3.17)$$

where

$\hat{\mathbf{x}}_k(-)$ is the predicted estimate of the state vector at epoch k

$\Phi_{k,k-1}$ is the transition matrix describing dynamics from epoch $k-1$ to k

$\hat{\mathbf{x}}_{k-1}(+)$ is the updated estimate of the state vector at epoch $k-1$

As can be seen from Eq. 3.17, the predicted state uses a priori knowledge of the last known state update as well as the kinematic model of the system. It is entirely possible $\hat{\mathbf{x}}_{k-1}(+)$ came from a prediction rather than an update (i.e., prediction taking place in between updates) in which case it would be $\hat{\mathbf{x}}_{k-1}(-)$. As has been hinted before, the Kalman filter requires knowledge of the quality of the estimates throughout. As such, we need to know the quality of the predicted state at $\hat{\mathbf{x}}_k(-)$ which is computed as follows:

$$\mathbf{P}_k(-) = \Phi_{k,k-1} \mathbf{P}_{k-1}(+) \Phi_{k,k-1}^T + \mathbf{Q}_{k-1} \quad (3.18)$$

where \mathbf{Q}_{k-1} is as defined in Eq. 3.15 and

$\mathbf{P}_k(-)$ is the predicted covariance of the states at epoch k

$\mathbf{P}_{k-1}(+)$ is the updated covariance of the states at epoch $k-1$

Equation 3.18 shows that $\mathbf{P}_k(-)$ depends on the sum of squares of the transition matrix, which is weighted by the previous state's covariance, $\mathbf{P}_{k-1}(+)$ as well as the

system process noise. The system's noise spectral densities, which are the backbone of the process noise, are arguably the most difficult part of the Kalman filter.

As mentioned earlier, the update part of the Kalman filter is applied whenever new observations are fed into the filter. The update part first consists in computing a gain matrix, \mathbf{K}_k , that determines to what degree the incoming observations are trusted. As such, the gain matrix is optimized to produce the minimum error variance (Godha, 2006).

$$\mathbf{K}_k = \mathbf{P}_k(-) \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k(-) \mathbf{H}_k^T + \mathbf{R}_k)^{-1} \quad (3.19)$$

As can be seen in Equation 3.19, the measurement covariance, \mathbf{R}_k , is the main contributor in determining \mathbf{K}_k . The higher \mathbf{R}_k , the smaller \mathbf{K}_k and vice-versa. The gain matrix is also a function of the current geometry and the predicted state covariance found in Eq. 3.18.

The next part of the update is to compute the difference between the actual observations, \mathbf{z}_k , and the predicted observations, $\hat{\mathbf{z}}_k$ (Godha, 2006). This is referred to as the innovation sequence and can be thought of as the residuals, \mathbf{V}_k , for the current epoch. As can be seen below, this is possible by mapping the predicted state into the observation plane via the design matrix:

$$\mathbf{V}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_k(-) \quad (3.20)$$

The updated states are then found by adding the current state estimate to the residuals after being weighted by the gain matrix:

$$\hat{\mathbf{x}}_k(+) = \hat{\mathbf{x}}_k(-) + \mathbf{K}_k \mathbf{V}_k \quad (3.21)$$

Finally, the updated covariance of the system can be found as follows:

$$\mathbf{P}_k(+) = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k(-) \quad (3.22)$$

The step-by-step iterative Kalman Filter is shown below in Figure 3-1:

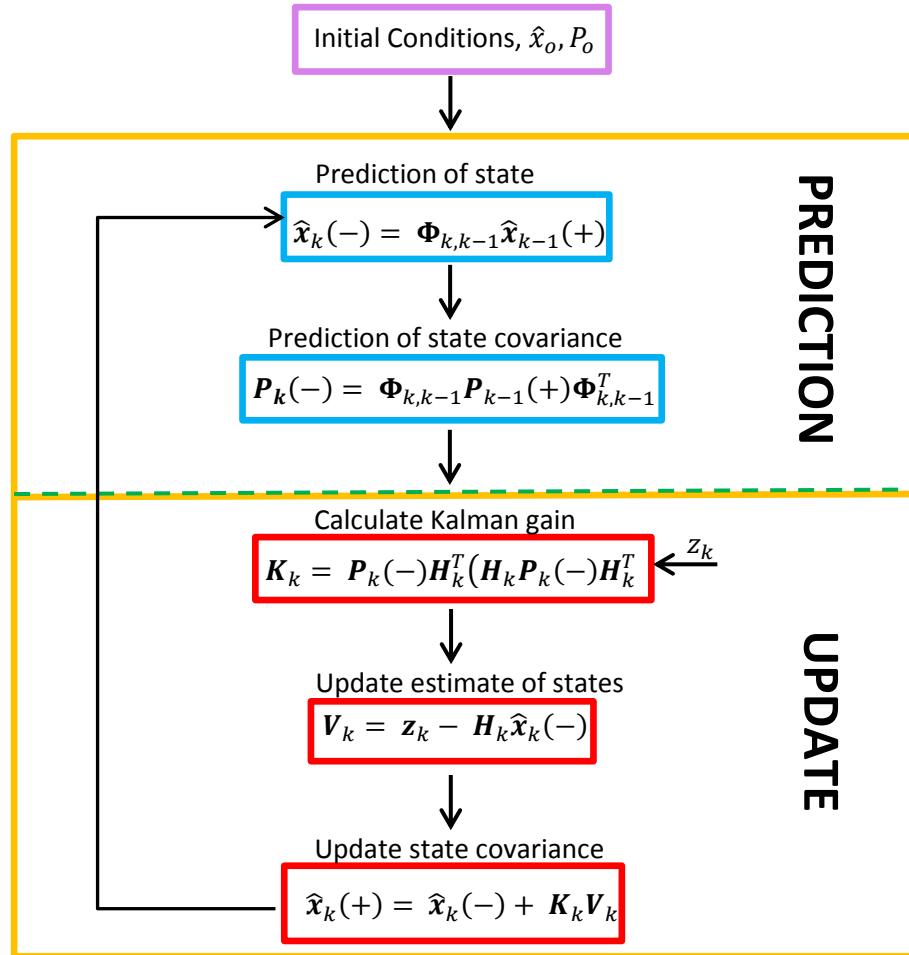


Figure 3-1: Discrete Kalman filter (adapted from El-Sheimy, 2012)

3.4 Kalman Filter in Non-Linear Systems

The Kalman filter, like the least squares adjustment is linear by nature. Thus, when dealing with non-linear functions such as the ones in INS equations, linearization is

required. The linearization is a first order Taylor series representation of the non-linear model about a point of interest which can be either a nominal value, or the last estimate in the Kalman filter (Petovello, 2003). A non-linear process and measurement model can be represented by:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, k) + \mathbf{w}_k \quad (3.23)$$

$$\mathbf{z}_{k+1} = h(\mathbf{x}_k, k + 1) + \mathbf{v}_{k+1} \quad (3.24)$$

where f and h are non-linear functions. Let \mathbf{x}^* be the point of interest at which the functions are linearized at such that:

$$\delta\mathbf{x}_{k+1} = \mathbf{x}_{k+1}^* - \mathbf{x}_{k+1} \quad (3.25)$$

Where $\delta\mathbf{x}_{k+1}$ is the perturbation from the point of linearization, and \mathbf{x}_{k+1} is the actual state. Eq. 3.25 holds true if the first order Taylor series expansion performed about the point of interest yields a small enough perturbation. If \mathbf{x}_{k+1}^* is chosen to be a nominal trajectory then a linearized Kalman filter (LKF) is being used. LKFs are then meant to be used offline where a nominal trajectory is known. However, one of the drawbacks of this implementation is that the deviation between actual and nominal trajectories can drift in time without bounds. This in turn conflicts with the small error assumption used in Eq. 3.25. The second method called the extended Kalman filter (EKF) consists in linearizing about the previous Kalman filter estimate and considering the perturbations as deviations between the estimated state and the actual state. Unlike the LKF, the EKF does not drift without bounds due to the fact the linearization trajectory is continuously updated with the estimated results. On the other hand, large initial uncertainties and measurement noise may lead to filter divergence (Gao, 2007).

Usually, the EKF implementation for INS applications has the state vector resetting after every update. This holds true because of the fact that the Kalman filter is an optimal filter with zero mean errors (Petovello, 2003).

Going back to Eq. 3.25, if the perturbations are small enough, the discrete dynamics and measurement models are represented by:

$$\delta \mathbf{x}_{k+1} = \Phi_{k+1,k} \delta \mathbf{x}_k + \mathbf{G}_k \mathbf{w}_k \quad (3.26)$$

$$\delta \mathbf{z}_k = \mathbf{H}_k \delta \mathbf{x}_k + \mathbf{v}_k \quad (3.27)$$

Thus, just like the state vector, measurements used in the filter are the differences between the estimated and actual measurements. Other than this linearization step, the Kalman filter process is the same as in Figure 3-1. The final state can then be computed by manipulating eq. 3.25 such that

$$\mathbf{x}_{k+1} = \mathbf{x}_{k+1}^* - \delta \mathbf{x}_{k+1} \quad (3.28)$$

The following block diagram, shown in Figure 3-2, is based on El-Sheimy (2012). It shows the EKF functionality in INS applications.

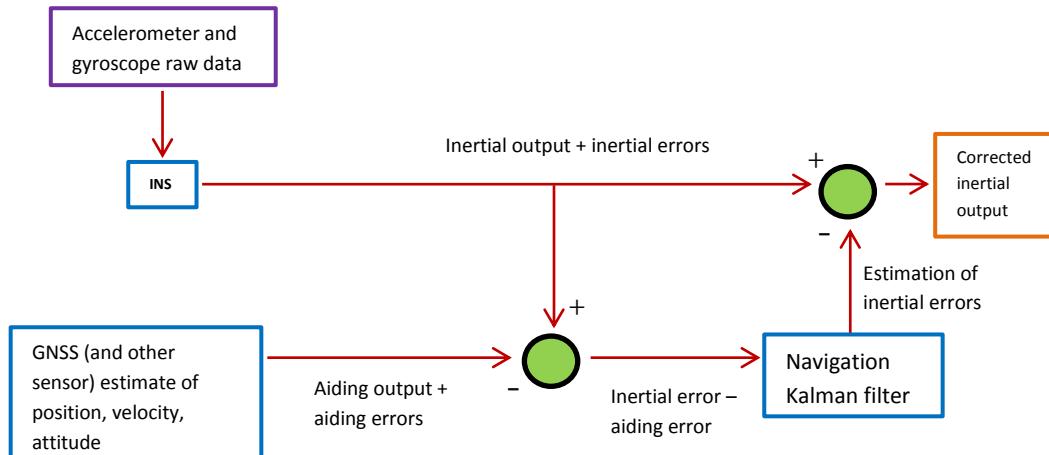


Figure 3-2: INS extended Kalman filter block diagram (adapted from El-Sheimy, 2012)

A final important point to note about the INS filter is that as shown in Figure 3-2, the measurements used in the EKF are actually the differences between the observations and the estimated observations. These observations are either in the form of positions, velocities, or raw GNSS measurements depending on the type of GNSS/INS integration which will be explained in the next chapter. However, regardless of the type of integration, one of the most important updates is called the zero velocity update (ZUPT). Zero velocity updates are important because by knowing the vehicle is stationary, the velocity errors in the filters can be constrained. Suppose for example the INS goes through an urban canyon where no satellites can be observed. As explained earlier, in the absence of GNSS, the filter will run in its prediction mode only and the INS errors will grow. However, by stopping the vehicle, the velocity errors are constrained. The position is already off at this point because no position updates have taken place, but the growth of the position error is going to be much lower than if the vehicle did not stop throughout the GNSS outages in the urban canyon.

CHAPTER 4: GNSS/INS INTEGRATION

The previous chapter explained the Kalman filter that needs to be run in order to optimize the INS errors. Recall that the Kalman filter operates in a two-step process consisting of predictions and updates. The dynamics model attempts to predict what the position, velocity, and attitude of the vehicle will be whereas the measurement model inherently keeps all errors in check during the update stage. The INS will operate in the prediction stage in the absence of observations and will drift at a rate depending on the quality of the INS sensors as well as how accurately the behaviour of the IMU and its errors are modelled. This chapter first looks into the two measurement coupling method followed by an explanation of the importance of properly measuring the IMU to GNSS antenna offset. This is followed by a detailed look at the matrices behind the EKF introduced in Chapter 3. Thirdly, an overview of the mathematics behind the aided heading using GNSS and magnetometers is presented. The chapter concludes with an introduction to vehicle motion constraints used to constrain INS errors when GNSS outages are present.

4.1 Integration Strategies

There are two integration strategies that are implemented in INS: tight coupling and loose coupling. The difference between both integration methods is in how GNSS updates are fed into the INS; both methods are discussed below.

4.1.1 Tight Coupling

In tightly coupled filtering, raw GNSS observations such as raw pseudorange measurements, carrier-phase measurements, and Doppler measurements are used in the filter update process (NovAtel Inc, 2015b). This then means that with this type of integration, both GNSS and INS updates are computed from within the same filter. The advantage of tightly coupled filtering is twofold: statistically rigorous sharing of information is provided amongst the filter states, and there is lower filter noise when compared to the loosely coupled solution (Petovello, 2003). However, tightly coupled filtering is heavier computation-wise due to the added states compared to the loosely coupled alternative. The biggest advantage of the tightly coupled filter is that a measurement update can be computed from two satellites as opposed to four which would be required from a loosely coupled integration. That is, a position can be computed from 2 satellites instead of 4. The reason being that each satellite provides more than one independent type of observation, with two satellites meeting the minimum four linearly independent observations required to compute a position. This makes this type of integration essential for challenging scenarios such as urban canyons. The tightly-coupled integration is shown below in Figure 4-1.

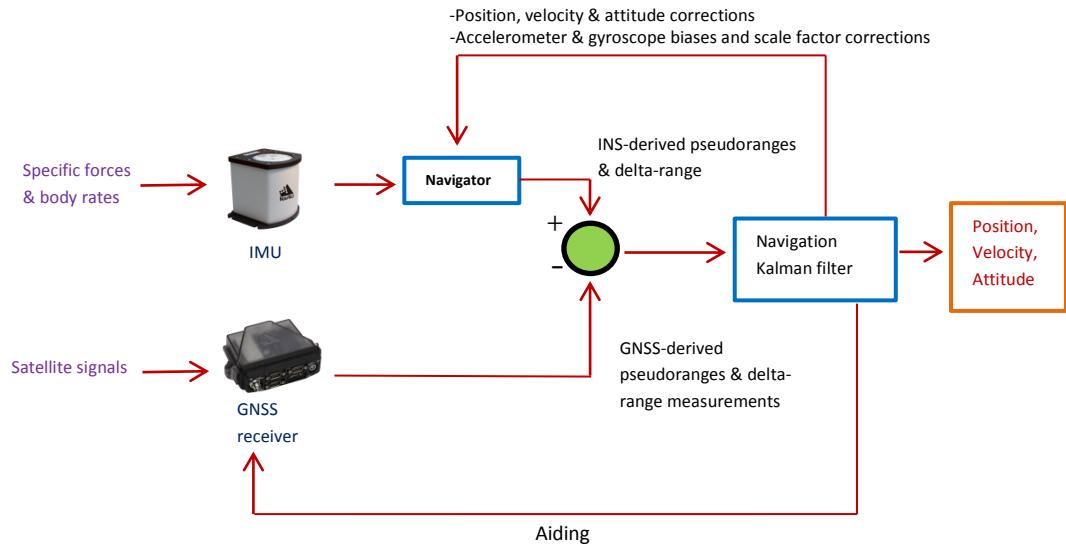


Figure 4-1: Tightly coupled integration (adapted from El-Sheimy, 2012)

4.1.2 Loose Coupling

A loosely coupled GNSS/INS integration consists in separating the GNSS from the INS filter. That is, the vehicle position and velocity are first computed by a GNSS filter. The results of this filter are then fed as updates to the INS filter in the form of position and/or velocity updates. The main advantage of the loosely coupled integration is that it can be implemented in systems that are not able to output raw GNSS measurements. For example, the Android smartphone used in this thesis research is only able to output GNSS positions and velocities meaning it can't be used in a tightly-coupled approach. The second advantage is that loose coupling is a relatively simple and flexible approach that lends itself to fusing other sensors. That is, the fact the INS is a stand-alone filter means this integration method does not require as many states in the Kalman filter.

during each update as tight coupling does. The big disadvantage of the loosely-coupled integration is that four satellites are required in order to provide an update. That is, if the vehicle were travelling in an urban canyon scenario where fewer than 4 satellites were in view, it would not be possible to provide an update to the filter. Another disadvantage is that having two separate filters increases noise. This can however be accounted for by increasing the weights of the GNSS updates prior to feeding them into the filter (Petovello, 2003). The loosely-coupled integration is shown below in Figure 4-2.

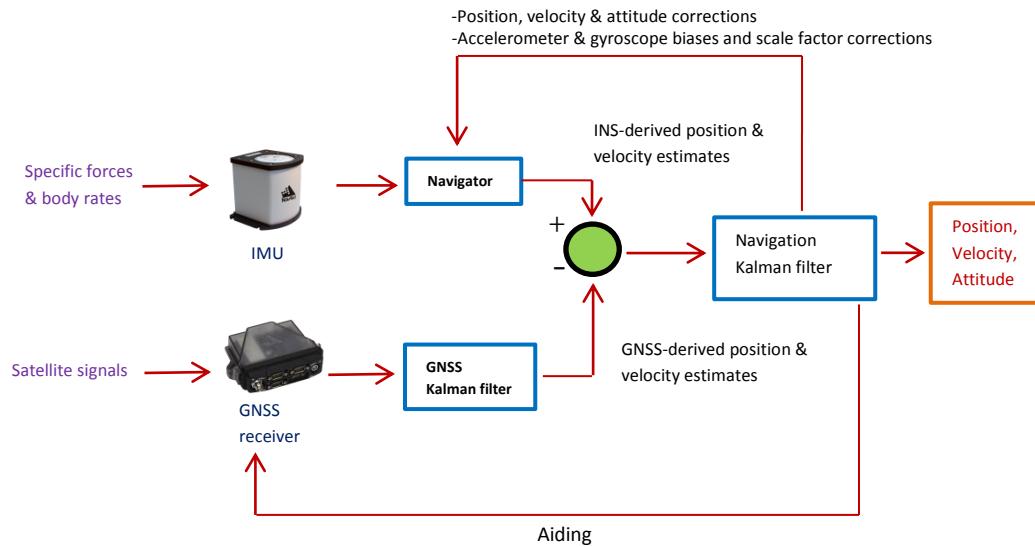


Figure 4-2: Loosely coupled integration (adapted from El-Sheimy, 2012)

As has been mentioned before, this thesis deals with loosely-coupled integration and as such no further details will be provided on tight coupling. Readers are invited to read Petovello (2003) and Godha (2006) for further details on how to construct the matrices used in the tightly-coupled integration.

4.2 Effect of Lever Arm in INS Solution

The lever arm is the vector that goes from the IMU centre of navigation (where the INS solution is computed with respect to) to the GNSS antenna phase centre. If using a dual GNSS setup, then both lever arms have to be known. Lever arm(s) need(s) to be taken into account in real-time in order to consider the fact the raw GNSS signals are not being received at the IMU centre of navigation. The longer the separation, the greater the effect of an uncompensated lever arm on position and velocity estimates. Let us first show the expressions used to take lever arm separations into account (Petovello, 2003):

$$\Delta \mathbf{r}_e = \mathbf{R}_b^e \mathbf{l}_b \quad (4.1)$$

$$\Delta \mathbf{v}_e = \boldsymbol{\Omega}_{ie}^e \mathbf{R}_b^e \mathbf{l}_b + \mathbf{R}_b^e \boldsymbol{\Omega}_{ib}^e \mathbf{l}_b \quad (4.2)$$

where \mathbf{l}_b represents the lever arm in the body frame. Thus, at every GNSS update (usually every second), the measured positions and velocities need to be translated to the IMU centre of navigation. What is the effect of an error in the lever arm computation? Or better yet, what is the required accuracy of the lever arm computation? Applying covariance propagation to Eqs. 4.1 and 4.2 we end up with:

$$\delta \Delta \mathbf{r}_e = \mathbf{R}_b^e \boldsymbol{\delta l}_b \quad (4.3)$$

$$\delta \Delta \mathbf{v}_e = \boldsymbol{\Omega}_{ie}^e \mathbf{R}_b^e \boldsymbol{\delta l}_b + \mathbf{R}_b^e \boldsymbol{\Omega}_{ib}^e \boldsymbol{\delta l}_b \quad (4.4)$$

Thus, the lever arm effect on position is dependent on the degree of accuracy of the estimated positions. In other words, the lever arm needs to be estimated as accurately as the positions estimated by the filter. However, from looking at Eq. 4.4 it can be seen that

the lever arm effect on velocity depends on the rate of rotation. Assuming a land vehicle that is capable of reaching 30 deg/sec, a 1 cm error in the lever arm can cause a velocity error of 5 mm/s (*ibid*). Thus, the lever arms should always be measured as accurately as possible using high quality instruments such as a total station. Alternatively, the lever arm error can be estimated throughout the lifetime of the filter, thus requiring an initial uncertainty to be input. If so, it is very important to make sure the lever arm uncertainty is valid. Specifying wrong lever arms with high weight could cause the filter to break (Bahan, 2015).

4.3 INS Filter

Chapter two showed the mechanization equations where position, velocity, and attitude estimates are found through integration of the IMU raw data. The process also involved accounting for the effects of turn-on biases, Coriolis force, gravity, and the Earth's rotation rate. However, just integrating the raw data will cause a drift in the solution because of the presence of the INS errors. The practical aspects of the EKF introduced in Chapter 3 are discussed in this section.

4.3.1 System Model

Recall from Chapter 3 that linearization about the current estimates in the EKF means the error state rather than the states themselves are computed. At the very minimum, the

EKF running the INS will require nine error states: three for position, three for velocity, and three for attitude. However, this would also mean that the INS error sources are completely known, which certainly isn't the case. The number of states required depends on the knowledge of the INS sensors being used which would directly depend on their quality. For example, scale factor and non-orthogonality errors are so small in high-end IMUs that they can be ignored. The same can't be said about MEMS IMUs. We will refer to the nine main states as the navigation error states and subsequent ones as INS error states. Only the loosely-coupled integration method will be shown.

4.3.1.1 Navigation Error States

The mechanization equations were shown to be the solution to Equation 2.15 shown here again for convenience:

$$\begin{pmatrix} \dot{\mathbf{r}}^e \\ \dot{\mathbf{v}}^e \\ \dot{\mathbf{R}}_b^e \end{pmatrix} = \begin{pmatrix} \mathbf{v}^e \\ \mathbf{R}_b^e \mathbf{f}^b - 2\Omega_{ie}^e \mathbf{v}^e + \boldsymbol{\gamma}^e \\ \mathbf{R}_b^e (\boldsymbol{\Omega}_{ei}^b + \boldsymbol{\Omega}_{ib}^b) \end{pmatrix}$$

Readers are invited to read Shin (2001), El-Sheimy (2012), and Savage (2000) for step-by-step derivations on the perturbations to the above expressions. The final result is:

$$\begin{pmatrix} \delta \dot{\mathbf{r}}^e \\ \delta \dot{\mathbf{v}}^e \\ \delta \dot{\mathbf{R}}_b^e \end{pmatrix} = \begin{pmatrix} \delta \mathbf{v}^e \\ \mathbf{R}_b^e \delta \mathbf{f}^b - 2\Omega_{ie}^e \delta \mathbf{v}^e + \mathbf{N}^e \delta \mathbf{r}^e - \mathbf{F}^e \boldsymbol{\varepsilon}^e \\ -\Omega_{ie}^e \boldsymbol{\varepsilon}_e + \mathbf{R}_b^e \delta \mathbf{w}_b \end{pmatrix} \quad (4.5)$$

where, the dot on top of the variable represents the rate of change, e corresponds to the e-frame, b corresponds to the b-frame, and i corresponds to the i-frame. The remaining symbols are:

δr is the position error state vector ($\delta r_x, \delta r_y, \delta r_z$)

δv is the velocity error state vector ($\delta v_x, \delta v_y, \delta v_z$)

ε is the misalignment error state vector ($\varepsilon_x, \varepsilon_y, \varepsilon_z$)

F is the skew symmetric matrix of the specific force (f_x, f_y, f_z)

N is the gravitational gradient coefficient matrix

Ω_{ie} is the skew symmetric matrix of the Earth's rotation rate from the i-frame to the e-frame

R_b^e is the rotation matrix from the b-frame to the e-frame

δf is the raw accelerometer measurement errors ($\delta f_x, \delta f_y, \delta f_z$)

δw is the raw gyroscope measurement errors ($\delta w_x, \delta w_y, \delta w_z$)

The gravitational gradient coefficient matrix, N , is shown in detail in Appendix A.

4.3.1.2 INS Sensor Error States

The navigation error states shown are the minimum number of states required in an INS filter. However as mentioned earlier we also have to take into account the fact that

not all of the INS sensor errors are completely known and instead must be modelled. Recall that for high-end IMUs the turn-on bias, axes misalignment error, and scale factor error can be estimated deterministically through a six-position calibration or a local level calibration. The remaining IMU errors (bias drift and sensor noise) can only be dealt with stochastically and as such have to be modelled as separate error states.

The accelerometer and gyroscope measurement errors shown in Eq. 4.5 ($\delta\mathbf{f}$, $\delta\mathbf{w}$) are represented by the following expression (Petovello, 2003; Zhang, 2003):

$$\delta\mathbf{f} = \delta\mathbf{b}_a + \boldsymbol{\eta}_a \quad (4.6)$$

$$\delta\mathbf{w} = \delta\mathbf{b}_g + \boldsymbol{\eta}_g$$

The accelerometer and gyroscope noise, shown in Eq. 4.6 as $\boldsymbol{\eta}_a$, $\boldsymbol{\eta}_g$ respectively, is modelled as zero mean white Gaussian. In the literature, the INS bias drifts are modelled as first order Gauss-Markov processes. Gauss-Markov processes are often used in engineering as they describe many physical random processes with good approximations (Brown and Hawng, 1997). The differential equation representing bias drift takes the following form (El-Sheimy, 2012; Schwarz and Wei, 2000):

$$\dot{\delta\mathbf{b}}_a = -\boldsymbol{\beta}_a \delta\mathbf{b}_a + \boldsymbol{\eta}_{ba} \quad (4.7)$$

$$\dot{\delta\mathbf{b}}_g = -\boldsymbol{\beta}_g \delta\mathbf{b}_g + \boldsymbol{\eta}_{bg}$$

where accelerometer-specific values have a subscript a and gyroscope-specific values have a subscript b . To simplify the notation the subscript i will be used from now on to indicate the expression applies to both accelerometers and gyroscopes.

β_i is the inverse correlation time (1/sec) for the sensor

η_{bi} is the Gauss-Markov process driving noise with spectral density q_{bi}

β_i and η_{bi} (and therefore q_{bi}) should theoretically be computed from an autocorrelation sequence. A stationary (time-invariant process) is often used to model INS random errors such as the bias drift. Stationary processes are completely defined by the autocorrelation sequence which is defined by its time constant and variance (Nassar, 2003). There is however a limitation to this. Studies have shown that a large amount of data is needed in order for the autocorrelation sequence to be accurate enough. For example, if the desired uncertainty level is 10%, the required length of data will be approximately equal to 200 times the correlation time of the process (ibid). Thus, assuming a reasonable correlation time of 1 hour, 200 hours of data would be required in order for estimating the autocorrelation sequence with a 10% accuracy (ibid). The autocorrelation method from a static dataset can help with a starting point for β_i and η_{bi} . However, realistically, these initial values will have to be fine-tuned empirically. It is also worth stating the bias drift noise (and hence spectral density) is the more sensitive of the two to improper modelling.

The process to compute the bias drift and sensor noise values is as follows and has been applied in previous studies such as Petovello (2003), Godha (2006), and Gao (2007). Static raw IMU data is collected for as long as possible. For this thesis the data was collected for about 8 hours although as explained above, the longer the dataset, the more accurate the estimate of the Gauss-Markov parameters. The variance of raw accelerometer and gyroscope data is computed over evenly spaced one-second data

intervals. One-second intervals have been used in these previous studies to separate low frequency noise from the actual signal as much as possible. Once the mean of the variances is computed, the spectral noise density for the sensor noise can be found through the following expression as per Gao (2007):

$$q_{ni} = \frac{\sigma_{ni}^2}{2BW_i} \quad (4.8)$$

where again the subscript i is used to indicate the variable applies to both accelerometers (a) and gyrosopes (g).

σ_{ni}^2 is the high frequency noise variance for the sensor.

BW_i is the bandwidth of the sensor.

Once the high frequency noise is estimated through the above procedure, wavelet decomposition is performed on the raw signals. Choosing the appropriate level of decomposition will separate the high frequency noise from the low frequency part of the signal. This is found by comparing the standard deviation of the low frequency component (called the approximation) through different decomposition levels (Nassar, 2003). The chosen decomposition level is that which provides the smallest standard deviation (ibid). The autocorrelation sequence is then computed on the approximated signal and is represented by the expression below.

$$R_i(t) = \sigma_{bi}^2 e^{-\beta_i t} \quad (4.9)$$

where σ_{bi}^2 is the variance of the approximation and β_i is the inverse correlation time from Eq. 4.7. A least-squares analysis is then done in order to find the values σ_{bi}^2 and β_i .

Knowing these two values is important as it allows us to compute the bias drift spectral noise density, q_{bi} , (Gao, 2007):

$$q_{bi} = 2\sigma_{bi}^2 \beta_i \quad (4.10)$$

Computing q_{ni} , q_{bi} for accelerometers and gyroscopes gives us the spectral noise densities required to compute the process noise outlined in Eq. 3.15. Putting Eqs. 3.9, 4.5 – 4.10 together, gives the state-space model for this 15-state system. This system is made up of nine navigation error states and six sensor error states as shown below in Eq. 4.11:

$$\begin{pmatrix} \delta \dot{\mathbf{r}}^e \\ \delta \dot{\mathbf{v}}^e \\ \dot{\boldsymbol{\epsilon}}^e \\ \delta \dot{\mathbf{b}}_a \\ \delta \dot{\mathbf{b}}_g \end{pmatrix} = \underbrace{\begin{pmatrix} 0_{3x3} & \mathbf{I}_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ \mathbf{N}_{3x3}^e & -2(\Omega_{ie}^e)_{3x3} & -\mathbf{F}_{3x3}^e & (\mathbf{R}_b^e)_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & -(\Omega_{ie}^e)_{3x3} & 0_{3x3} & (\mathbf{R}_b^e)_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & -\boldsymbol{\beta}_a_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & -\boldsymbol{\beta}_g_{3x3} \end{pmatrix}}_F \begin{pmatrix} \delta \mathbf{r}^e \\ \delta \mathbf{v}^e \\ \boldsymbol{\epsilon}^e \\ \delta \mathbf{b}_a \\ \delta \mathbf{b}_g \end{pmatrix} + \underbrace{\begin{pmatrix} 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ (\mathbf{R}_b^e)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & (\mathbf{R}_b^e)_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & \mathbf{I}_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & \mathbf{I}_{3x3} \end{pmatrix}}_G \underbrace{\begin{pmatrix} \boldsymbol{\eta}_a \\ \boldsymbol{\eta}_g \\ \boldsymbol{\eta}_{ba} \\ \boldsymbol{\eta}_{bg} \end{pmatrix}}_w \quad (4.11)$$

and the corresponding spectral noise densities for w are:

$$\mathbf{Q}(t) = \begin{pmatrix} (q_a)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & (q_b)_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & (q_{ba})_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & (q_{bg})_{3x3} \end{pmatrix} \quad (4.12)$$

Note that the scale factor error and non-orthogonality errors are negligible for high-end IMUs but become more troublesome for lower-end sensors. Making this more difficult is the fact the scale factor error and turn-on biases change from turn on to turn on for lower-end sensors. As has been mentioned throughout, the lower the quality of the INS sensors, the harder it is to model its errors. The scale factor error could be modelled directly in the INS filter, or indirectly by relaxing (increasing) the uncertainty of the sensor noise (Bobye, 2014). Also, attempting to model all parameters may weaken the model to a point where the system becomes unstable (Gelb, 1974). If modelled in the INS, the scale factor error is considered to be a Gauss-Markov process. This would then mean there would be an inverse correlation time (β_{si}) and spectral noise density (η_{si}) associated with it, just like for the bias drift.

Likewise, there are a couple of ways to deal with the varying turn-on bias. The first option is to observe the biases for about half a minute while the vehicle is static (Kong, 2000). The turn-on biases for the accelerometers can be removed through the following expression:

$$\mathbf{b}_{a,turn-on} = \widehat{\mathbf{F}}_b - \mathbf{F}_b \quad (4.13)$$

$$\mathbf{b}_{a,turn-on} = \widehat{\mathbf{F}}_b - \mathbf{R}_l^b \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix}^l \quad (4.14)$$

where

$\widehat{\mathbf{F}}_b$ is the vector of the average specific forces over the defined static period.

\mathbf{F}_b is the expected specific forces by rotating the gravity vector from the 1-frame to the body frame as shown in Eq. 4.14. In this method, the turn-on biases for the gyroscope are found by computing the average raw gyroscope data throughout the static, half-minute initialization (ibid).

The alternate method is to consider the turn-on biases as white Gaussian noise in which case the differential equation would be represented by the expressions below (Bobye, 2014):

$$\dot{\mathbf{b}}_{a,turn-on} = 0 \quad (4.15)$$

$$\dot{\mathbf{b}}_{g,turn-on} = 0$$

Thus, if deciding to model scale-factor errors and turn-on biases as part of the INS sensor error states, the following would need to be appended to Eq. 4.11:

(4.16)

$$\underbrace{\begin{pmatrix} \dot{\mathbf{b}}_{a,turn-on} \\ \dot{\mathbf{b}}_{g,turn-on} \\ \dot{\mathbf{S}}_a \\ \dot{\mathbf{S}}_b \end{pmatrix}}_{\dot{x}} = \underbrace{\begin{pmatrix} 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & -\boldsymbol{\beta}_{sa}{}_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & -\boldsymbol{\beta}_{sg}{}_{3x3} \end{pmatrix}}_F \underbrace{\begin{pmatrix} \mathbf{b}_a \\ \mathbf{b}_g \\ \mathbf{S}_a \\ \mathbf{S}_b \end{pmatrix}}_x + \underbrace{\begin{pmatrix} 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} \\ \mathbf{I}_{3x3} & 0_{3x3} \\ 0_{3x3} & \mathbf{I}_{3x3} \end{pmatrix}}_G \underbrace{\begin{pmatrix} \boldsymbol{\eta}_{sa} \\ \boldsymbol{\eta}_{sg} \\ w \end{pmatrix}}_w$$

with a spectral noise density of

$$\mathbf{Q}(t) = \begin{pmatrix} (q_{sa})_{3x3} & 0_{3x3} \\ 0_{3x3} & (q_{sg})_{3x3} \end{pmatrix} \quad (4.17)$$

resulting in the 27-state filter shown in Appendix A.

4.3.2 Measurement Model

The measurement model for the loosely-coupled integration is the same regardless of whether the filter is 15-state or 27-state. Recall from Chapter 3 that in an EKF the error states are being estimated and that as such, the INS measurement vector is composed of the difference between the INS estimates and the observations. The observations in the loosely-coupled integration are positions and velocities. Thus the measurement vector becomes:

$$\delta \mathbf{z} = \begin{pmatrix} r_{INS}^e - r_{GNSS}^e \\ v_{INS}^e - v_{GNSS}^e \end{pmatrix} \quad (4.18)$$

The expression above is used in the innovation sequence previously shown in Equation 3.20. Let us show it again for convenience:

$$V_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_k(-)$$

The difference between Equation 4.18 and Equation 3.20 is that the former uses δ to indicate it is the error state being modelled rather than the state itself. The innovation sequence is comparing the actual incoming observation updates with the INS predictions. The innovation sequence was then shown to be weighted via the gain matrix prior to being used to compute the updated states. Let us show Equation 3.21 again for convenience:

$$\hat{\mathbf{x}}_k(+) = \hat{\mathbf{x}}_k(-) + \mathbf{K}_k V_k$$

It is also worth recalling the gain matrix, \mathbf{K}_k , uses knowledge of the accuracy of the incoming GNSS updates as well as the predicted INS states in order to determine to

what degree should the updates be trusted. The accuracy of the incoming GNSS updates were represented by \mathbf{R}_k in Equation 3.19 shown below for convenience:

$$\mathbf{K}_k = \mathbf{P}_k(-) \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k(-) \mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

The measurement noise variance is then:

$$\mathbf{R} = \begin{pmatrix} \sigma^2 r_{GNSS}^e & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & \sigma^2 v_{GNSS}^e \end{pmatrix} \quad (4.19)$$

Note even though the measurement used is the difference between INS estimate and GNSS observations, the measurement noise is composed of the variance of the observations. The reason for this is that the INS accuracies are already being taken into account through the current state covariance, $\mathbf{P}_k(-)$, in the gain matrix computation.

It is entirely possible to have epochs where only positions or velocities are observed (e.g. zero velocity updates), but having both will provide the best filter updates. The design matrix in the loosely-coupled integration is then:

$$\mathbf{H} = \begin{pmatrix} \mathbf{I}_{3x3} & \mathbf{0}_{3x3} & \mathbf{0}_{3x(n-6)} \\ \mathbf{0}_{3x3} & \mathbf{I}_{3x3} & \mathbf{0}_{3x(n-6)} \end{pmatrix} \quad (4.20)$$

4.4 External Aided Heading

As was stated on Chapter 3, the best way to align (initialize) the INS is through a dual GNSS antenna due to the fact it can be performed on any IMU grade regardless of whether the vehicle is kinematic or static. The standard aided heading is computed from

an RTK solution between two GNSS receivers. One of the receivers, called the master, sends raw observations to the other receiver (called the rover), which then uses double differencing to compute a very accurate baseline between both receivers (NovAtel Inc, 2015a). From the very accurate baseline, it is possible to find the heading and pitch/roll depending on whether the baseline is oriented across track or along track with respect to the direction of travel. The only difference with standard RTK is that in the latter the base sending corrections to the rover is fixed. The very accurate baseline is computed, and the offset is added to the fixed position of the base, thus providing very accurate absolute positioning on the rover.

The external aided heading from GNSS was not used in this project as it is usually used in the higher-end receivers that provide the capability of outputting raw measurements (something that cannot be done on the version of Android used herein). However, the Android smartphone does have magnetometers that could have been used as a heading aiding source. Whether the aiding source comes from GNSS, magnetometer, or another sensor, the aided heading plays a very important role not only for INS alignment purposes. As was explained in Chapter 2, heading is the most difficult attitude component to observe due to the fact it is not measured through any force (like pitch and roll, which can be estimated if gravity is known). Instead, heading accuracy depends on the quality of the gyroscopes inside the IMU (mainly turn-on biases and bias drifts) as well as the kinematics in the system. For example, suppose a high-end IMU (single antenna scenario) is used in an application where the vehicle travels in a straight path for a long period of time. The EKF running the INS uses previous knowledge of its dynamics as well as current observations to determine the position, velocity, and attitude

and to control other errors being modelled. Gravity is observed indirectly through a gravity model, which requires position, meaning the pitch and roll are indirectly taken care off. The same cannot be said about heading. Thus in a scenario such as this one, the filter's heading will start drifting as it will lose knowledge of changes in heading over time. This is the secondary case in which a dual GNSS antenna or magnetometer or any secondary aiding source will help. By providing constant heading updates, the heading drift is constrained (slowed down).

If using an external heading sensor or any other type of sensor for that matter, all that is required is to model the design matrix describing the relation of the measurements to the error states as well as use an appropriate measurement variance. In the external heading sensor case, the corresponding row of the design matrix is:

$$\mathbf{H}(\psi) = \left(0_{1 \times 3} \ 0_{1 \times 3} \ \frac{\partial \hat{\psi}}{\partial \varepsilon_x} \ \frac{\partial \hat{\psi}}{\partial \varepsilon_y} \ \frac{\partial \hat{\psi}}{\partial \varepsilon_z} \ 0_{1 \times (n-9)} \right) \quad (4.21)$$

where, $\frac{\partial \hat{\psi}}{\partial \varepsilon_x}$ $\frac{\partial \hat{\psi}}{\partial \varepsilon_y}$ $\frac{\partial \hat{\psi}}{\partial \varepsilon_z}$ are the derivatives of the heading function with respect to the attitude errors in the X, Y, and Z of the e-frame. Readers are invited to take a look at Godha (2006) for further details. The corresponding row of the measurement vector is:

$$\delta\psi = (\psi_{INS} - \psi_{external}) \quad (4.22)$$

Note that because these equations represent the geometry of the heading measurement model with respect to the unknowns, the expressions are the same regardless of whether the heading estimate comes from a secondary GNSS antenna, a magnetometer, or some other sensor. Magnetometer calibration is shown in Appendix B.

4.5 Motion Constraints

As has been mentioned over the last two chapters, an INS solution will drift if there is a GNSS (measurement) outage due to the system having to rely on its predicted model that, without guidance, will drift. The drift of the system is dependent on the quality of the INS sensors and has been shown before, the INS sensors inside the Android smartphone are orders of magnitude worse than entry-level sensors used in the industry (MEMS), let alone high-end ones. Given the fact it is entirely possible for the vehicle to encounter a GNSS outage in an urban canyon scenario, something has to be done. The idea in this section is to use the knowledge of the application where the INS is to be used in order to constrain these drifts. Specifically, the INS tests in the research reported in this thesis were carried out in land applications. This doesn't mean it can't be used in other (e.g. aerial) applications but it means the filter is optimized for land kinematics. The first constraint is to consider the fact the vehicle will only move along-track (Y-axis of vehicle frame), thus the vehicle should only accelerate as such (Sukkarieh, 2000). The second constraint is the vehicle will not experience rapid changes in height over consecutive epochs. Recall the IMU operates at a high rate (~ 25 Hz for the sensors used herein), thus a sudden height change from epoch to epoch is unexpected. Due to the fact the constraints are to control error state growth during outages, they are only applied when no measurements are present.

4.6 Velocity Constraints

The vehicle constraints mentioned above are derived from the fact the vehicle will not slip, and that it won't accelerate upwards (Godha, 2006). If both assumptions are correct, then the across-track and vertical velocities of the vehicle must be zero (Sukkarieh, 2000). For simplicity's sake, let us assume the IMU enclosure frame matches that of the vehicle frame thus accelerations along the X and Z axes of the b-frame should be zero. Mathematically, we will require considering measurement noise due to imperfections in the above assumptions. The expected imperfections determine the magnitude of the noise (ibid). The following block diagram shown in Figure 4-3, explains where velocity constraints would be used in the filter:

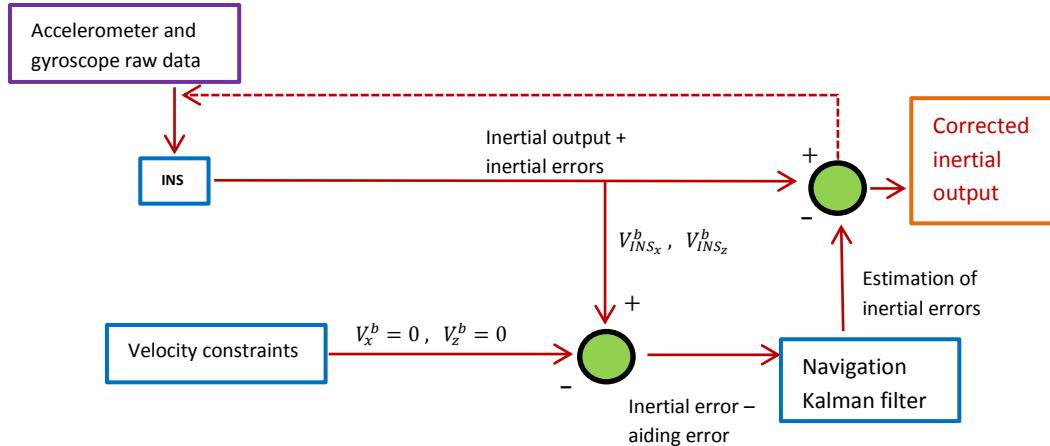


Figure 4-3: Velocity constraints (adapted from El-Sheimy, 2012)

A complete derivation of the velocity constraints can be found in Godha (2006). Only the final measurement model will be shown here. Recall the INS solution was

mechanized in the e-frame. Therefore, the INS velocity estimates need to be transformed from the e-frame to the b-frame as shown below:

$$\boldsymbol{v}^b = \boldsymbol{R}_e^b \boldsymbol{v}^e \quad (4.23)$$

The design matrix relating the velocity updates from the e-frame to the b-frame is:

$$(4.24)$$

$$\boldsymbol{H}_{velConstraint} =$$

$$\begin{pmatrix} 0_{1 \times 3} & R_{11} & R_{12} & R_{13} & R_{12}v_z^e - R_{13}v_y^e & R_{13}v_x^e - R_{11}v_z^e & R_{11}v_y^e - R_{12}v_x^e & 0_{1 \times (n-9)} \\ 0_{1 \times 3} & R_{31} & R_{32} & R_{33} & R_{32}v_z^e - R_{33}v_y^e & R_{33}v_x^e - R_{31}v_z^e & R_{31}v_y^e - R_{32}v_x^e & 0_{1 \times (n-9)} \end{pmatrix}$$

The measurement updates used are:

$$\delta \mathbf{Z}_{velConstraint} = \begin{pmatrix} \delta v_x^b \\ \delta v_x^b \end{pmatrix} = \begin{pmatrix} v_{INS_x}^b - 0 \\ v_{INS_z}^b - 0 \end{pmatrix} \quad (4.25)$$

In previous research carried out by Shin (2001), estimated velocity constraint measurement noise was estimated by projecting the forward velocity into the x and z axes using an assumption of the expected misalignment angle between the IMU body frame and the vehicle frame. For the research carried out in this thesis better results were obtained using the INS velocity accuracies in the x and z axes for the last known INS epoch that contained an update.

4.7 Height Constraints

The second type of constraints mentioned earlier has to do with knowing that the epoch-to-epoch change in elevation is not going to vary much. A constant value of the height in the area is used as a measurement update, with the value being used from a priori information or from the last known computed INS update. Applying a height constraint will prevent the vehicle from appearing to move vertically when the INS is in prediction mode during a GNSS outage while also improving the horizontal accuracies (Schwarz and Wei, 2000). However, as with any type of INS update, care must be taken in choosing the measurement variance of the height estimate; using wrong values can skew the horizontal position (MacGougan, 2003). The block diagram for using height constraints in the EKF is shown below in Figure 4-4:

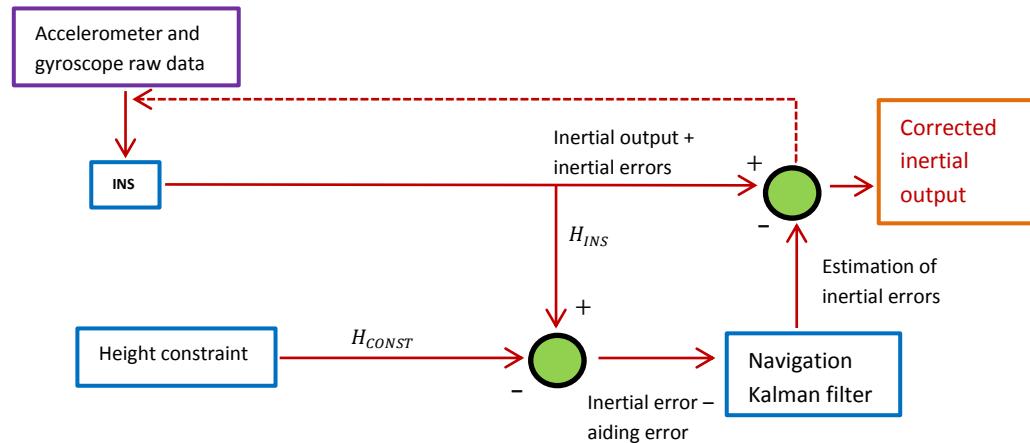


Figure 4-4: Height constraints (adapted from El-Sheimy, 2012)

As with the velocity constraints, the measurement model isn't straight forward due to the fact the mechanization takes place in the e-frame but the l-frame is used for output.

First, let us look at the coordinate transformation from geodetic coordinates to ECEF Cartesian coordinates:

$$r_x = (N + h) \cos \lambda \cos \phi$$

$$r_y = (N + h) \cos \lambda \sin \phi \quad (4.26)$$

$$r_z = (N - e^2 N + h) \sin \lambda$$

where,

r_x, r_y, r_z are Cartesian coordinates

ϕ, λ, h are the geodetic latitude, longitude, and height respectively

N is the ellipsoidal prime vertical radius of curvature

e is the ellipsoidal eccentricity

The expression representing the position perturbations in the e-frame with respect to the perturbations in the l-frame is as follows. Complete derivation can be found in Farrell and Barth (2001).

(4.27)

$$\begin{pmatrix} \delta r_x \\ \delta r_y \\ \delta r_z \end{pmatrix} = \underbrace{\begin{pmatrix} A \cos \phi \cos \lambda - (N + h) \sin \phi \cos \lambda & (-N - h) \cos \phi \sin \lambda & \cos \phi \cos \lambda \\ A \cos \phi \sin \lambda - (N + h) \sin \phi \sin \lambda & (N + h) \cos \phi \cos \lambda & \cos \phi \sin \lambda \\ A(1 - e^2) \sin \phi + N(1 - e^2) \cos \phi + h \cos \phi & 0 & \sin \phi \end{pmatrix}}_{\hat{H}} \begin{pmatrix} \delta \phi \\ \delta \lambda \\ \delta h \end{pmatrix}$$

where

$$A = \frac{-ae^2 \sin \phi \cos \phi}{\sqrt{(1-e^2 \sin^2 \phi)^3}} \quad (4.28)$$

The design matrix for the height constraint update is computed from the last row of \hat{H} :

$$\mathbf{H}_{\text{HeightConstraint}} = (\hat{H}_{3,1}^{-1} \quad \hat{H}_{3,2}^{-1} \quad \hat{H}_{3,3}^{-1} \quad \mathbf{0}_{1 \times (n-3)}) \quad (4.29)$$

The measurement updates used are:

$$\delta \mathbf{Z}_{\text{HeightConstraint}} = \delta H = (H_{\text{INS}} - H_{\text{Const}}) \quad (4.30)$$

where H_{Const} is a constant height value for the duration of the outage. In the research carried out in this thesis, H_{Const} was the height in the last INS epoch that contained an update. As per Godha (2006), the measurement noise for H_{Const} depends on the quality of the height being used as constant. Since INS heights from the last known epoch were used as H_{Const} , the INS derived height for the respective epoch was used as the measurement noise.

Putting together Figure 4-3 and Figure 4-4, the block diagram representing both velocity and height constraints is shown below in Figure 4-5.

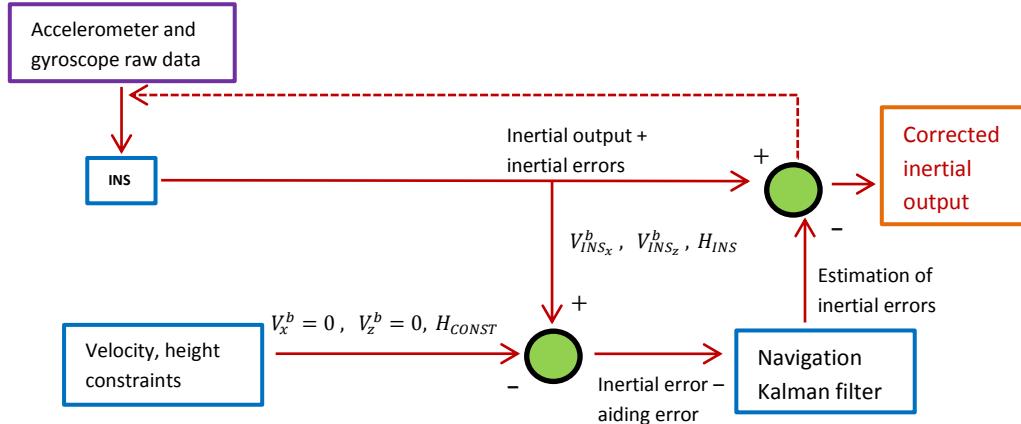


Figure 4-5: Motion constraints (adapted from El-Sheimy, 2012)

If the motion constraints were to be also used during the update stage of the filter, the \mathbf{H} , \mathbf{Z} , \mathbf{R} matrices would be made up from appending the above equations to the standard measurement model equations used during loosely coupling shown earlier in the chapter (Gao, 2007). Due to the increasing processing times of doing so, the motion constraints used in this thesis were only applied during the prediction stage at a rate of 1 Hz in the absence of GNSS updates.

CHAPTER 5: FILTER VERIFICATION THROUGH TRUTH TESTING

As has been mentioned in previous chapters, the purpose of the research described in this thesis was to develop a loosely coupled GNSS/INS integration using sensors inside the Android Moto-X smartphone and to assess its performance in land vehicle kinematics. The GNSS/INS filter was written from scratch in C++ with the theoretical background coming from the sources referenced throughout this thesis. Because of the fact it was known ahead of time it would be hard to get good results with the smartphone sensors, the INS code had to be first tested against higher-end, easier-to-model sensors. This would allow verifying the backbone of the filter was operating as expected. Otherwise, it would have been practically impossible to distinguish code issues from filter tuning due to high sensor noise and sensor variability. Once the filter was verified, the smartphone sensors' performance was assessed through comparisons with a truth trajectory obtained at the same time with the vehicle.

5.1 IMU-LN200

The first part of the task consisted in comparing the tightly-coupled real-time single point NovAtel SPAN solution against the INS solution using the created filter. Single point positioning was used due to the fact the smartphone GNSS receiver would not be receiving any external corrections either. This in turn means the filter's performance is assessed to perform to the expected level of the smartphone. As mentioned in the

introduction, high-end sensors were required for this initial comparison which is why the IMU-LN200 was chosen. The specifications for this IMU are shown below (NovAtel, 2015g):

Table 5-1: IMU-LN200 characteristics

Gyro input range	$\pm 1000 \text{ }^{\circ}/\text{s}$
Gyro rate bias	$1 \text{ }^{\circ}/\text{hr}$
Gyro rate scale factor	100 ppm
Angular random walk	$0.07 \text{ }^{\circ}/\sqrt{\text{hr}}$
Accelerometer range	$\pm 40 \text{ g}$
Accelerometer scale factor	300 ppm
Accelerometer bias	0.3 mg
Input voltage range	+12 to +28 V DC
Power Consumption	16 W
Dimensions	13.5 cm x 15.3 cm x 13.0 cm
Weight	2.99 kg

The IMU-LN200 is a high performance IMU containing three accelerometers and gyroscopes orthogonally mounted to one another. The IMU enclosure contains the LN200 as well as an interface card whose main role is regulate IMU power and control

the highly precise IMU timing. The relatively small size and weight for a high performance IMU makes this dual-connector enclosure variant easy to install in tight-to-fit spaces. Important values to make note of are the gyroscope input range, gyroscope bias, angular random walk and accelerometer bias. The high gyroscope input range of 1000 deg/s and overall quality make this IMU a controlled good meaning it is export regulated by International Traffic in Arms Regulations (ITAR), among other regulatory bodies. The IMU-LN200's gyroscope input range in particular matches high rotation rates achieved by autonomous military projectiles. The low gyroscope turn-on bias of 1°/hr makes this IMU highly sensitive to the Earth's rotation rate thus not requiring any external input in order to align the INS. For the purpose of this project, this IMU offers a highly accurate truth trajectory when used in NovAtel SPAN firmware. The following are achieved when the IMU-LN200 is integrated with the tightly-coupled INS solution from NovAtel SPAN under ideal conditions, where an ideal condition corresponds to a properly set up system experiencing high kinematics:

Table 5-2: NovAtel SPAN + IMU-LN200 (NovAtel Inc, 2015e)

Outage Duration	Position Mode	Position Accuracy RMS (m)		Velocity Accuracy RMS (m/s)		Attitude Accuracy RMS (degrees)		
		Horizontal	Vertical	Horizontal	Vertical	Pitch	Roll	Heading
0 s	Single	1.200	0.600	0.020	0.010	0.011	0.011	0.022
	RTK	0.020	0.050	0.020	0.010	0.011	0.011	0.020
10 s	Single	1.340	0.670	0.030	0.012	0.012	0.012	0.029
	RTK	0.120	0.070	0.025	0.011	0.011	0.011	0.022

60 s	Single	3.510	0.960	0.105	0.019	0.015	0.015	0.039
	RTK	2.790	0.630	0.102	0.023	0.013	0.013	0.031

Table 5-2 above shows the RMS accuracy of the SPAN+IMU-LN200 tightly-coupled solution for both single point and RTK under different outages. Note that attitude and velocity are quite accurate after a 60-second outage, even when in single point positioning. It is also worth mentioning that due to the fact the INS filter developed was loosely coupled, the filter's performance was assessed with the SPAN single point solution rather than RTK. That is, the filter inputs are single point position and velocities. The IMU-LN200 used is shown below in Figure 5-1.



Figure 5-1: IMU-LN200 strapped to vehicle

5.2 Other Equipment Used

The other GNSS-related equipment required to perform the truth testing consisted of a NovAtel Flexpak6 loaded with GPS+GLONASS dual frequency SPAN firmware and a

NovAtel GPS-702-GG pinwheel antenna. Note that only a single antenna was used throughout in order to assess the INS filter performance without heading aiding. This allows for a more direct attitude convergence comparison of the NovAtel SPAN tightly-coupled solution with the developed loosely-coupled solution.

The high grade antenna was attempted to be placed directly over the IMU's centre of navigation to facilitate lever arm measurements. This was done eyeballing it with the help of a couple people. The actual lever arms themselves were measured as follows. First, a static point (let's refer to it as the reference point, RP) was placed outside on the right hand side of the vehicle. The offsets of both antenna phase centre and IMU centre of navigation were then measured with respect to this external RP using a tape measure. From these two offsets, the IMU centre of navigation to antenna phase offset was computed. The arithmetic mean of three different sets of measurements was then used. An uncertainty of 7 cm was chosen to take into account the fact the measuring tape wasn't completely level, and the fact human error could play a part in the measurements. This procedure was chosen because the external reference point was easier to measure to rather than the IMU centre of navigation to antenna phase centre offset directly. The 7 cm uncertainty is just fine for these tests given the fact the GNSS updates used are single point positioning, with metre-level standard deviation (Bahan, 2014).

The other equipment used consisted of a laptop to store GNSS computed positions and velocities (to be used as INS updates in developed filter) as well as the real-time NovAtel SPAN solution for truth trajectory comparison. Two 12 volt car batteries were used to power the Flexpak6 and IMU-LN200 separately. The setup is shown below in Figure 5-2.



Figure 5-2: Equipment used for truth testing

5.3 Procedure for Acquisition used for Truth Testing

Truth testing was performed on three datasets with different conditions in the city of Calgary, AB. The first dataset was an open-sky static test that was used to test filter convergence with zero velocity updates (ZUPTs). The remaining datasets were kinematic. The first kinematic dataset was obtained under open skies in NW Calgary on April 18th, 2015. The second kinematic dataset was obtained in an urban canyon scenario in downtown Calgary on April 20th, 2015. Both kinematic scenarios started with a static INS alignment which is sometimes referred to as coarse alignment in the literature. This was followed by making figure 8s with the vehicle in order to introduce kinematics and provide the filter with changes in heading (sometimes referred to as fine alignment). This was done until the SPAN INS status changed from

INS_ALIGNMENT_COMPLETE to INS SOLUTION_GOOD, which would be at the point the INS heading standard deviation is less than 1° . GNSS pseudorange positions, Doppler-derived velocities, and IMU raw data were stored from the time the receiver was powered on. These would later be run into the developed INS filter. The dataset locations are shown below in Figure 5-3 and the tests of the results are discussed in the upcoming sections.



Figure 5-3: Overview of truth tests in Calgary (Google, 2015)

5.4 Static Test

The static test was the first one performed in order to assess filter convergence under position updates and ZUPTs. It was carried out in the Calgary Transit parking lot in Panorama Hills in NW Calgary, on a weekend morning when it is for the most part

empty. A relatively empty parking lot was desired to minimize the effect of multipath on the position estimates.

The NovAtel SPAN solution with the IMU-LN200 was once again used as truth. This static test was very important because it was a good starting point for verifying the overall effectiveness of the mechanization equations and filter updates. The static test environment is shown below in Figure 5-4.

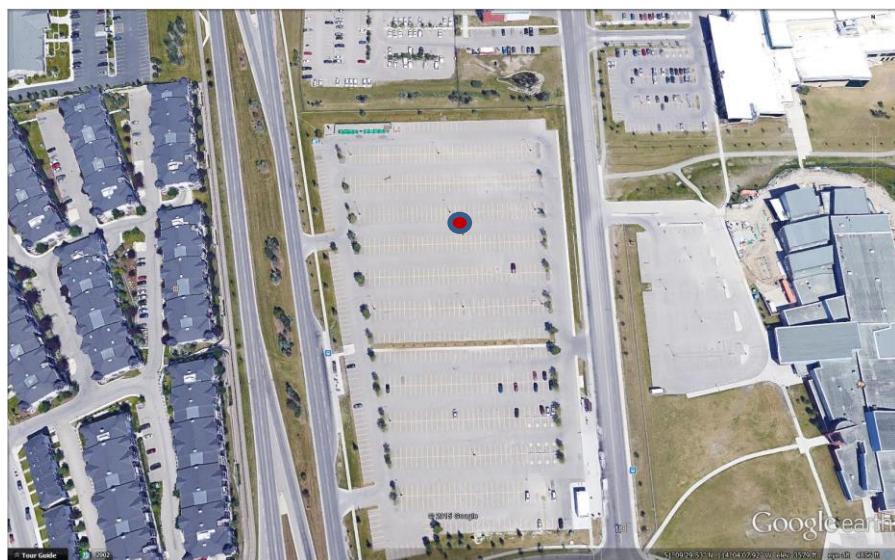


Figure 5-4: Static test environment (Google, 2015)

The main assessment of the static test is in attitude as the position and ZUPT updates should maintain the position and velocity in check. The following is the change in position for the filter with respect to the first position estimate.

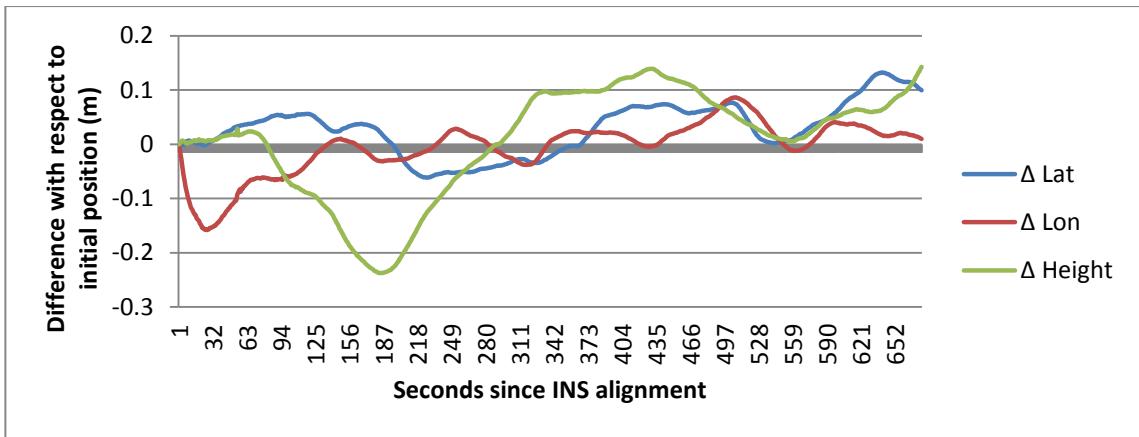


Figure 5-5: Differences in position for static test

As can be seen in Figure 5-5, the position estimates of the filter are well within the expected 1.2 metres uncertainty (NovAtel Inc, 2015g) for single point positioning provided by the NovAtel firmware. The position updates were weighted as per their respective standard deviations reported in the firmware, which were as shown below in Figure 5-6.

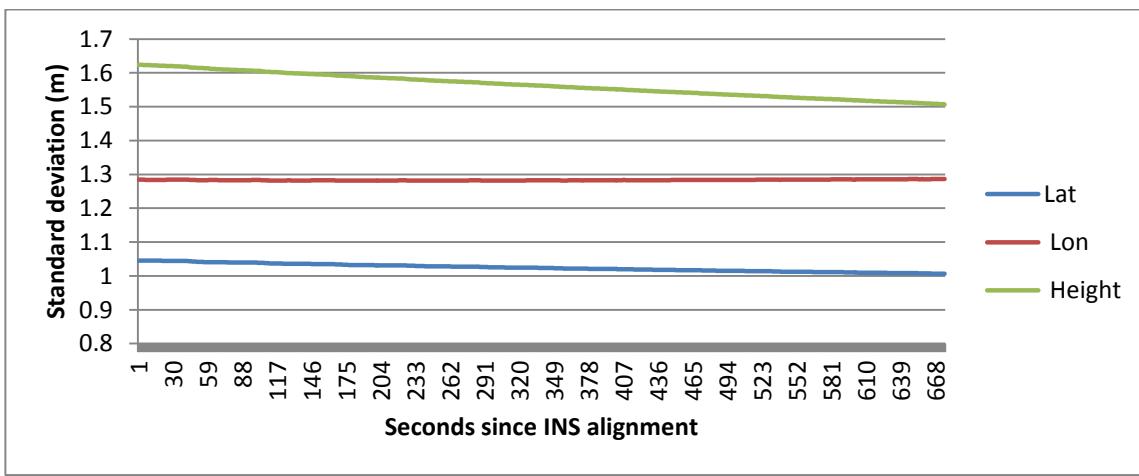


Figure 5-6: Standard deviations of single point position

The reason why the position estimates are much better in the INS compared to the input updates is that the final position estimate is not only dependent on the pseudorange position updates, but rather also on the ZUPTs that were highly weighted in the filter. This shows that at the single point update level, the INS is going to compute position estimates of greater quality than those that are being fed as updates. If the vehicle was moving or if the ZUPT knowledge wasn't as high as it is here, the INS position estimates are expected to be less accurate than shown here, but still better than the single point estimates being originally fed. If the position updates fed into the filter were of RTK quality, then the expectation would be for the difference in position with respect to the initial epoch to be at the cm-level.

The best assessment on this test is to see what the attitude in the created filter is like. It is important to recall that an INS is in essence a kinematics filter and that in the absence of heading aiding sources, the INS heading estimate will drift at a rate inversely proportional to that of the quality of the gyroscopes inside the IMU. The following two figures show the attitude computed by the INS filter during the static test.

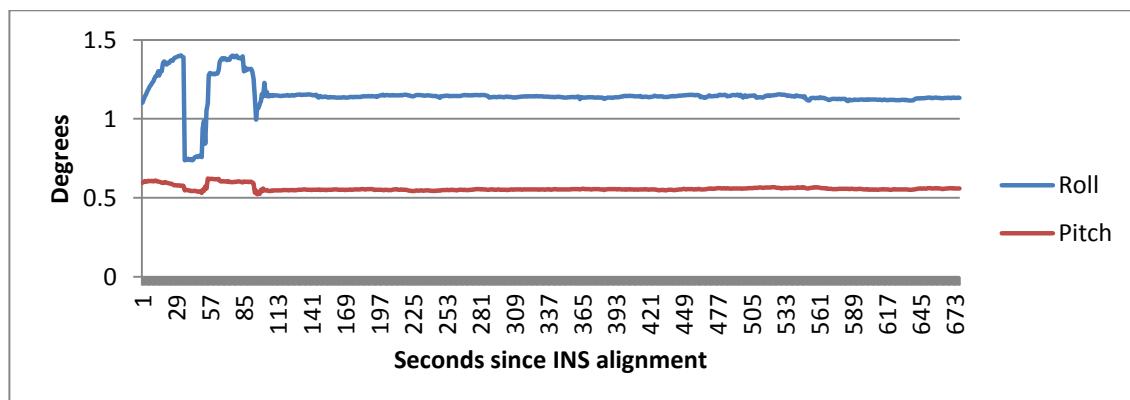


Figure 5-7: Pitch and roll during static test

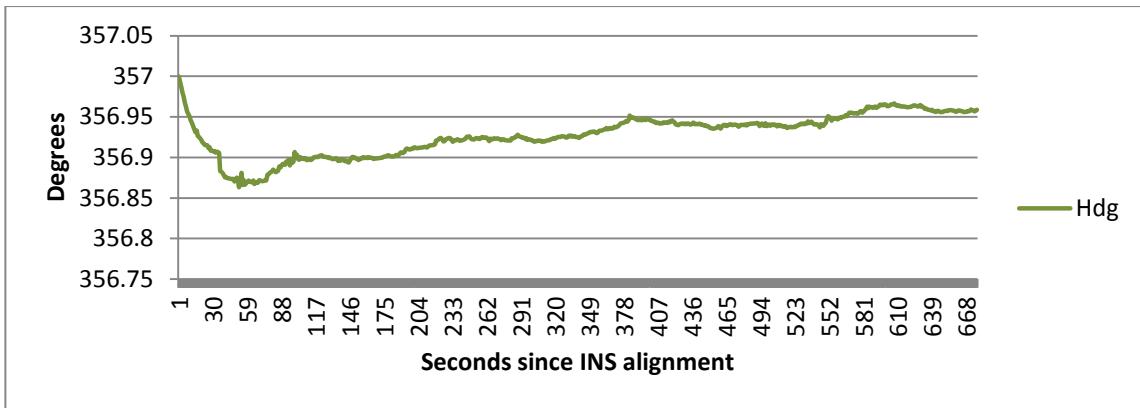


Figure 5-8: Heading during static test

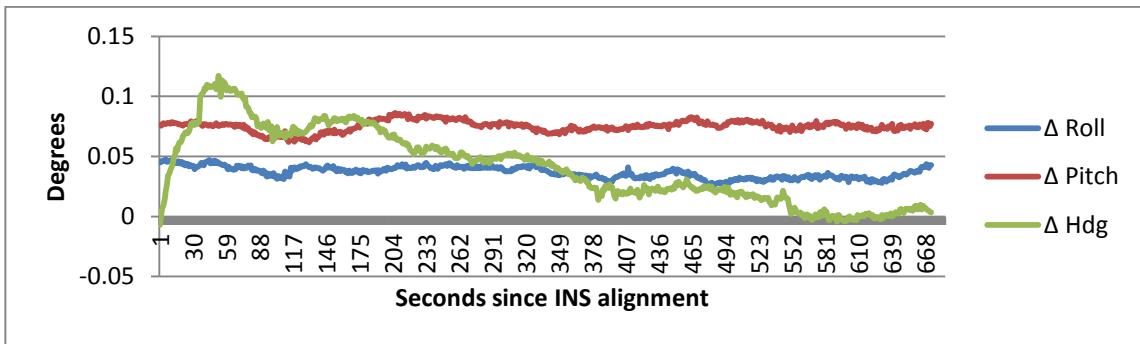


Figure 5-9: Difference in attitude with respect to truth in static test

Figure 5-9 above shows there is a bias in the attitude differences between the SPAN filter and the author-generated INS. The biases come from differences in spectral noise densities between both filters. The bias found in this test is added to the turn-on-biases already applied to the IMU-LN200 raw data in the kinematic datasets for higher accuracy. After taking into account the biases, the differences in pitch/roll are about 0.01 degrees, which is within the expected accuracies as per the NovAtel SPAN solution (about 0.01 degrees). The heading difference (sans bias) is about twice as much as the NovAtel SPAN specifications which is attributed to better heading constraints in the

SPAN firmware during ZUPTs. An important point to make is that the NovAtel solution has the benefit of years of experience tuning the process noise values which is a never-ending process (Dixon, 2015). The results being produced by the author-developed INS are therefore considered to be within expectations. The final computed differences are shown below in Table 5-3.

Table 5-3: Differences in attitude between author-generated INS and SPAN

	Pitch (deg)	Roll (deg)	Heading (deg)
Mean	0.0752	0.0368	0.0415
Standard Deviation	0.0046	0.0048	0.0310
RMS	0.0753	0.0371	0.0518

5.5 Open-sky Kinematic Truth Test

The open-sky kinematic truth test started at the same parking lot where the static test was done and consisted of a round trip to Cross Iron Mills, a shopping complex, just outside of NW Calgary. The test was pretty much in open-sky conditions for the entire duration except for an overpass that was passed underneath on the way back from Cross Iron Mills. Going below this overpass required an epoch of INS prediction as it wasn't possible to get a single point pseudorange GNSS position from the Flexpak6. There were a few red lights encountered on the way which were good to confirm the ZUPT

capabilities seen in the static test. The traverse also includes a few spots of large heading change, which are perfect for testing the attitude robustness of the author-created INS compared to the tightly-coupled SPAN solution. The results were quite good and confirmed what was said in Chapter 3. That under open-sky, there isn't much benefit accuracy-wise for a tightly-coupled over a loosely-coupled integration. The complete open-sky position results are shown below in Figure 5-10.

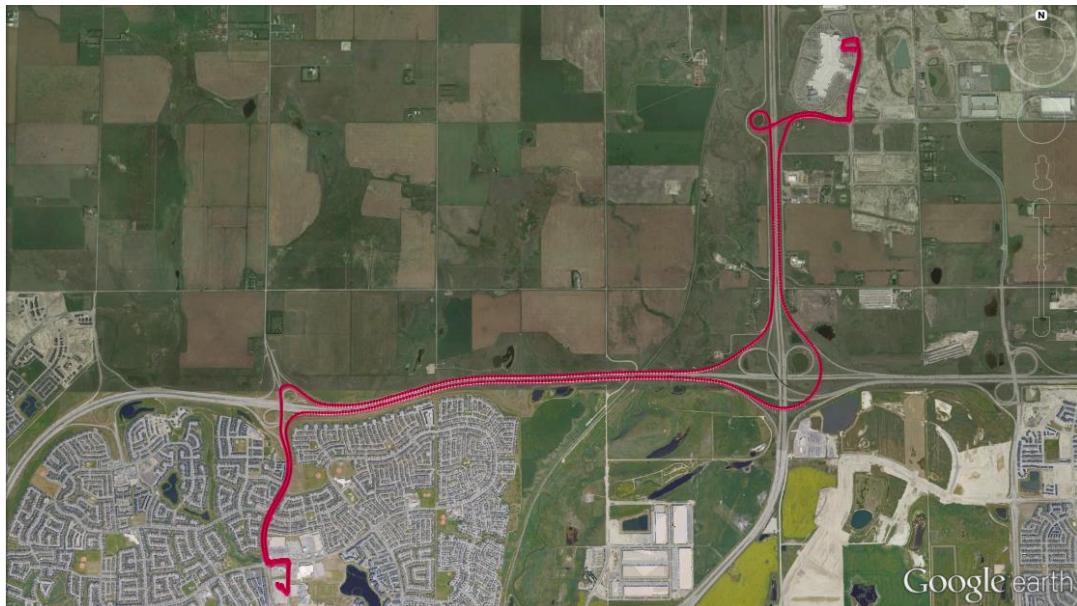


Figure 5-10: Open-sky kinematic truth test (Google, 2015)

The traverse can be divided into three different sections. The first section is mainly north-south at the start consisting of a number of red lights with maximum speeds of around 60 km/hr. It is followed with a fast-moving east-west part once the highway (Stoney Trail) is reached. Speeds in this part were as high as 110 km/hr. The second section is a long free-flowing road interchange for merging in/out of Deerfoot Trail (AB Hwy 2). The third section goes through a couple of red lights and is slower moving

when approaching Cross Iron Mills. The left-hand side of this section shows the merge back to Deerfoot Trail, where the underpass mentioned earlier is located.



Figure 5-11: Sections in open-sky truth testing (Google, 2015)

The three different sections mentioned above in Figure 5-11 have been shown on all the plots below to better identify whether there is a direct correlation to solution quality with the kinematics encountered. The first and last sections correspond to the first leg, and the middle section corresponds to the Cross Iron Mills sections and includes the underpass where the INS predicts without receiving an update.

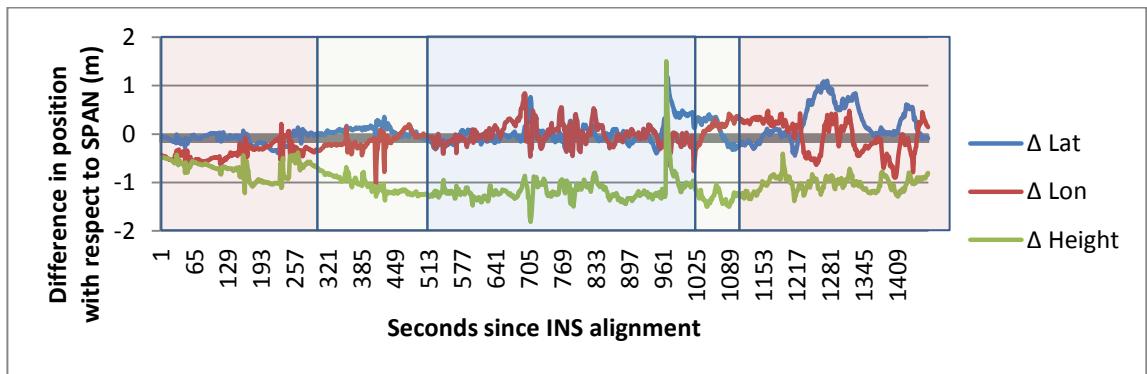


Figure 5-12: Position difference between SPAN and author-generated INS (open sky)

Figure 5-12 above shows that the position accuracy of the author-generated INS is within the NovAtel single point specifications of 1.2 m horizontal and 1.8 m vertical accuracies. The SPAN INS specifications shown in Table 5.2 represent better accuracies than seen in this dataset, which can be attributed to the difference in tightly-coupled vs. loosely-coupled implementations as well as how the updates fed into the INS are being weighted. Recall the position and velocity updates in the author-generated INS are weighted as per the accuracies reported in the raw PSRXYZB message from the NovAtel firmware. The spike seen in the mid-section of Figure 5-12 corresponds to the underpass where the GNSS outage took place. Its appearance indicates differences in how the vehicle motion constraints were handled in SPAN compared to the loosely-coupled INS.

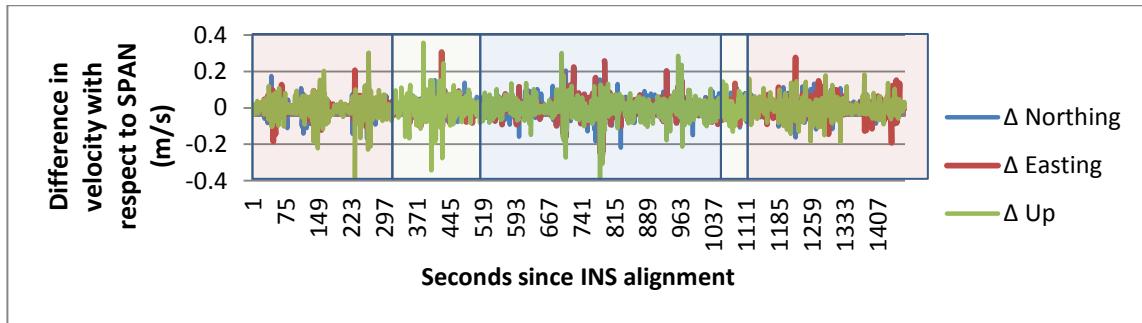


Figure 5-13: Velocity difference between SPAN and author-generated INS (open sky)

Figure 5-13 shows the velocity difference between the truth and the loosely-coupled filter. The larger differences correspond to the up component which is expected from the fact satellite geometry is better horizontally than vertically. The differences are within the expectation of the filter, which is for smart-phone sensor use.

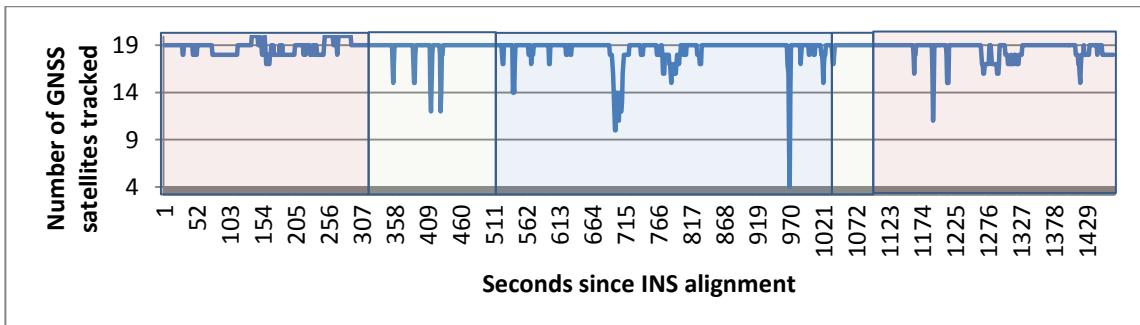
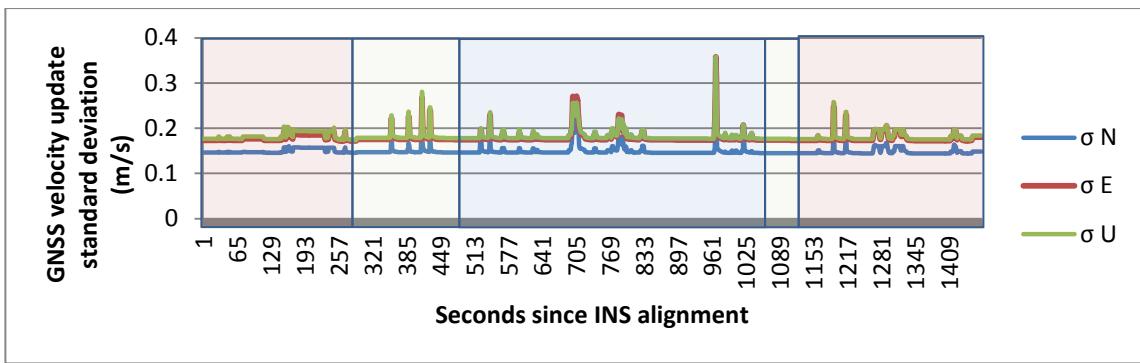
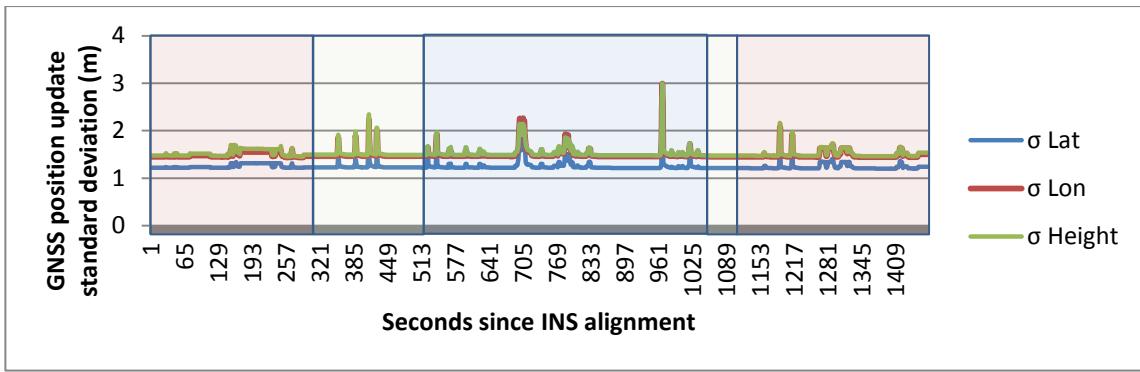


Figure 5-14 through Figure 5-16 show the correlation between the position and velocity updates fed into the INS and number of satellites being tracked. Even though the only GNSS outage occurred at the underpass on the way back from Cross Iron Mills, there are a few times in the dataset where the number of satellites tracked dips and these dips correspond to a decrease in position and velocity precision. As was explained in Chapter 2, GNSS accuracy is directly proportional to satellite geometry, where a better geometry is achieved with higher number of satellites used. Note that towards the end of the mid-section of the dataset there is a large dip in the number of satellites tracked. This corresponds to when the car was right beside the Cross Iron Mills building while driving in the parking lot. The vehicle speed was quite low, part of the sky was covered, and there was significant multipath from the large number of vehicles around. One last thing worth mentioning is that ZUPTs were applied if horizontal velocities fed into the updates were smaller than 5 cm/sec and were weighted higher than non-zero velocity updates.

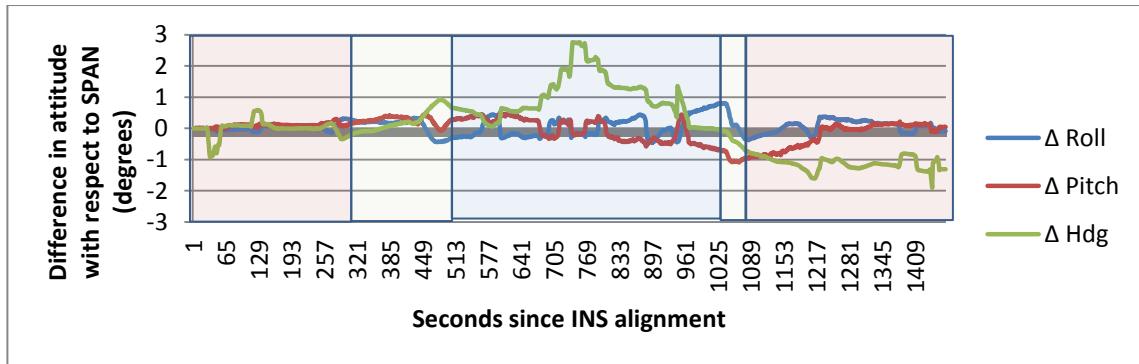


Figure 5-17: Attitude difference between SPAN and author-generated INS (open sky)

This last plot, Figure 5-17, the attitude difference between the loosely-coupled filter and the truth. The pitch and roll differences are quite small throughout the dataset

meaning the normal gravity model was being properly handled in the mechanization equations and corrected accordingly in the INS filter. The heading, as mentioned in Chapter 3, is the least accurate attitude component due to it not being directly measured (as pitch/roll are through gravity) but instead depend on vehicle kinematics and validity of sensor modelling. The sections in the plot where heading differences are increased to just below 3° are those in which velocities are lower, the number of satellites tracked dips, and there is a higher multipath from surrounding environment. Higher multipath impacts the accuracy of the position and velocity updates in the filter which in turn affects how the INS errors are handled. Recall the error states in the filter are indirectly controlled through filter updates. Thus, if the quality of the updates fed into the filter suffers, so will how the errors are handled. These things added to the fact the kinematics are lowered and that there isn't heading aiding and you end up with a decrease in quality with which the INS heading is estimated. As was mentioned earlier, the SPAN filter benefits from a tightly-coupled implementation that has been maintained, improved, and tested for many years. The position/velocity/attitude differences between both filters can be seen in Table 5-4 below.

Table 5-4: Differences between author-generated INS and SPAN in open-sky test

	Mean	Std. Dev	RMS
Latitude (m)	0.055	0.273	0.279
Longitude (m)	-0.108	0.280	0.300
Height (m)	-1.034	0.262	1.068
Velocity N (m/s)	-0.013	0.042	0.044
Velocity E (m/s)	-0.024	0.043	0.050
Velocity U (m/s)	-0.031	0.059	0.067
Roll (deg)	0.059	0.259	0.266
Pitch (deg)	-0.058	0.381	0.385
Heading (deg)	0.085	0.935	0.939

Note there is a large position bias, especially in the vertical component, which is directly attributed to the pseudorange positions fed into the filter. Figure 5-18 below shows the difference in height between the raw pseudorange positions fed into the filter and the SPAN solution. The plot is also overlaid with the height standard deviation. This shows that SPAN is better at smoothing position estimates, partly because of the tightly-coupled integration and partly because of how the updates are weighted in the filter.

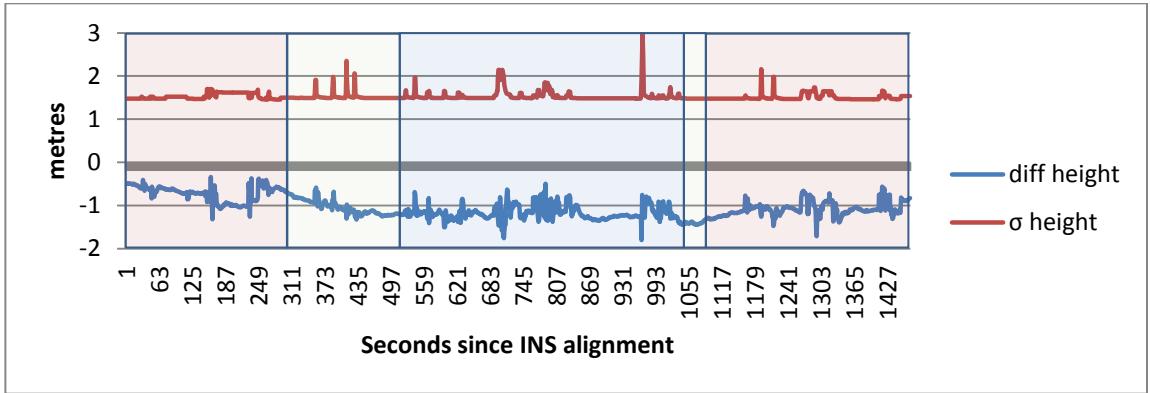


Figure 5-18: Height differences between raw pseudorange position and SPAN (open sky)

The velocity and attitude differences summarized in Table 5-4 show the loosely-coupled filter matched the SPAN solution well. The loosely-coupled filter is thus considered to be performing within its expectations.

5.6 Urban Canyon Kinematic Truth Test

The second truth test carried out to verify the robustness of the filter was in an urban canyon scenario in downtown Calgary as shown below in Figure 5-19.



Figure 5-19: Urban canyon in downtown Calgary (Google, 2015)

The start and end of the dataset were open-sky for the most part with the section of interest being the downtown core. The dataset has been split into six different colour-coded sections as shown below in Table 5-5 and Figure 5-20.

Table 5-5: Urban canyon sections

Section	Colour	Start	End
1	Blue	Macleod Trail and 29 th Ave. SW	Macleod Trail and 6 th Ave. SE
2	Red	Macleod Trail and 6 th Ave. SE	6 th Ave. SE and 3 rd St. SW
3	Green	6 th Ave. SE and 3 rd St. SW	9 th Ave. SW and 3 rd St. SW
4	Purple	9 th Ave. SW and 3 rd St. SW	9 th Ave. SW and Centre St. SW
5	Cyan	9 th Ave. SW and Centre St. SW	Centre St. SW and 5 th Ave. SW
6	Orange	Centre St. SW and 5 th Ave. SW	Memorial Dr. NE and 9 th St. NE

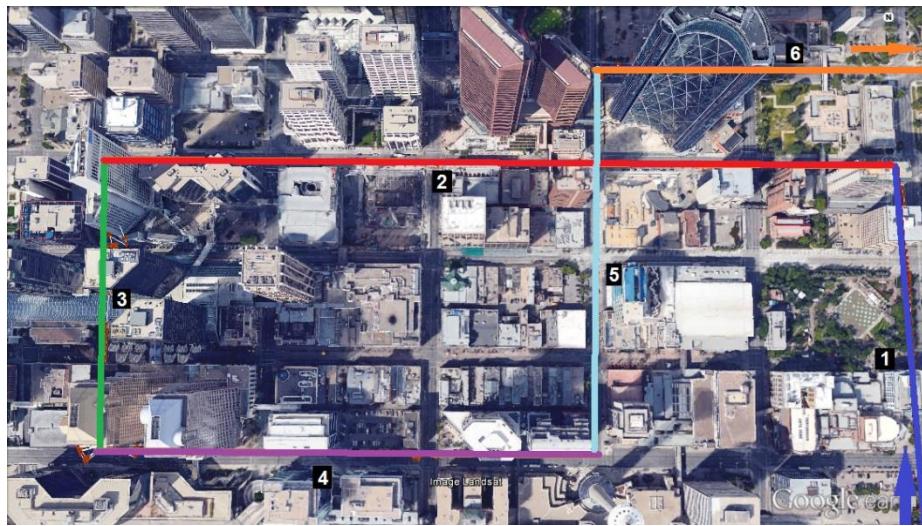


Figure 5-20: Urban canyon downtown core (Google, 2015)

Figure 5-21 below shows the loosely-coupled INS and SPAN tightly-coupled INS respectively in the downtown core.

The red Xs in the bottom plan view of Figure 5-21 represent the areas in which SPAN used the INS prediction without updates. The left-most part (section 3) is where most GNSS outages occurred. This can also be seen in the loosely-coupled plan view which is where the largest deviations with respect to truth are present. The tightly-coupled filter is going to produce better results due to the fact a full INS update can be performed when only 2 satellites are seen whereas an INS update can only be performed in the loosely-coupled solution if 4 satellites are seen. Section 3 for the loosely-coupled integration shows the INS performed well when in prediction-only mode. There are some issues towards the end of this section, due to lower quality pseudorange-computed positions fed as updates. The metrics will be seen in the upcoming plots.

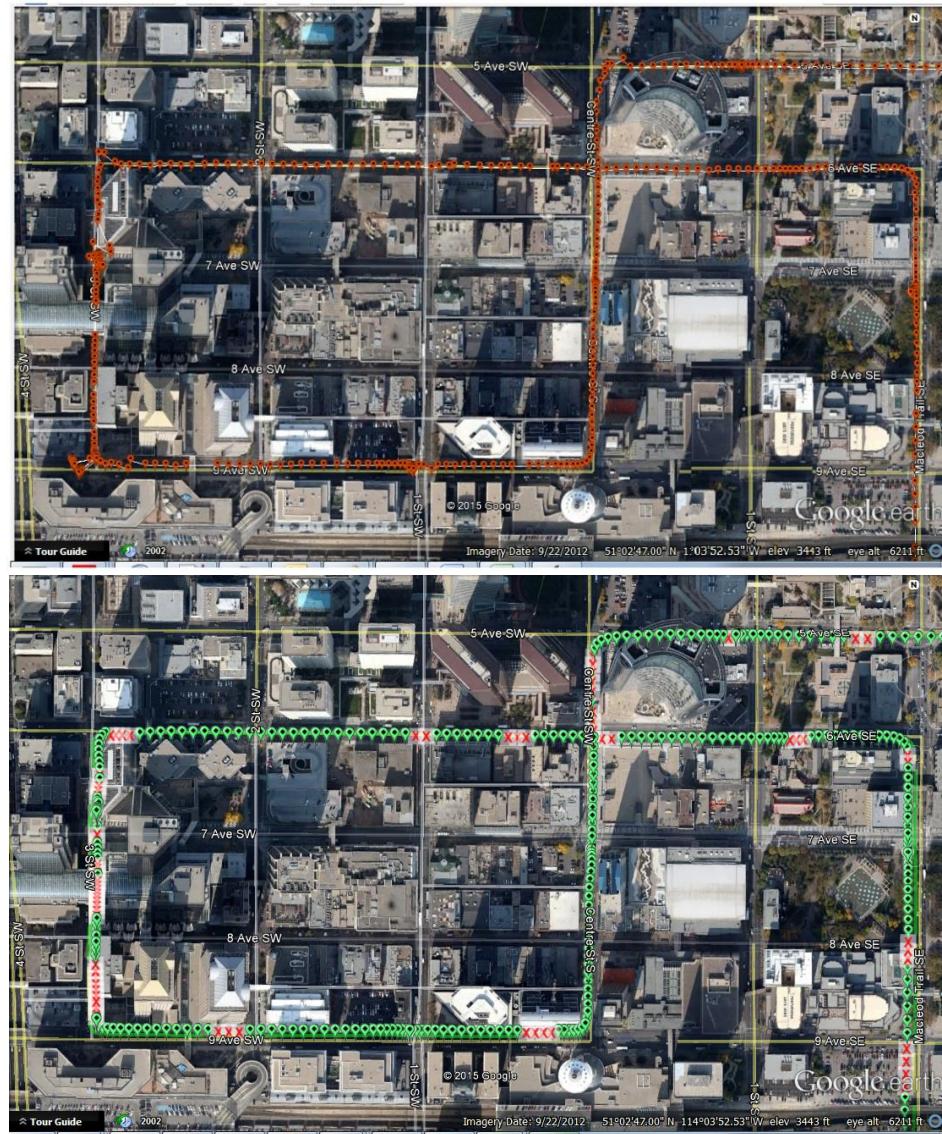


Figure 5-21: Loosely-coupled INS (top) vs. SPAN tightly-coupled INS (bottom)
 (Google, 2015)

Figure 5-22 shown below shows the difference in position between the loosely-coupled INS and SPAN. Section 1 is very good for the most part except for the part towards the end, which is where some of the high rise buildings start appearing. Section two was surprisingly very good given there were a lot of surrounding obstructions.

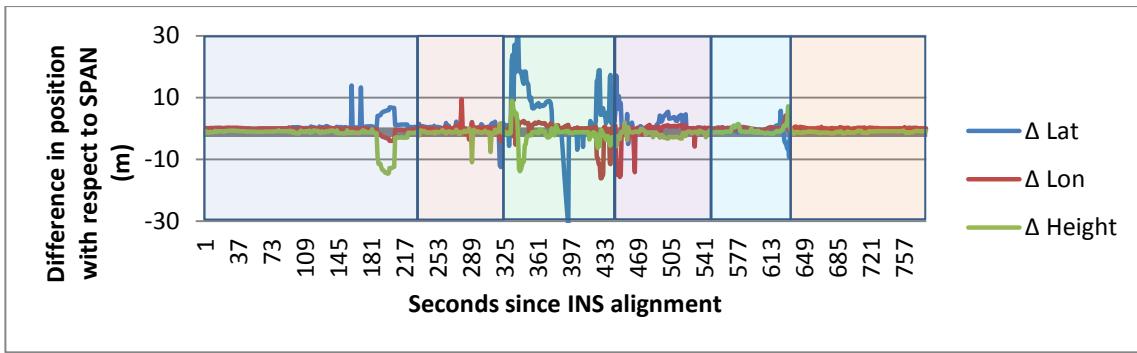


Figure 5-22: Position difference between SPAN and author-generated INS (urban canyon)

The big issues are seen in the third section, which is what was seen in Figure 5-21. It is important to note the error is in the latitude component, and that this third section traverses north-south. This then shows the effect of the vehicle motion constraints explored earlier. Without these constraints one expects to have a much larger longitude error than what was seen. The issues of the third section carry over to the fourth section in the dataset, which is where the largest longitudinal error is encountered. The longitudinal error is a product of lower accuracy position estimates being fed into the filter. Sections 5 and 6 are pretty good except for the interface between both. This issue is caused by the large building on the corner of Centre St. SW and 5th Ave. SW.

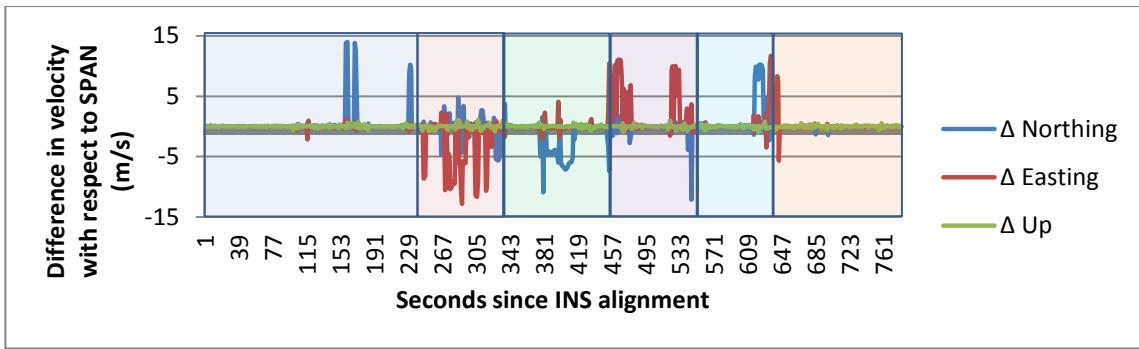


Figure 5-23: Velocity difference between SPAN and author-generated INS (urban canyon)

Figure 5-23 above shows the sections with largest velocity differences were a bit different compared to what was seen with the position. That is, section 2 showed the greatest difference in velocity with it being in the forward direction of motion. Just like it was seen in the position differences, the velocity differences are mostly in the along-track component. Section 4 has some velocity differences towards the start and the end and the same can be said about the interface between sections 5 and 6 which already had been detected as being caused by the building on the corner of Centre St. SW and 6th Ave. SW. The loosely-coupled filter then performed better in position compared to velocity in the urban canyon test. The sections where the greatest differences in velocity are seen match those where the vehicle wasn't static thus why ZUPTs could not be performed to minimize the errors.

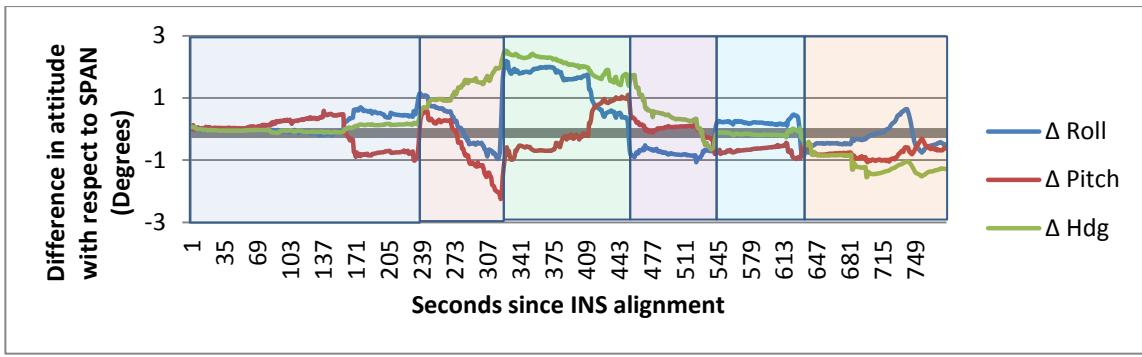


Figure 5-24: Attitude difference between SPAN and author-generated INS (urban canyon)

Figure 5-24 shows the attitude differences between both filters. The results were quite good given the circumstances, which shows the robustness of the filter for attitude determination. The reason why attitude performed better than position and velocity is that attitude is highly dependent on the process noise modelling of the INS errors. Position and velocity are being updated directly through observed updates, which in an urban canyon scenario are not optimal. The filter then needs to be smart enough to know when to use the presented updates and when to use the INS dead-reckoning using the IMU raw data only. This is a potential future enhancement that would improve the robustness of the filter in urban canyons. It can be seen in Figure 5-23 that the largest differences in pitch, roll, and heading are surprisingly not through sections 2 and 3, which is where the largest velocity and position errors were encountered. The estimates improve once the position and velocity updates improve in sections 4 -6.

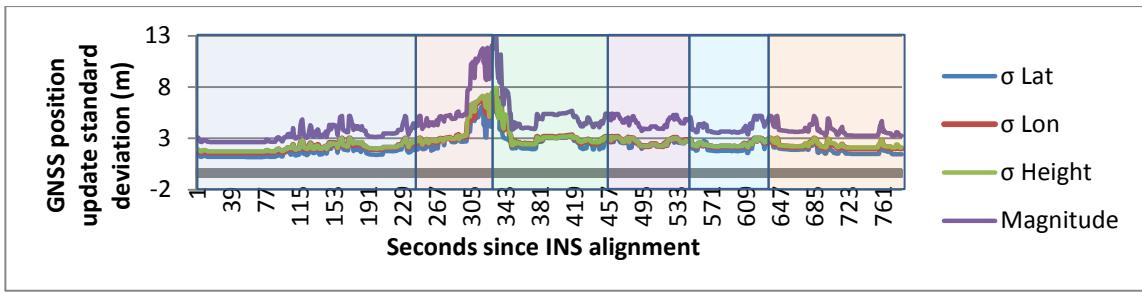


Figure 5-25: Standard deviation of pseudorange position updates fed into INS (urban canyon)

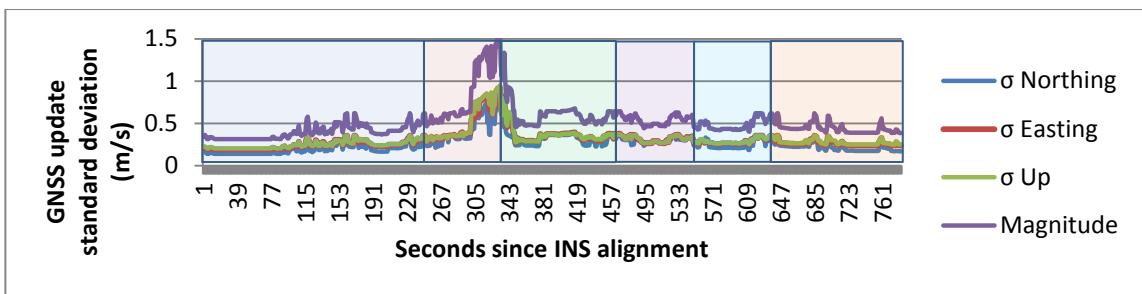


Figure 5-26: Standard deviation of velocity updates fed into INS (urban canyon)

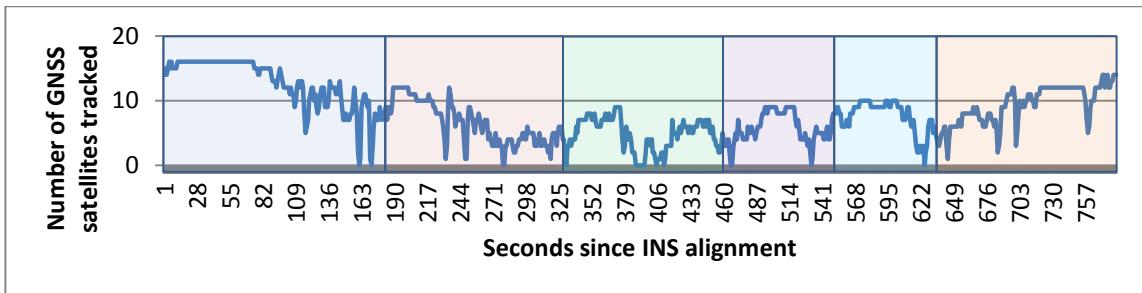


Figure 5-27: Satellites tracked (urban canyon)

Figure 5-25 to Figure 5-27 above show the quality of the updates being fed into the loosely-coupled filter. As expected, the largest update error is seen in section 2. Even

though the update quality improves in the third section, the errors from section 2 bleed into the INS for some time until the filter is able to stabilize the weighting of the updates vs. that of the update of the predicted INS solution. From the results seen above, things start improving towards the start of section 4. Surprisingly, the number of satellites seen throughout the entire data-set is more or less similar, which is why it can be concluded the issues in the accuracy of the updates in section 2 are mostly related to multipath in the environment.

Table 5-6 below shows the final metrics for the urban canyon test. Since most of the position errors were observed in section 3, which runs north-south, it makes sense that the highest RMS encountered in position is in latitude. Note the mean of the height is quite high as was seen in the open-sky test. Recall this was shown to be directly related to the pseudorange positions being fed as updates. The velocity RMS was similar in both latitude and longitude components. The low RMS for the upwards velocity component shows the robustness of the vehicle motion constraint under GNSS outages.

Table 5-6: Differences between author-generated INS and SPAN in the urban canyon test

	Mean	Std. Dev	RMS
Latitude (m)	1.000	4.623	4.730
Longitude (m)	-0.287	2.240	2.258
Height (m)	-1.627	2.398	2.900
Velocity N (m/s)	-0.046	2.239	2.240
Velocity E (m/s)	0.014	2.550	2.550
Velocity U (m/s)	-0.003	0.217	0.218
Roll (deg)	0.165	0.752	0.770
Pitch (deg)	-0.339	0.577	0.669
Heading (deg)	0.303	1.074	1.116

Table 5-7 below shows the RMS for each of the six sections previously identified in the dataset. The height difference RMS for sections 1 and 3 is higher than in the other sections due to a couple of spikes in the datasets related to outages and lower accuracy updates. The velocity values again show the along-track component is where the higher differences are seen. The attitude performed really well throughout, with the greatest issues being encountered in sections 3 and the start of section 6. The overall results of the filter can be considered successful given the fact its end use is with lower-end sensors. The SPAN tightly-coupled filter is a very robust filter that has been improved on for a number of years. Also, the fact it is tightly-coupled means it is able to perform full INS updates in more challenging conditions compared to the loosely-coupled filter. The loosely-coupled filter used some empirically derived process noise values for the LN200 but they could be further tuned to provide better results. This process is quite involved and is improved on over time.

Table 5-7: RMS of differences between author-generated INS and SPAN in the urban canyon test

	Section 1	Section 2	Section 3	Section 4	Section 5	Section 6
Latitude (m)	2.078	2.03	10.770	2.255	1.114	1.171
Longitude (m)	0.933	1.367	4.937	2.002	0.604	0.332
Height (m)	3.913	1.893	3.769	1.968	1.21	1.405
Velocity N (m/s)	2.304	1.382	3.141	0.824	3.537	0.659
Velocity E (m/s)	0.247	5.322	1.145	4.603	0.579	2.021

Velocity U (m/s)	0.1556	0.352	0.208	0.270	0.213	0.178
Roll (deg)	0.300	0.661	1.540	0.765	0.235	0.466
Pitch (deg)	0.470	1.017	0.726	0.196	0.686	0.796
Heading (deg)	0.096	1.334	2.076	0.698	0.160	1.15

CHAPTER 6: PROCEDURES AND RESULTS

Now that the filter has been verified with a high-end IMU, it is time to look into its robustness using the ultra-low-quality sensors inside the Moto-X Android smartphone. The first part of this chapter is an introduction to the sensors found inside the Moto-X smartphone. The second part of the chapter will show the procedure used prior to, during, and after acquisition. The third part of the chapter presents the magnetometer calibration results and the last part presents the results of the loosely-coupled filter with the Moto-X raw measurements under open-sky and urban canyon scenarios, similar to what was shown in Chapter 5 during the truth testing.

6.1 Sensors inside Moto-X smartphone

The Moto-X Android smartphone has sensors with the specified characteristics shown below in Table 6-1. From looking at the teardown for the Moto-X (iFixit, n.d.), it was possible to determine the GNSS receiver inside the Moto-X is the Qualcomm WTR1605L (Langley, 2016). This receiver is capable of tracking L1 GPS+GLONASS+Beidou (Klug, 2013).

Table 6-1: Manufacturer-reported characteristics of Moto-X smartphone sensors

	Accelerometer (m/s^2)	Gyroscope (rad/s)	Magnetometer (μT)
Name	LIS3DH	L3G4200G	AK8975
Manufacturer	ST Micro	ST Micro	Asahi Kasei
Maximum range	156.96	2000	1200
Resolution	4.79E-3	1	0.1

The manufacturer-reported specifications are not very useful for what is required in the loosely-coupled INS integration. Recall that in order for the INS to properly operate, it is required to have a detailed understanding of the sensor errors that can be removed deterministically and those that are modelled stochastically. What makes this very difficult for these types of sensors is the fact the deterministically determined values change with every system power-cycle and that these changes are quite sensitive to filter operation. The deterministic values ended up having to be determined empirically which was quite tedious and greatly impacts the ability of the loosely-coupled INS to be run in real time. Table 6-2 below shows the sensor error values used in open-sky and urban canyon environments. Note that SF stands for scale factor errors and NSD stands for noise spectral density.

Table 6-2: Moto-X INS sensor characteristics used

		Open-sky			Urban Canyon		
Time increment	s	0.0392			0.0392		
	units	X	Y	Z	X	Y	Z
Accelerometer bias	m/s ²	0.617	-0.383	0.057	0.441	-0.600	0.057
Gyroscope bias	rad/s	0.016	0.072	0.010	0.009	0.063	0.009
Time constants	s ⁻¹	-1/14400			-1/14400		
Accelerometer SF	-	0.01	0.01	0.01	0.13	0.13	0.01
Gyroscope SF	-	5.5	5.5	0.1	4.5	4	0.1
Accelerometer NSD	(m ² /s ⁴)/Hz	1E-4			1E-4		
Gyroscope NSD	(rad ² /s ²)/Hz	1E-4			1E-4		
Accelerometer bias NSD	(m ² /s ⁴)/Hz	1E-10			1E-10		
Gyroscope bias NSD	(rad ² /s ²)/Hz	1E-10			1E-10		

The first value that stands out from Table 6-2 is the time increment, also known as the inverse observation rate. Recall the observation rate for industry-used IMUs is quite high and is an even number such as 100 Hz, 125 Hz, or 200 Hz for the higher-end systems. However, in the Moto-X case, the observation rate is 0.0392^{-1} s = 25.51 Hz. The value isn't an integer such as in the other IMUs due to the fact the reported times

(just like any other reported observation) are quite noisy. Industry-used IMUs have real-time low-pass filters that allow for a fixed observation rate as per their respective specifications. Higher-rate IMUs are usually considered to have higher grade due to the fact they are integrating over a smaller time sample thus establishing the first order linearization approximation in the mechanization equations as a valid assumption. The lower the IMU rate, the larger the time increment, and thus the greater the risk of making an invalid assumption regarding first order approximations. This means that IMUs with higher operating rates can be used for high kinematics applications (e.g., automatically-guided military projectiles).

The second sets of values that stand out are the accelerometer and gyroscope biases. As can be seen, the values are completely different in both datasets. It was previously stated that the Moto-X sensors have a very high turn-on bias variability. The turn-on bias is a value determined through calibration for industry-used IMUs yet in this case the values are completely different every time the phone is power-cycled. A field calibration procedure described in the next section of this chapter was performed to come up with the values. The turn-on biases are quite important to get right. If these values are off then the entire solution will diverge.

Another very important set of values of are the scale factor errors. The accelerometers' scale factor errors are different for both datasets, but they don't vary as much as they do for the gyroscopes. This is expected as low quality accelerometers perform much better than gyroscopes. The specific forces are easier to measure than rates of rotation. Note how large the gyroscope scale factor errors are. Recall that IMUs used in the industry report scale factor errors in terms of parts per million due to how

small they are. It was also previously mentioned in Chapter 5 that scale factors for high-end IMUs can safely be ignored due to their small magnitude.

Recall from Chapter 3 that the EKF is modelling position, velocity, and attitude errors as well as sensor noise and sensor bias noise. These last two values are required to be modelled due to the fact it is not possible to find them deterministically. These values are tied to the noise of the raw measurements and the biases. In other words, these values are dependent on the noise in the raw measurements, which depend on whether the high frequency noise has been previously removed via a low-pass filter. The procedure applied to the Moto-X data is explained in the next section of this chapter. These values are initially estimated through the autocorrelation of low-frequency data mentioned in Chapter 4 and subsequently modified empirically.

Out of all the values shown in Table 6-2, the scale factor errors are the most important. Not only is the filter highly sensitive to these values, it is also not possible to find them through an in-field calibration. Thus, the scale factor errors end up being found entirely through empirical means. The biases are also quite important to the filter performance but they can at least be computed through the in-field calibration mentioned earlier. The filter is also very sensitive to the biases but at least there is a proven way to estimate them in the field. The noise spectral density values are quite important but the filter isn't as sensitive to these values as it is to scale factor errors and biases. Estimating all of these values is what is often called filter tuning. It is quite a tedious process requiring a lot of time and patience, especially for those values that are estimated empirically such as the scale factor errors, and noise spectral densities of sensor noise and sensor bias.

6.2 Procedure

The following section describes the procedure taken prior, during, and after acquisition for both open-sky and urban canyon environments.

6.2.1 Prior to Acquisition

The original concept of the thesis was to develop an Android loosely-coupled INS that would run in real time. A C++ program would be written to first verify filter adequacy prior to moving to the Android part. However, due to the already difficult task of creating the filter itself, and the highly variant turn-on biases and scale factor errors, the Android component of the project was limited to data acquisition. Instead, the C++ program is run offline with the raw data collected by the Android application.

An initial Android application was created. This application was based on the *Smartphone IMU GPS* application by Hector Kay (Kay, 2013). The original application outputs the raw data from any sensor, where each observation is time tagged with a timestamp indicating the number of seconds the phone has been turned on for. The so-called “GPS” sensor in the application outputs ellipsoidal latitude, longitude, height, horizontal accuracy, and horizontal speed. The application was modified so that the “GPS” sensor output the following:

1. Geodetic coordinates: latitude, longitude, height
2. Coordinates in e-frame

3. Velocity in e-frame
4. Velocity in l-frame
5. Positional accuracy in e-frame
6. Positional accuracy in l-frame

The idea of outputting all of the information above was in order to have position, position accuracy, and velocity in both e and l-frames. Recall the INS was mechanized in the e-frame so having the input information in the e-frame made it easier to feed the data to the INS. The l-frame observations were kept due to them being easier to relate to physically. There were two important assumptions that had to be made in order to be able to compute the estimates above. First, the course made good of the vehicle was used in order to find change in easting and change in northing. These values were paired with the horizontal velocity value reported originally by the application in order to find the easting and northing velocity components. Note upwards velocity was assumed to be zero. Knowing the latitude, it was then possible to rotate the l-frame velocities to the e-frame which is the frame used in mechanization. Secondly, the vertical position component was assumed to be two times the magnitude of the horizontal position accuracy. The assumption is derived from the fact NovAtel receivers report the vertical position accuracy to be 1.5 times that of the horizontal magnitude (Godha, 2014). Recall vertical positioning is less accurate than horizontal positioning due to the difference in VDOP compared to HDOP. Both assumptions proved to work just fine for the level of accuracy desired in the filter. It is also worth stating the velocity accuracy was derived from applying covariance propagation to the velocities from the course made good estimate.

6.2.1 During Acquisition

Two datasets were acquired with the Moto-X smartphone on August 22nd 2015, one under open-sky and another one in an urban canyon. The paths traversed were a bit different from those used for the truth performance tests. The same NovAtel equipment used in the truth tests was strapped to the car for truth trajectory comparison purposes. That is, a Flexpak6 with SPAN firmware, a GPS-702-GG antenna, and the IMU-LN200. The IMU was mounted in the same place as it was earlier so measuring the lever arms was unnecessary. The only thing required was measuring the offset between the antenna phase centre and the middle of the phone (taped to the roof of the vehicle). It is very tough from the iFixit Moto-X breakdown referenced earlier to tell the exact position of the antenna phase centre is. For the sake of simplicity, the antenna was assumed to be in the middle of the phone. Even if off by a few cm, this wouldn't be a big deal given the fact the difference in horizontal positioning is expected to be at the metre level due to the accuracy of the position updates fed into the filter. The centre of the phone was located 19 cm in front of the antenna phase centre, 28cm to the left, and 7.6 cm below; it was mounted in such a way that the IMU enclosure frame matched the vehicle frame for computational ease. The setup is shown below in Figure 6-1. Figure 6-2 shows the setup inside the vehicle.



Figure 6-1: Moto-X smartphone location

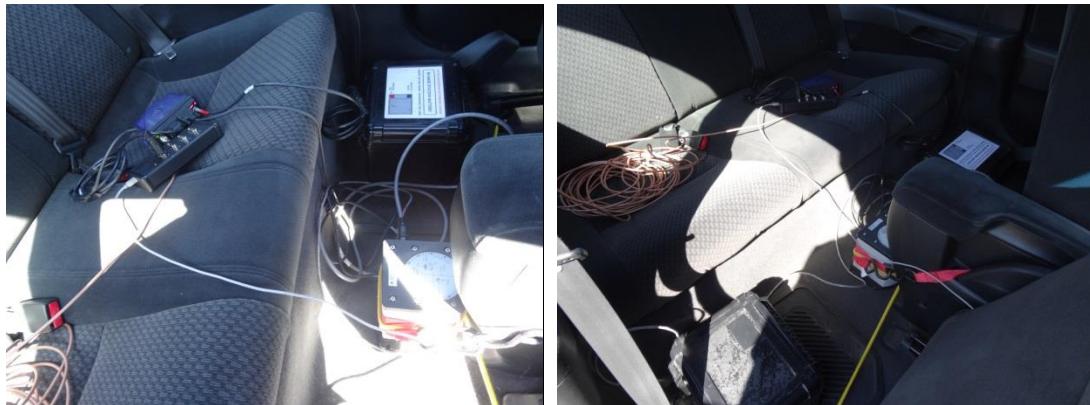


Figure 6-2: Location of Flexpak6 and IMU-LN200

Something important worth mentioning about the acquisition process is that the Moto-X has different selectable operation rates that are somewhat ambiguous. They range from *slow* to *fastest* or from *normal delay* to *fastest delay* depending on the application being used. These, as well as the actual measurement rates for the different options are shown below in Table 6-3.

Table 6-3: IMU operating rates in Moto-X Android smartphone

Terminology in application used	Official Android terminology	Official Android description	Operating frequency (Hz)
Slow	Normal delay	Rate (default) suitable for screen orientation changes	~ 10 Hz
Medium	UI (user interface) delay	Rate suitable for user interface	~ 25 Hz
Fast	Game delay	Rate suitable for games	~ 40 Hz
Fastest	Fastest delay	Get sensor data as fast as possible	~ 50 Hz

Tests were performed to see which rate would work best. The highest data rate was desired in order for the first-order approximation assumptions during mechanization to be as accurate as possible (i.e: to have shortest possible time between epochs). However, the fastest delay proved to create data gaps so the IMU rate was slowed down until data gaps weren't apparent. The chosen data rate for all datasets was *medium (user interface delay)*, which as mentioned earlier ended up being a 25.51 Hz rate (0.0392 s sampling interval) after the application of a low-pass filter, which will be explained in the following subsection.

The last thing worth mentioning is that there was a half a minute static period right after both systems were powered on. The reason was to be able to estimate the turn-on biases from Eqs. 4.16 and 4.17 (Kong, 2000).

6.2.3. After Acquisition: Discrete Wavelet Decomposition

Once the data was acquired and downloaded from the phone, it was fed through a wavelet decomposition script created through Scilab, the open-source MATLAB variant. Scilab uses many function calls from MATLAB making it very easy to use. The sensors inside the Moto-X smartphone have the signal buried by noise. The noise can be alternatively taken into account through the spectral noise densities in the filter. The wavelet signal decomposition method was chosen due to it being more scientifically based rather than trial-and-error based.

The purpose of wavelet decomposition is to project the signal onto a family of basis functions generated through compressions and translations of what is called the mother wavelet; where the mother wavelet is a short oscillating zero mean function that decays quickly at both ends (Nassar, 2003; El-Sheimy et al., 2003). Wavelet decomposition splits the original function into a low frequency signal referred as the approximation and a high frequency signal referred as the detail. The approximation is then further decomposed into its own approximation and details section, and so forth. Every time this happens, the signal is being decomposed further and is referred to as the level of decomposition. The level of decomposition (LOD) is crucial in controlling how much filtering should be done. Also of importance is that according to the Nyquist theorem, if

a signal is sampled at f_s the highest frequency component of the signal in the first level of decomposition is $0.5f_s$. The second level of decomposition would be half of that ($0.25f_s$) and so on (Langley, 2016). This applies to each level of decomposition and is an important concept which will be used in determining at what level of decomposition to stop. Having more levels of decomposition than required will end up filtering out actual observed kinematics while having a lower level of decomposition than required will not filter out as much noise as required. For a detailed introduction to wavelets readers are asked to refer to Nassar (2003) and El-Sheimy et al (2003). Figure 6-3 shown below illustrates the wavelet decomposition process.

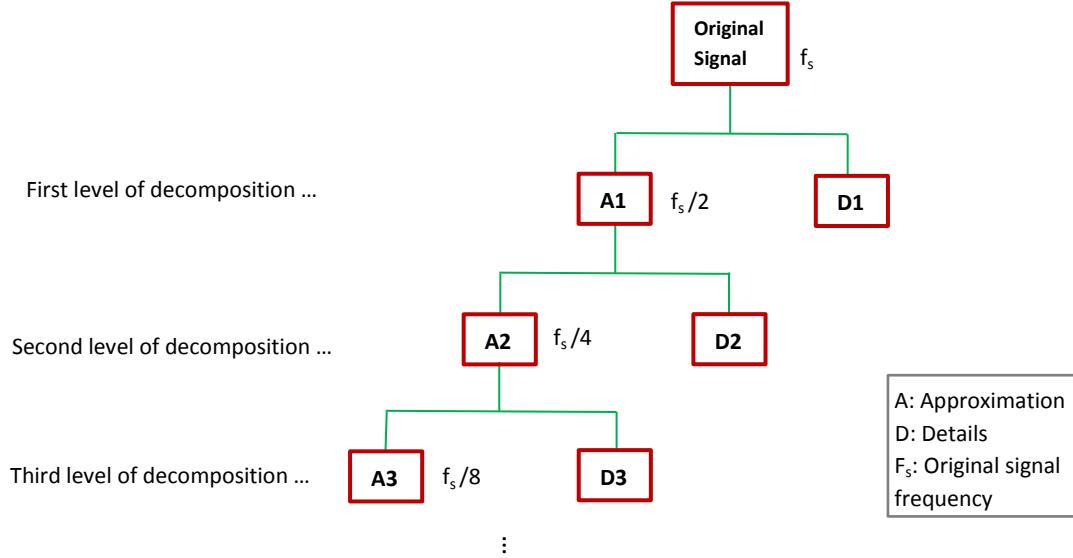


Figure 6-3: Wavelet multiple level of decomposition (adapted from Nassar, 2003)

Finding the level of decomposition for static data is simple because the only forces present are gravity, Earth's rotation rate and biases or any long term errors (*ibid*). Kinematic datasets like the ones dealt with here have accelerations and rotations whose

effects with the data need to be maintained. The procedure used for determining the wavelet level of decomposition for a kinematic dataset is as follows. To keep the section concise, only the plots corresponding to the raw x-axis accelerometer signal will be shown but the procedure was applied to all accelerometer, magnetometer, and gyroscope signals in both open-sky and urban canyon datasets. The procedure was followed from Nassar (2003).

1. Compute Fourier transform of raw signal. Since the IMU operating rate in this case is 25.45 Hz, the Nyquist frequency is half the sampling rate (12.7 Hz) and corresponds to the highest frequency that a sampled dataset can be reproduced without error. The frequency response is shown below in Figure 6-4.

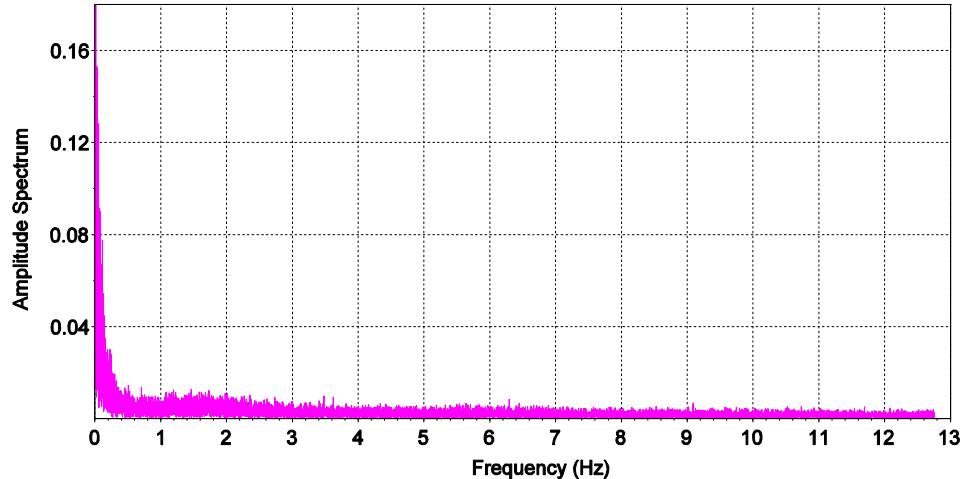
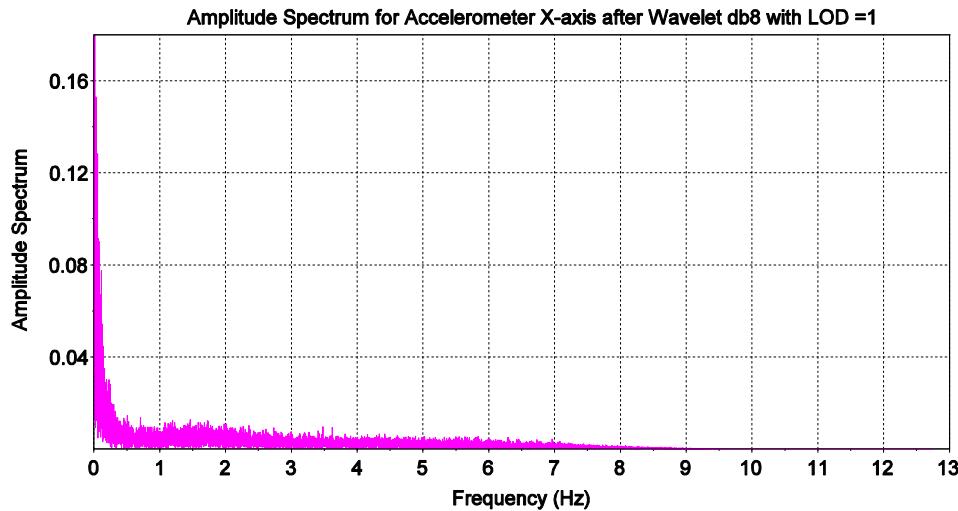


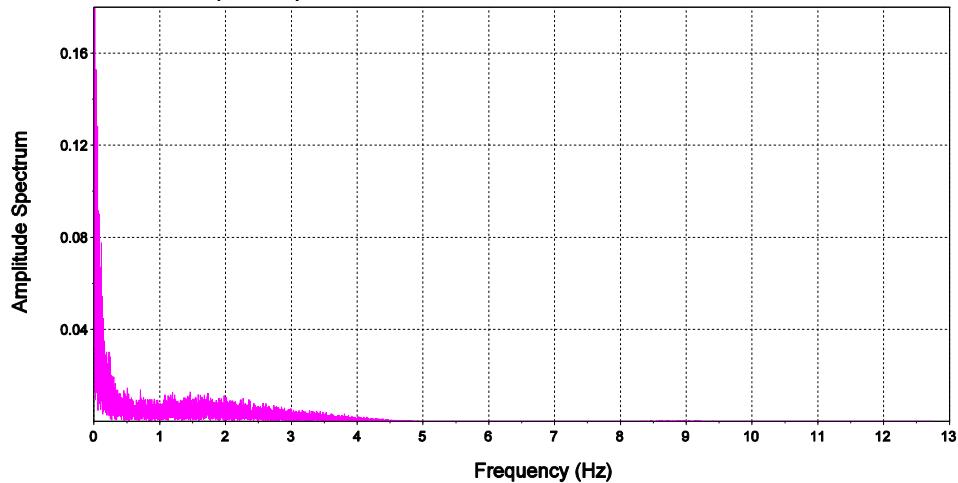
Figure 6-4: Amplitude spectrum for raw x-axis accelerometer (open-sky)

2. Use the frequency response plot to determine the frequency threshold of usable motion kinematics and high frequency noise. For the x-axis accelerometer signal shown above, this would be around 0.5 Hz.

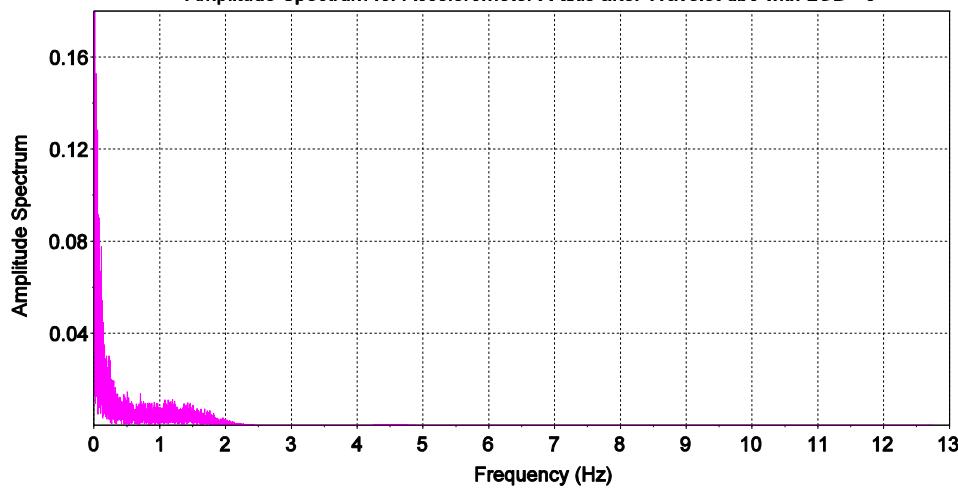
3. Run discrete wavelet at different levels of decomposition until the frequency response of the remaining approximated (low frequency signal) does not contain any responses beyond the threshold specified in step 2. The db8 family of wavelets was selected due to it being previously successfully used in INS data by Gao (2007). Figure 6-5 below shows the first five level of decompositions (LOD).



Amplitude Spectrum for Accelerometer X-axis after Wavelet db8 with LOD =2



Amplitude Spectrum for Accelerometer X-axis after Wavelet db8 with LOD =3



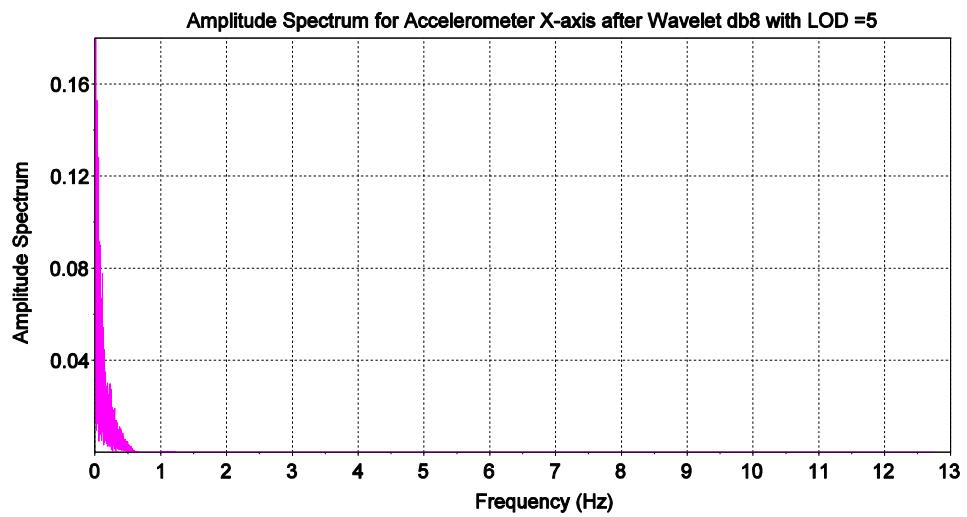
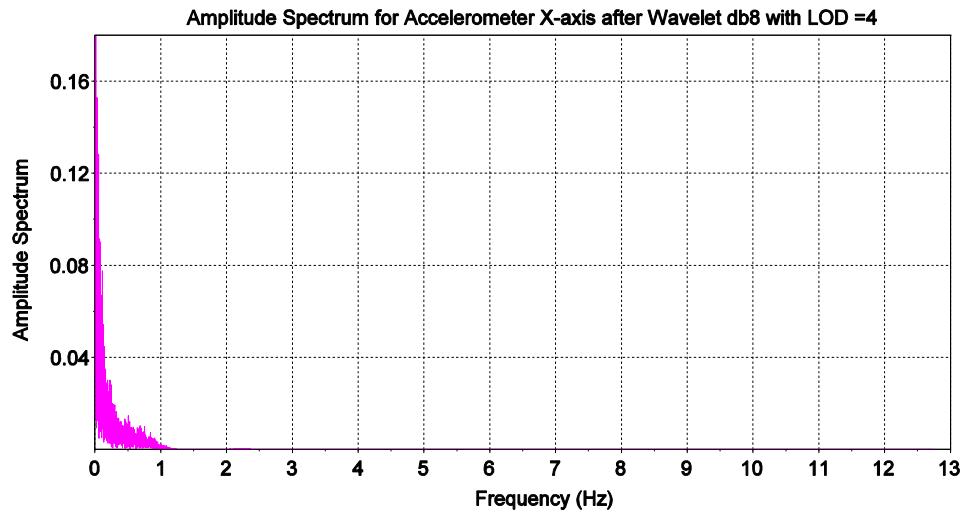


Figure 6-5: First 5 LOD for raw x-axis accelerometer (open-sky)

4. Use final approximated signal (LOD=5 from above) as raw data in INS. The filtered x-axis accelerometer data is shown below in Figure 6-6.

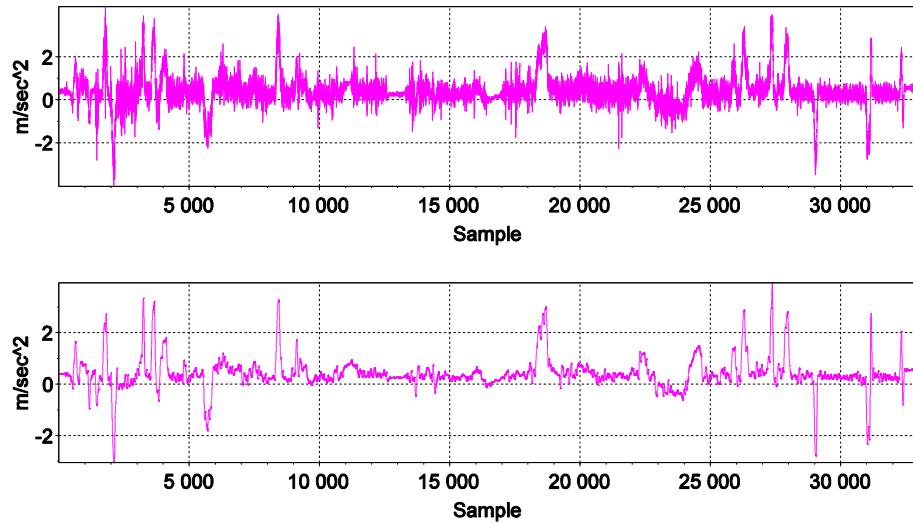


Figure 6-6: Original and filtered x-axis accelerometer (open-sky)

The procedure for filtering the time signal (i.e. the timestamps, which are shared for every sensor signal) was different due to the fact the signal is not experiencing any kinematics. Timing is very important because the time between IMU observations is required in both the mechanization equations and the EKF. In this case, we are interested in filtering the Δt signal; that is the change in time from epoch to epoch. Should this value be off, it would be impossible to get the INS to converge to a true solution. The procedure for filtering noisy static signals is again developed from Nassar, 2003.

1. Apply wavelet decomposition to static data at different levels of decomposition.
2. Compute standard deviation for each obtained approximation component.
3. Proper LOD is the one after which the standard deviation reaches a minimum value.

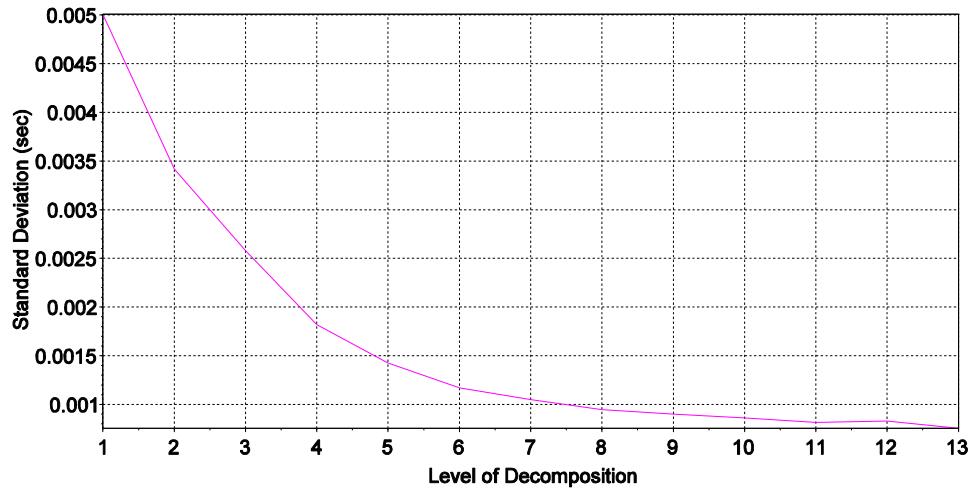


Figure 6-7: Standard deviations of approximations for each LOD

Figure 6-7 shown above shows the first standard deviation minimum (before it increases again) is found at LOD of 11. Figure 6-8 shows the raw and filtered Δt signal. This is how the 0.0392 s between epochs used in the INS was found. Note that originally, the idea was to use the raw Δt from each epoch difference. However, as shown in Chapter 2, the success of the mechanization equations depends on the valid assumption the kinematics can be represented by a first order approximation. The added noise from the raw time signal proved to be relevant enough to cause the filter to diverge due to invalid duration of integration. Filter convergence was attained when removing the noise from the time signal.

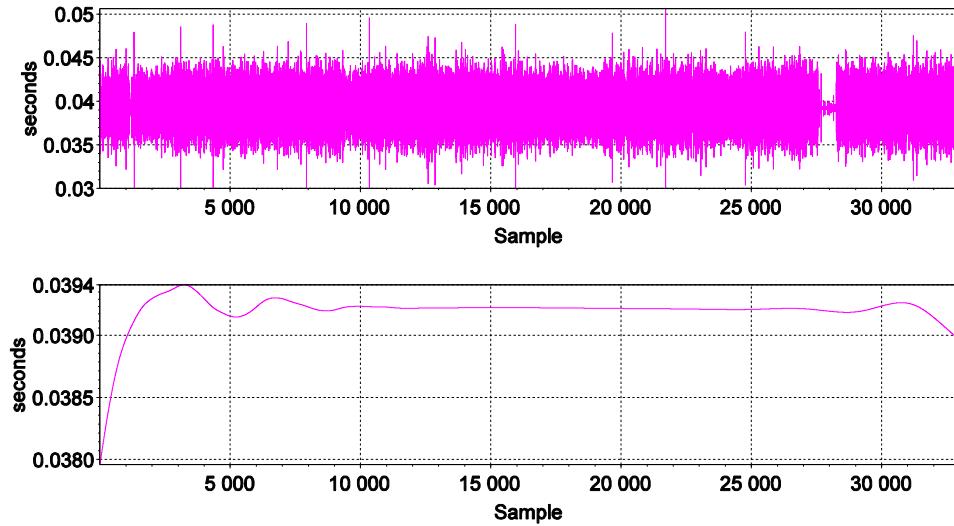


Figure 6-8: Comparison of original and filtered Δt signal

6.3 Magnetometer Assessment

The Moto-X smartphone has three orthogonal AK8975 Hall effect magnetometers that can be used for determining the phone's azimuth. Hall effect magnetometers are the most common magnetic sensing devices used and consist in producing a voltage across a metallic surface in response to a magnetic field perpendicular to the metallic surface (Cai et al., 2012; Langley, 2003). These types of magnetometers lend themselves more as switches (e.g: for an anti-lock braking system) than as heading measuring devices due to their lower measuring accuracy and significant drift. Hall effect magnetometers produce smaller output signals and have comparatively low sensitivity and stability compared to other methods such as anisotropic magnetoresistive (AMR) sensors; AMR sensors have been shown to produce accurate heading estimates after being calibrated (Renaudin et al., 2010; Langley, 2003). Even so, that study only dealt with static

headings and calibration of the environment in which the magnetometer was mounted. The ferromagnetic components of the outside environment in a kinematic application will certainly affect the raw measurements.

The smartphone magnetometers measure the sum of geomagnetic field plus any magnetic interference caused by ferromagnetic components in the circuit board (Ozyagcilar, 2012a). The magnetic interference is caused by hard-iron and soft-iron effects. The former are caused by permanently magnetized components in the circuit board and induces a bias in the magnetometer measurements. The latter distorts the raw magnetometer measurements and is caused by temporarily magnetized components in the circuit board (ibid). Five calibrations were done on different days in order to find how consistent the values are. Table 6-4 below shows the final results. The offset represents the hard-iron effect and the W^{-1} unitless matrix represents the soft-iron effect (see Appendix B for further details on magnetometer calibrations).

Table 6-4: Magnetometer calibration results

Calibration	Bfield (μT)	Offset (μT)	W^{-1}
Magn_June6_2015.csv	65.0014	$\begin{pmatrix} 0.0224 \\ -0.0185 \\ 0.2652 \end{pmatrix}$	$\begin{pmatrix} 0.0155 & -0.006 & -0.0004 \\ -0.0006 & 0.0150 & 0.0001 \\ -0.0004 & 0.0001 & 0.0157 \end{pmatrix}$
Magn_Aug1_2015.csv	63.4919	$\begin{pmatrix} -0.4077 \\ -0.1408 \\ -0.2064 \end{pmatrix}$	$\begin{pmatrix} 0.0159 & -0.0002 & -0.0003 \\ -0.0002 & 0.0153 & 0.0001 \\ -0.0003 & 0.0001 & 0.0160 \end{pmatrix}$
Magn_Sept5_2015.csv	63.0598	$\begin{pmatrix} -0.1447 \\ -0.5405 \\ -0.1861 \end{pmatrix}$	$\begin{pmatrix} 0.0160 & -0.0004 & -0.0004 \\ -0.0004 & 0.0154 & 0.0001 \\ -0.0004 & 0.0001 & 0.0161 \end{pmatrix}$
Magn_Dec10_2015.csv	63.8331	$\begin{pmatrix} -0.2162 \\ -0.0087 \\ 0.0736 \end{pmatrix}$	$\begin{pmatrix} 0.0159 & -0.0005 & -0.0003 \\ -0.0005 & 0.0154 & 0.0001 \\ -0.0004 & 0.0001 & 0.0161 \end{pmatrix}$

Magn_Jan3_2015.csv	62.1444	
		$\begin{pmatrix} -0.3116 \\ -0.2122 \\ -0.0795 \end{pmatrix} \quad \begin{pmatrix} 0.0164 & -0.0004 & -0.0004 \\ -0.0004 & 0.0156 & 0.0001 \\ -0.0004 & 0.0001 & 0.0164 \end{pmatrix}$

Table 6-4 shows the hard-iron effects are very small compared to the magnitude of the magnetic field, both of which are shown in units of μT . The soft-iron effect matrix, W^{-1} , appears to be fairly consistent for all calibrations. The five values were tested on both open-sky and urban canyon datasets to find how much of an effect the values would be and the results were pretty much the same. Figure 6-9 below shows the results of the Moto-X magnetometers compared to SPAN (truth) in the open-sky test.

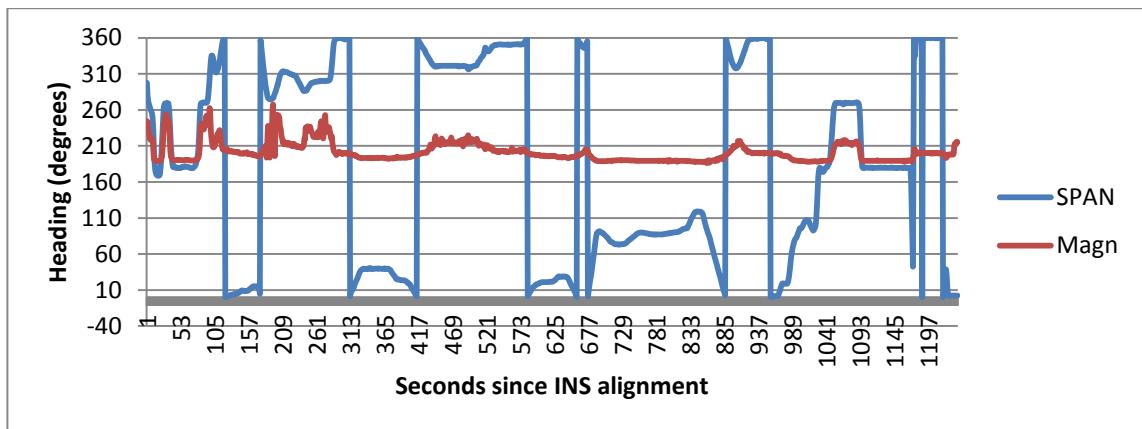


Figure 6-9: Heading from magnetometer compared to SPAN (Open-sky)

The open-sky test shows the magnetometer-derived heading closely followed that of the SPAN heading until about the 60th epoch. From this epoch onwards, the magnetometers appear to be recognizing the changes in heading but are not as sensitive as they should be causing the heading estimates to never go beyond 260 degrees and below 188 degrees. Even though the magnetometer-derived heading is quite off from the

truth, it somewhat follows the shape of the SPAN heading. In fact sections in the dataset where the heading does not change are properly represented in the magnetometer-derived heading. This means the sensor errors are not completely random but are rather related to sensitivity, scale factor errors, and system biases that haven't been removed. A few scale factor errors were empirically tested but they didn't help much so the major culprit appears to be a sensitivity issue. It is also worth noting that the magnetometer was previously tested in the same parking lot used for static filter validation in Chapter 5. In these tests, the Android orientation sensor was used to assess the validity of the sensors while being far from any metal objects/surfaces with unsuccessful results. Given previous successful tests of smartphone magnetometers such as in Mourcou et al. (2015), it is possible the magnetometers inside the Moto-X used in this thesis research are defective.

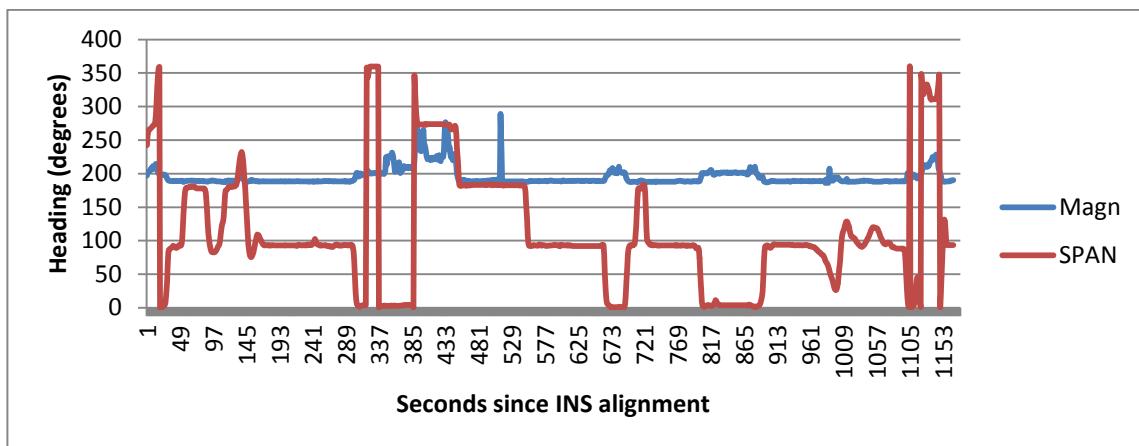


Figure 6-10: Heading from magnetometer compared to SPAN (Urban Canyon)

Figure 6-10 above shows the magnetometer-derived heading as well as that of the SPAN solution. The first thing that stands out is the changes in heading are much less

apparent in this scenario which could be attributed to how different the environment is compared to the open-sky test. The difference between both is much more substantial in this dataset.

In order for the magnetometers to be able to provide heading aiding to the loosely-coupled INS they certainly need to be able to have more accurate heading estimates than what was possible with the sensors inside the Moto-X sensor. Studies regarding phone magnetometer use for heading estimation have been previously made in similar urban environments (e.g. Blum et al. (2012)) with similar unsuccessful results and those were done with smartphone magnetometers that proved to work unlike the one used in this thesis research. The upcoming Moto-X results therefore do not consider any type of heading aiding.

6.4 Open sky Results

The main purpose of this thesis is to assess the performance of the loosely-coupled INS using the Moto-X Android smartphone sensors. Just like it was done with the truth testing of the filter, there are two datasets; the first is in open-sky and the second in an urban canyon scenario. Both tests were carried out in Calgary, AB. The open-sky test started at 1120 – 68th Ave. NE and consisted in traversing northwards towards the Cross Iron Mills outlet mall, just outside of Calgary. There is a quicker, more direct route through Deerfoot Trail but the chosen route contained bigger changes in heading and passed through multiple traffic lights. The trajectory is shown below in Figure 6-11.

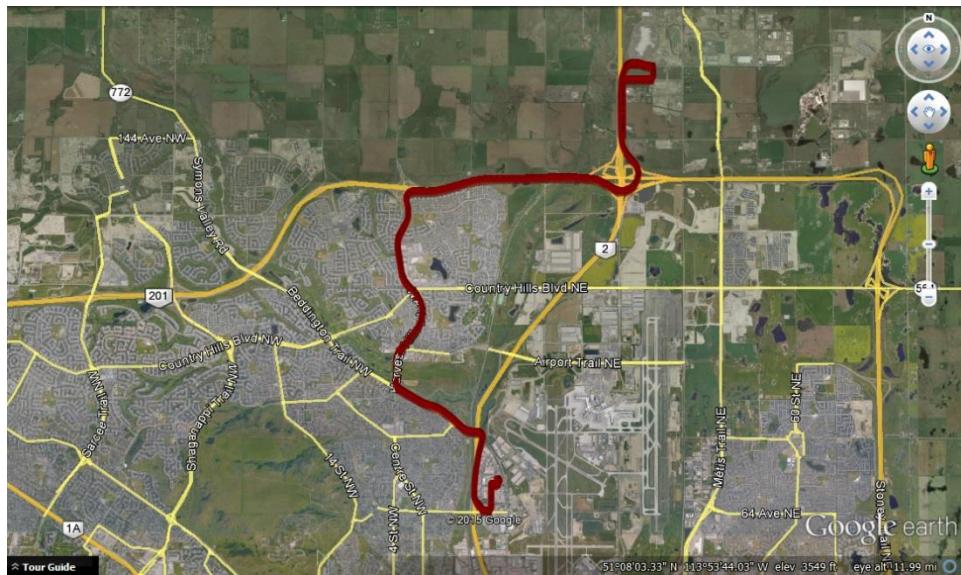


Figure 6-11: Open-sky dataset with Moto-X (Google, 2015)

An assessment similar to the ones that were carried out in the truth testing part of the thesis was performed here. That is, the route traversed was split up into different sections to make it easier to identify the different sections when doing the analysis. For this open-sky test, the route traversed was split up into five different sections shown below.

The first section starts at 1120 – 68th Ave. NE and goes right until after the Deerfoot Trail overpass that turns into Beddington Trail NW. The very start of this section contains speeds of around 50 km/hr with a few tight heading changes at first on the way to 64th Ave. NE. The vehicle then speeds up to 100 km/hr once it has merged onto Deerfoot Trail before slowing down to around 70 km/hr for the end of this first section. This part does not contain a lot of changes in heading.



Figure 6-12: Section 1 & Section 2 of open-sky dataset (Google, 2015)

The second part of the dataset does not contain a lot of changes in heading. It starts from when it merged into Beddington Trail NW and ends right after merging into the northward bound Harvest Hills Blvd. NW. There weren't any stops in this section as the traffic light at the intersection of Beddington Blvd. NE and Beddington Trail NW was green when the vehicle passed. Speeds were between 60 km/hr and 85 km/hr.

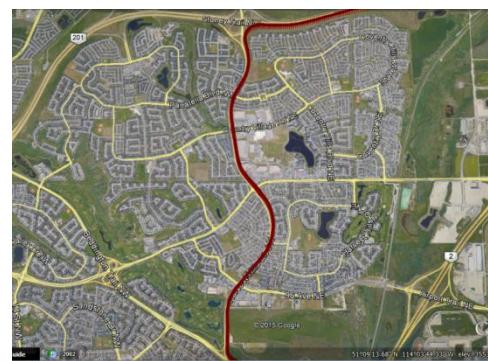


Figure 6-13: Section three of open-sky dataset (Google, 2015)

The third section shown above in Figure 6-13 was traversed northbound on Harvest Hills Blvd. NW. There are a lot of stop lights in this section and speed was limited to 65 km/hr. There are a few slight changes in heading but no tight corners.



Figure 6-14: Section four of open-sky dataset (Google, 2015)

The fourth section of the open-sky dataset (shown above in Figure 6-14) starts as soon as the car has merged onto Stoney Trail NW and goes up until it has merged back onto Deerfoot Trail. This section contains speeds of up to 115 km/hr and eastwards up until the long overpass that merges into Deerfoot Trail. There were many vehicles around this section but there aren't any buildings that could induce multipath into the GNSS receiver.

The fifth and last section of the open-sky test is shown below in Figure 6-15. It starts Northbound on Deerfoot Trail until merging onto Crossiron Dr. However, unlike the truth test shown in Chapter 5, the vehicle did not go to Cross Iron Mills. Instead it took a right on Deerfoot Blvd. and then drove Southwards on a road that parallels with Deerfoot Trail. The test ended at the intersection of this road (Writing Creek Crescent) with 144 Ave. NE.



Figure 6-15: Section five of open-sky test (Google, 2015)

Table 6-5: Start, end epochs and cumulative time for open-sky test

Section	Start epoch	End epoch	Time	Cumulative time
1	1	186	3' 5''	3'5''
2	187	289	1'42''	4'47''
3	290	693	6'43''	11'30''
4	694	922	3'48''	15'18''
5	923	1172	4'9''	19'27''

Recall that throughout these tests, the SPAN unit was strapped to the car in order to have a truth dataset to compare the vehicle against. The following plots assess the

validity of the loosely-coupled filter using Moto-X sensors with that of the tightly coupled SPAN system that uses the high-end IMU-LN200.

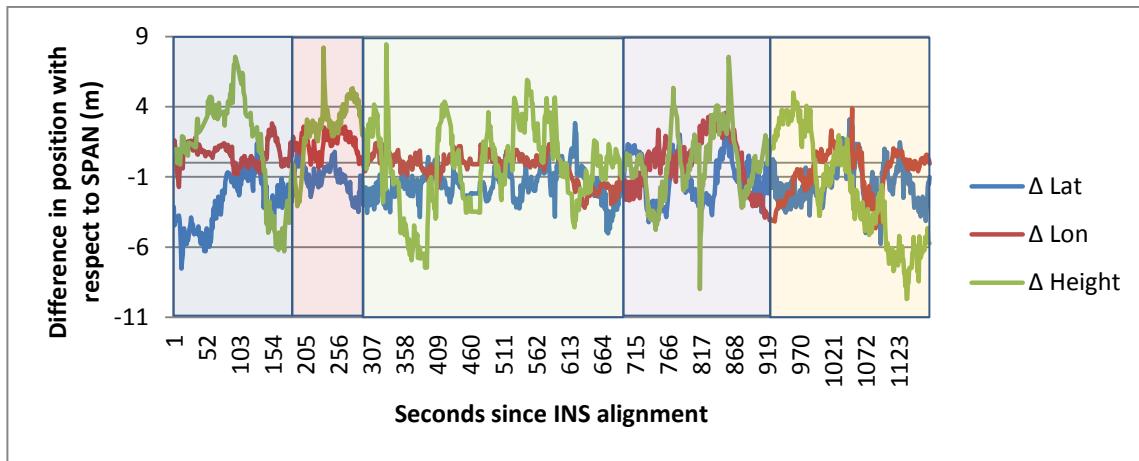


Figure 6-16: Difference in position with respect to SPAN (open sky)

The first plot shown above in Figure 6-16 shows the distance separation between the loosely-coupled solution and SPAN. The first thing that stands out is that the height component contains the highest deviations out of the three and is also the noisiest. Other than being the least accurate component, recall that the standard deviation of the height measurements were not reported by the phone. Instead, these values were assumed to be 2 times the magnitude of the horizontal accuracy (using NovAtel's 1.5 times horizontal magnitude as a guide). There isn't any clear pattern as far as position quality is concerned with respect to the different sections of the dataset identified earlier. There are a few spikes in vertical accuracy but these are not directly related to the accuracies of the input positions fed into the filter as shown below.

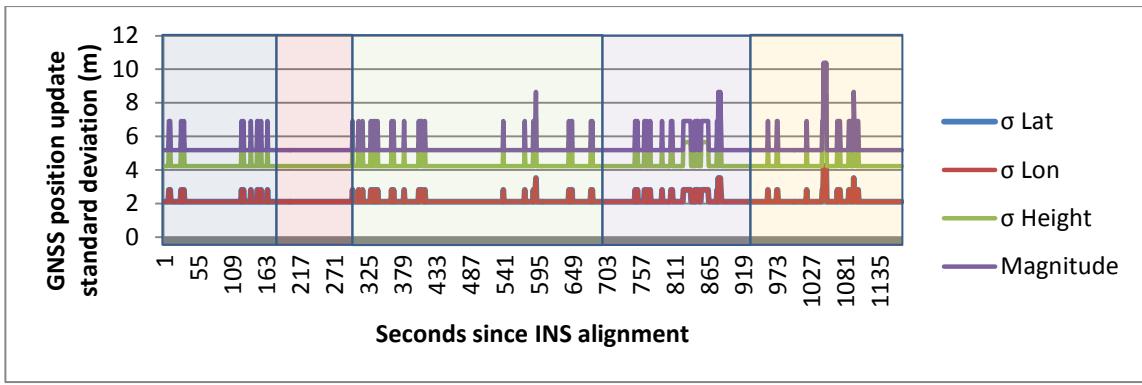


Figure 6-17: GNSS position update standard deviation (open sky)

Figure 6-17 shown above shows the standard deviation of the GNSS positions that were used as updates in the filter. Recall that the phone only reports a horizontal accuracy. This standard deviation was split up into northing and easting components using the changes from the course made good computation. Figure 6-17 shows there are quite a few spikes in the dataset but there aren't any in section two of the dataset. At first, it appears this could be due to section two having clearer sky and being away from multipath-inducing obstructions. However, the spikes are prevalent in the fourth section of the dataset, which takes place in Stoney Trail under what is considered to be away from any buildings that cause multipath. The difference is that Stoney Trail (section four) was much more congested than Beddington Trail (section two). Also, around the time data collection took place, there were a number of large vehicles around the test vehicle, which induced much more multipath. One last thing worth mentioning is the magnitude of the three dimensional accuracies is 6 metres at minimum and goes up to over 10 metres, with the height component being the main culprit at why it is so high.

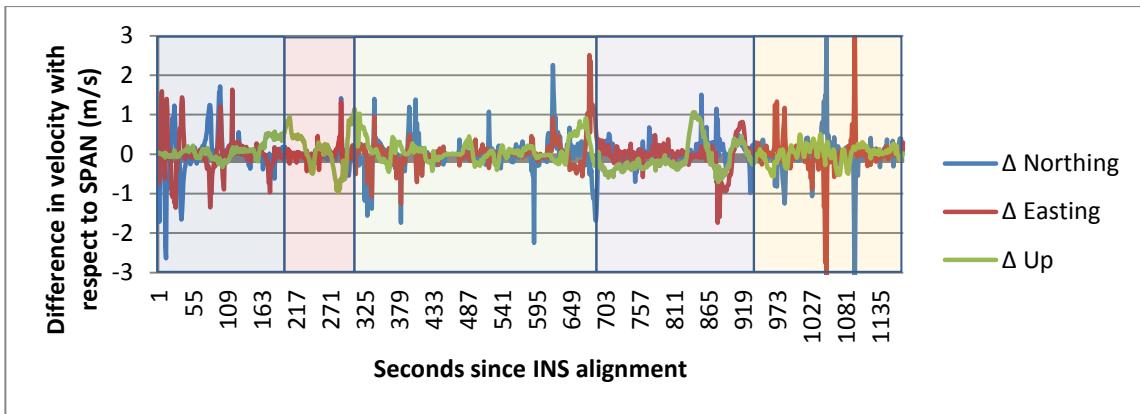


Figure 6-18: Difference in velocity with respect to SPAN (open sky)

Figure 6-18 shows the separation in velocity between the loosely-coupled filter and SPAN. The start of the dataset shows a bit of a difference in northing and easting, which can be attributed to initial bias convergence taking place in the filter. There are quite a few spikes in the data which indicates noisy filter updates. Unlike with position, there was no indication of the velocity standard deviation from the GNSS receiver in the phone. Again, since the loosely-coupled filter uses GNSS position and velocity as updates, an estimate of some kind was required. The velocity standard deviation was computed by performing covariance propagation on the velocities found by differencing subsequent positions and dividing them by the change in time. Although far from ideal, this assumption was definitely required in order to have some sort of metric with which to judge the incoming velocities in the filter. After all, the standard deviations of the updates are used to determine how much the updates should be trusted. Having access to velocity standard deviation directly from the GNSS would have been better. Recall GNSS velocities are usually computed through Doppler measurements.

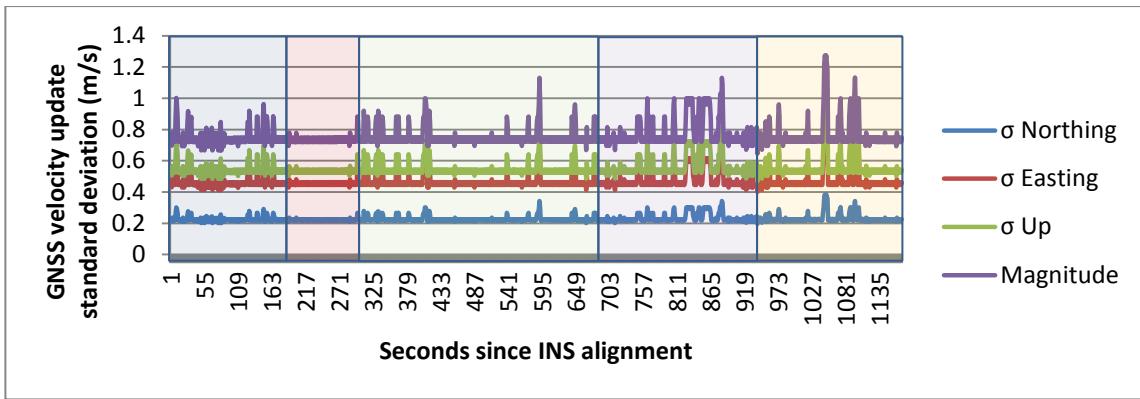


Figure 6-19: GNSS velocity update standard deviation (open sky)

Figure 6-19 above, shows the accuracies of the velocities used as updates in the loosely-coupled INS filter. Note the pattern of the spikes is identical to Figure 6-18 which makes sense considering the fact these were directly derived from position, hence the same analysis applies. The magnitude is about 80 cm/s throughout which is quite high and will have an impact on the attitude reported by the INS.

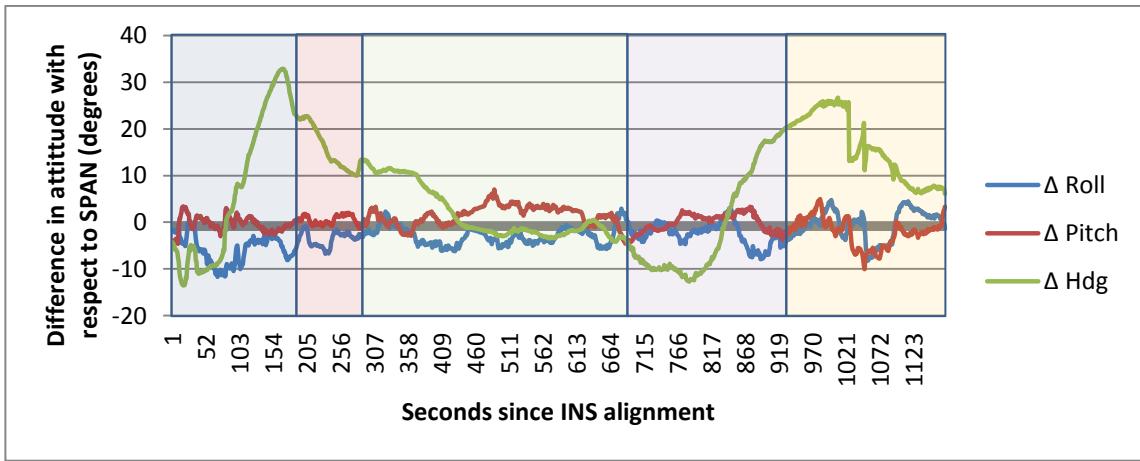


Figure 6-20: Difference in attitude with respect to SPAN (open sky)

Figure 6-20 shown above shows the attitude performance in the filter. Pitch and roll are bounded within 10 degrees throughout. This is good considering there are a number of pitch and roll changes encountered at the end of each section in the form of over/under passes. The first one is encountered at the end of the first section as the vehicle merges onto Beddington Trail NE, the second at the end of the second section as the vehicle merges onto Harvest Hills Blvd. NW; the third one as the vehicle merges onto Stoney Trail and the last one, which is the biggest one as the vehicle merges back onto Deerfoot Trail. Recall as was mentioned earlier that pitch and roll are dependent on removing gravity from the observed specific forces, provided the states in the filter are being well modelled. Thus, we expect to have pitch and roll metrics somewhat equivalent throughout the four sections of the dataset.

As has been previously mentioned, the hardest component to accurately model is heading. It was stated before that heading depends on kinematics as well as sensor quality and how well it is being modelled. From Figure 6-20 we can see heading starts a bit off from the SPAN truth (around 10 degrees), before worsening dramatically around epoch 154 in section 1. The fact the heading converges afterwards indicates some of the initial biases might have taken some time to converge meaning the initial estimates might have been a bit off. The estimate then improves through sections two and three which encounter slight heading changes at a speed of up to 65 km/hr. The heading component might have been helped from the more accurate updates that took place in section two and it is encouraging to see it continued to improve through the lower input position and velocity updates that were shown in Figure 6-18 and Figure 6-20. The heading difference starts increasing again in the third section of the dataset as the vehicle

is on the overpass that merges into Deerfoot Trail. The GNSS position and velocity inputs show a steady decrease in quality during this period presumably from multipath of surrounding vehicles which could explain the effect on the heading estimation. Section 4 does not contain much improvement as far as quality of inputs are concerned but there are a number of steep turns that help converge the heading difference back to the sub-10-degree range. The two spikes shown in this section four directly correlate to when the vehicle makes the sharp turns. Another way of seeing the heading difference is by plotting it for both author-created INS filter and SPAN as shown below.

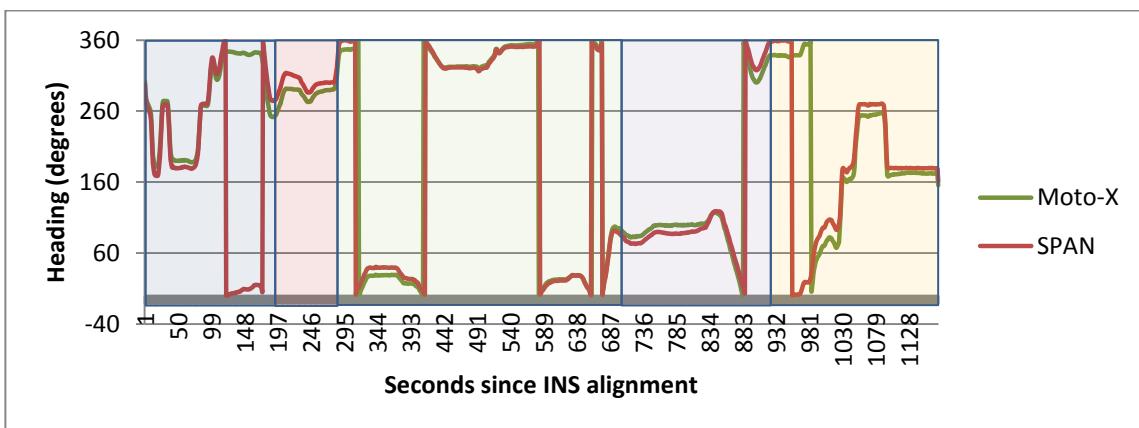


Figure 6-21: Heading comparison (open sky)

While Figure 6-21 shows the best solution that was attainable it is important to note the filter had to be run offline and that numerous attempts were required in order to best estimate the turn-on biases and scale factors. A slight change in any of these values and the entire solution completely diverges, producing a heading plot that looks nothing like the one above. Nonetheless the results are better than what would have been expected at the start of the thesis research. There are still a number of parameters that affect the

quality of the solution such as how the incoming GNSS updates were weighted, the low operating frequency of the IMU, mis-modelled biases, and kinematics that might have been removed when the data was smoothed via wavelet decomposition.

Table 6-6: Differences between Moto-X and SPAN in open sky

	Mean	Std. Dev	RMS
Latitude (m)	-1.612	1.529	2.224
Longitude (m)	0.009	1.460	1.460
Height (m)	-0.060	3.363	3.363
Velocity N (m/s)	0.004	0.406	0.407
Velocity E (m/s)	0.006	0.388	0.3885
Velocity U (m/s)	0.030	0.319	0.321
Roll (deg)	-2.833	2.967	4.102
Pitch (deg)	0.0437	2.506	2.506
Heading (deg)	6.316	11.608	13.215

Table 6-6 above shows the final metrics of the entire dataset. Position-wise, the height component is where the biggest RMS difference is seen, and the longitude component was considerably better than latitude. This is because of the -1.612 m bias in latitude which is skewed by the quality at the start of the dataset which is caused by the building in front of the car blocking some of the satellites in the northern direction. Velocity appears to have been quite good considering the assumptions on how the velocity updates were filtered into the INS filter. The mean velocities are very close to zero which is what was also seen in Chapter 5 when the filter was being assessed with the IMU-LN200. Attitude-wise, there is a bias in roll which is attributed to lower roll accuracy caused by the over/under passes encountered throughout the dataset. The pitch estimates were much closer to truth under these conditions. The biases in both roll and heading can also be caused by issues in the turn-on bias and scale factor errors. Heading

was actually quite good considering the quality of the sensors used but as was mentioned earlier it is pretty optimistic to expect this kind of behaviour in a filter that would run in real-time in the phone. The turn-on biases were modelled through the half-minute static calibration but the scale factor errors had to be found empirically afterwards. A slight variation in turn-on biases, scale factor errors, initial estimates, or process noise and the entire INS completely breaks down. Recommendations for improving the performance of the INS are described in Chapter 7.

Table 6-7 below shows the RMS of the differences for each specific section of the dataset. Position quality was very similar throughout (except for the latitude bias which skewed the start of the dataset). The RMS for the velocity differences was impacted in the first and fourth sections of the dataset by multipath from surrounding objects. Also, there is a decrease in along-track velocity accuracy that is evident in section 3 (predominantly northwards facing) and section 4 (predominantly eastwards direction of travel). Heading-wise, section three had the best heading estimation, which can be attributed to the high accuracy ZUPTs that were applied when the vehicle was stopped at a traffic light. Since the GNSS receiver does report the horizontal velocity, it is easy to know when a ZUPT should be applied.

Table 6-7: RMS of differences between Moto-X and SPAN in open sky

	Section 1 epochs: [1 -186]	Section 2 epochs: [187 -289]	Section 3 epochs: [290 -693]	Section 4 epochs: [694 -922]	Section 5 epochs: [923 -1172]
Latitude (m)	3.435	1.533	1.901	1.968	2.030
Longitude (m)	1.013	1.581	1.04	1.917	1.759
Height (m)	3.735	3.328	3.012	2.401	4.278

Velocity N (m/s)	0.567	0.161	0.395	0.248	0.467
Velocity E (m/s)	0.511	0.181	0.322	0.346	0.475
Velocity U (m/s)	0.184	0.459	0.319	0.416	0.219
Roll (deg)	6.455	3.977	3.480	3.461	3.320
Pitch (deg)	1.728	1.037	2.580	1.824	3.611
Heading (deg)	17.569	16.813	5.909	10.832	17.622

6.5 Urban Canyon

The last dataset to assess is the one that took place in downtown Calgary. The open-sky test delivered better than expected results but it took quite a bit of tinkering with the scale factor errors, and it had to be run offline. This dataset is certainly going to prove much more difficult. Compared to when the loosely-coupled filter was run with high-end IMU-LN200+FLEX6 data, the GNSS receiver inside the Moto-X smartphone should be able to provide more positions in this dataset due to the fact it is a high sensitivity receiver as explained in Chapter 2. However, the positions fed into the filter will be of very low accuracy due to the receiver being able to only track single frequencies, lower antenna quality, and lack of multipath mitigation firmware techniques found in the high precision FLEX6. These lower quality positions will greatly impact the filter. At least with the NovAtel raw data used in the truth test in Chapter 5, there are different flags indicating the quality of the GNSS data. Some examples are INSUFFICIENT_OBS,

NO_CONVERGENCE, SINGULARITY, INVALID_FIX, SOLN_COMPUTED, etc. If GNSS is very bad or non-existent, dead-reckoning could be used using the IMU-LN200 raw data. The difference is that now we are using very-low-quality sensors that will drift by a lot when an outage is encountered. The drift is going to be much worse than if the INS carries on using the low-accuracy positions and velocities from the high-sensitivity receiver. The figure below shows the location of the urban canyon dataset in downtown Calgary. The dataset coverage is directed primarily west-east.

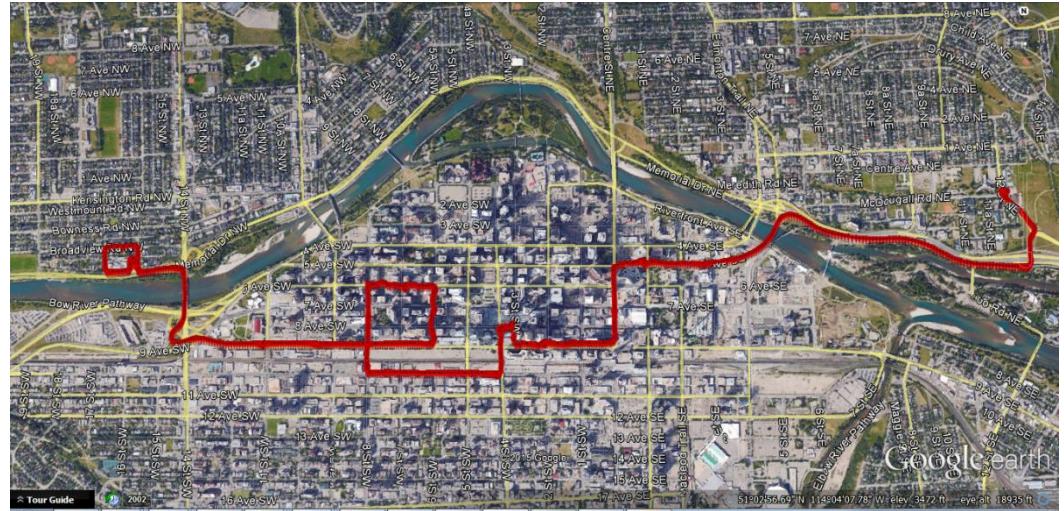


Figure 6-22: Urban canyon dataset overview with Moto-X sensors (Google, 2015)

As has been customary throughout the thesis, the dataset has been divided into smaller sections for ease of analysis. This time, the data will be divided into four sections.



Figure 6-23: First section of urban canyon dataset (Google, 2015)

The first section as shown above in Figure 6-23 starts with a very benign environment. The vehicle starts just north of the Bow River, travels south and merges onto 9th Ave. SW Eastbound. Speeds topped at 55 km/hr with the only stop taking place at the traffic light of the intersection with 11th St. SW.



Figure 6-24: Section two of urban canyon dataset (Google, 2015)

The second section as seen in Figure 6-24 goes through the heart of downtown, with large buildings all around a three block loop. The southernmost part of this section is a bit farther away from the tall buildings but towards the end of the section, as the vehicle heads north, it goes through a couple of underpasses. Speed was limited at a maximum of 50 km/hr and there were a few stops due to traffic lights. The following plots, Figure 6-25 and Figure 6-26, show the three block loop where most of the high-rises are encountered. The SPAN solution has been overlaid in green whereas the Moto-X loosely-coupled solution is shown in red. As before, the red Xs are the areas in which the SPAN solution did not have a GNSS update but rather used the IMU for dead-reckoning. The area is shown from a couple of perspectives to better identify the areas where position separation was the largest.

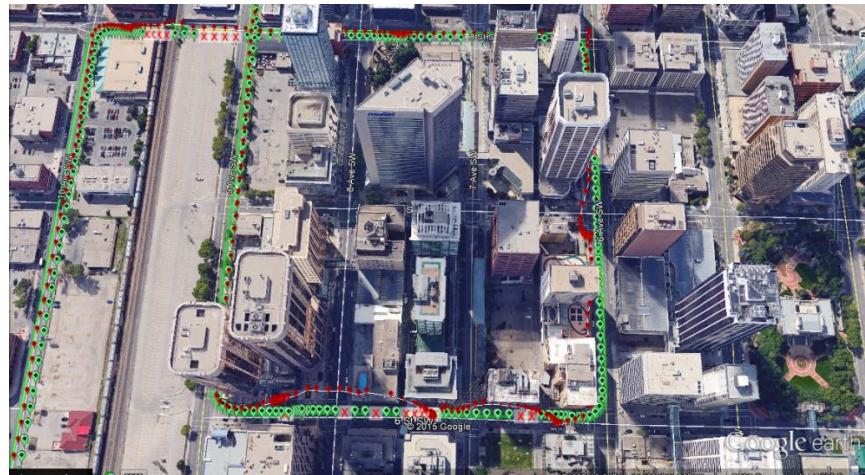


Figure 6-25: Section two looking west (Google, 2015)

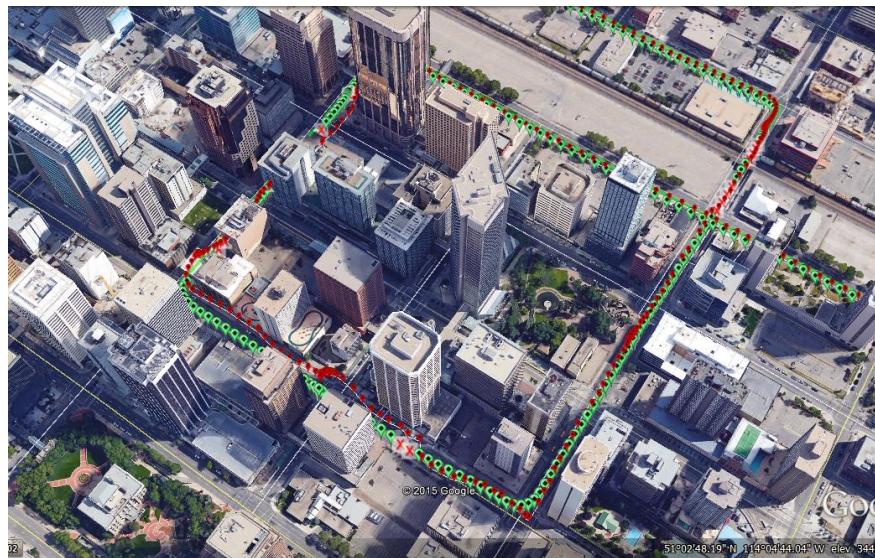


Figure 6-26: Section two looking southeast (Google, 2015)



Figure 6-27: Section three of urban canyon dataset (Google, 2015)

The third section shown above in Figure 6-27, starts with the vehicle heading northwards on 4th St. SW before turning right on 8th Ave. SW on its way back eastwards on 9th Ave. SW. The vehicle goes through an area with quite few large structures on both northern and southern ends of 9th Ave. SW (including the Calgary Tower) before

heading northwards. The start of this section, shown in Figure 6-28 and Figure 6-29 below visualize the effect of the high multipath environment on the position determinations.

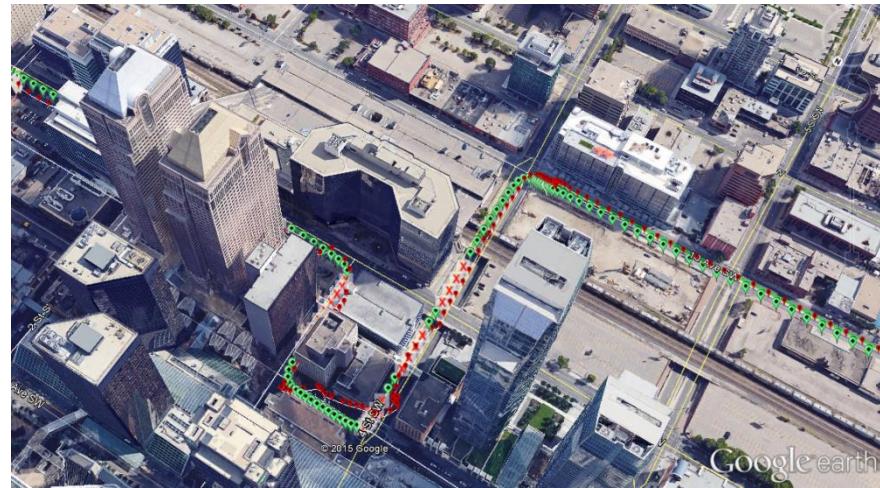


Figure 6-28: Start of section three looking southeast (Google, 2015)

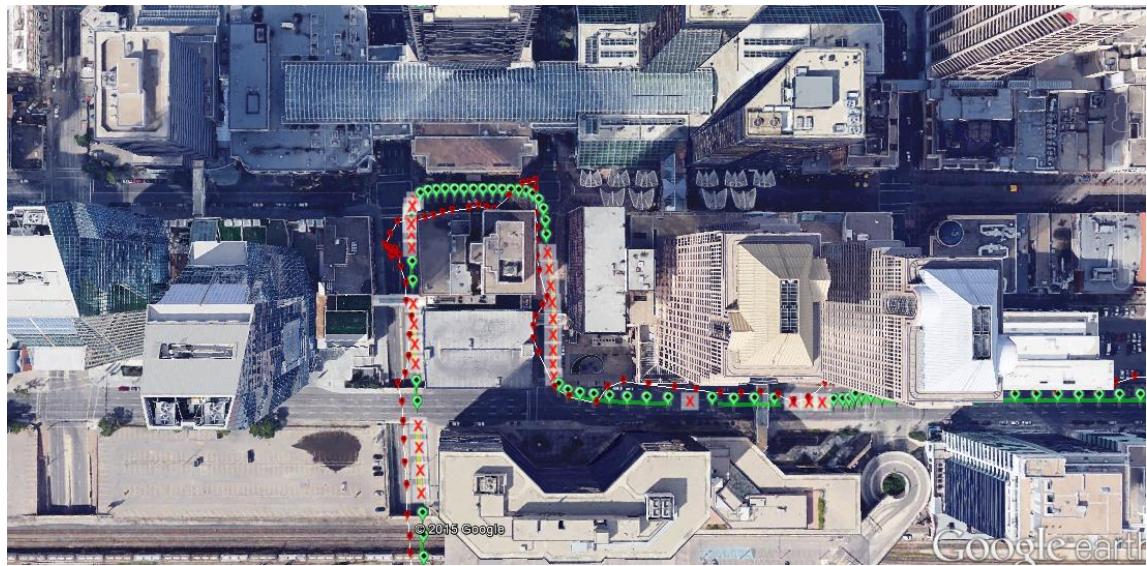


Figure 6-29: Plan view of start of section three (Google, 2015)

The next part of section three continues to suffer quite a bit from the high multipath environment with the vehicle trajectory offset enough to be off the road and going through buildings. This is shown below in Figure 6-30.

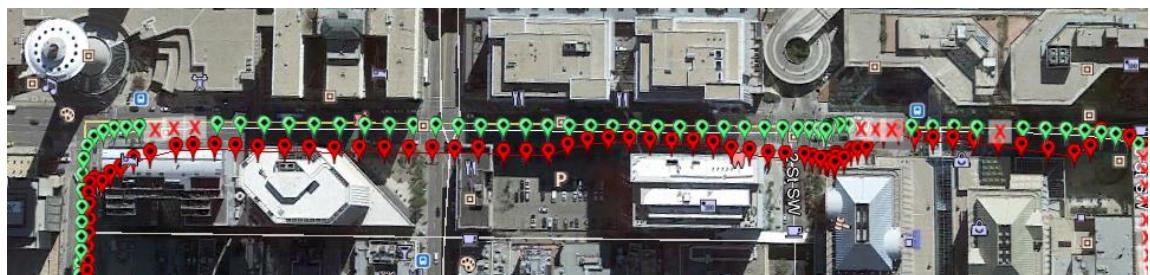


Figure 6-30: Last part of section three looking westward (Google, 2015)

The last part of this section has the vehicle going northbound on Centre St. SW before turning eastward on 5th Ave. SW. This intersection is important as it is where the tallest building in Calgary, The Bow, is located. The Bow is a 237 m, 58-story building with a glass exterior. The effect of The Bow on the position is shown below in Figure 6-31.

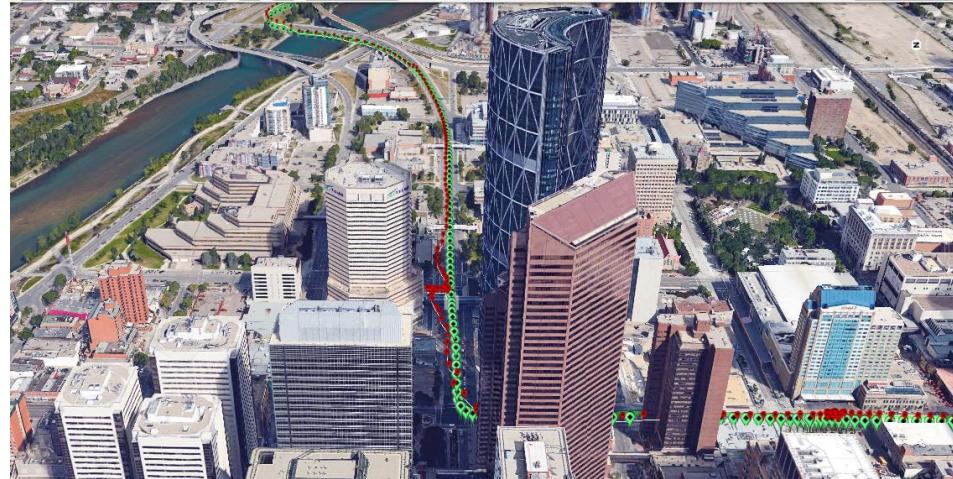


Figure 6-31: Effect of the Bow (Google, 2015)

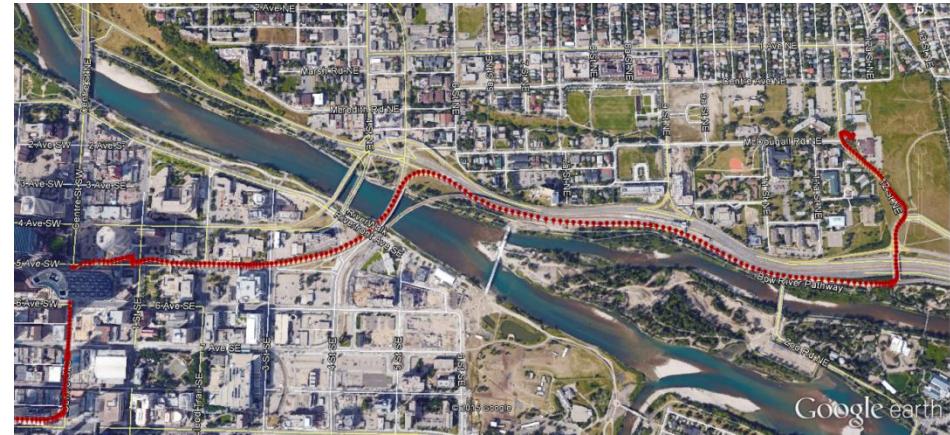


Figure 6-32: Section four of urban canyon dataset (Google, 2015)

The last section of this dataset is shown above in Figure 6-32. It starts on the corner of 5th Ave. SW and 1st St. SE, just a block eastwards from The Bow. This last section contains an ideal open-sky environment.

Table 6-8: Start, end epochs and cumulative time for urban canyon test

Section	Start epoch	End epoch	Time	Cumulative time
1	1	259	4' 18''	4'18''
2	260	674	6'54''	11'12''
3	675	953	4'38''	15'50''
4	954	1169	3'35''	19'25''

Although it was previously known this dataset would likely be too difficult for the filter using the Moto-X sensors, the interesting thing is to see whether the filter is able to converge once the urban canyon has been left behind. Let us now look at the separation plots.

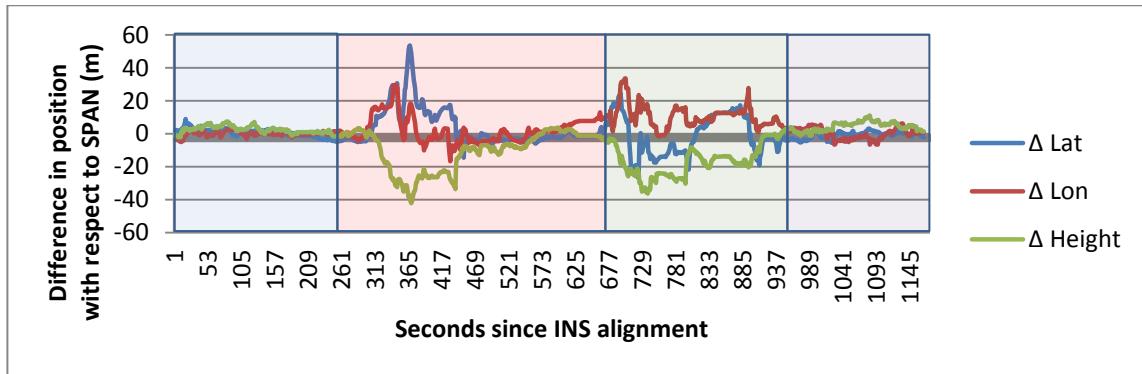


Figure 6-33: Difference in position with respect to SPAN (urban canyon)

Figure 6-33 shows the position differences in the first and fourth sections of the urban-canyon dataset were about the same as for the open-sky dataset. The middle

sections, which contained quite a challenging environment, have position differences of up to 60 m in the latitude. The longitude performed slightly better in section two but still had a couple of peaks of 30 m and 18 m difference. Both latitude and longitude were equally bad in the third section whereas height was equally bad in both middle sections. As has been stated previously, the quality of the computed position is as good as what is fed into the filter. As expected, the GNSS position updates fed into the INS were quite bad for these middle sections as shown below in Figure 6-34.

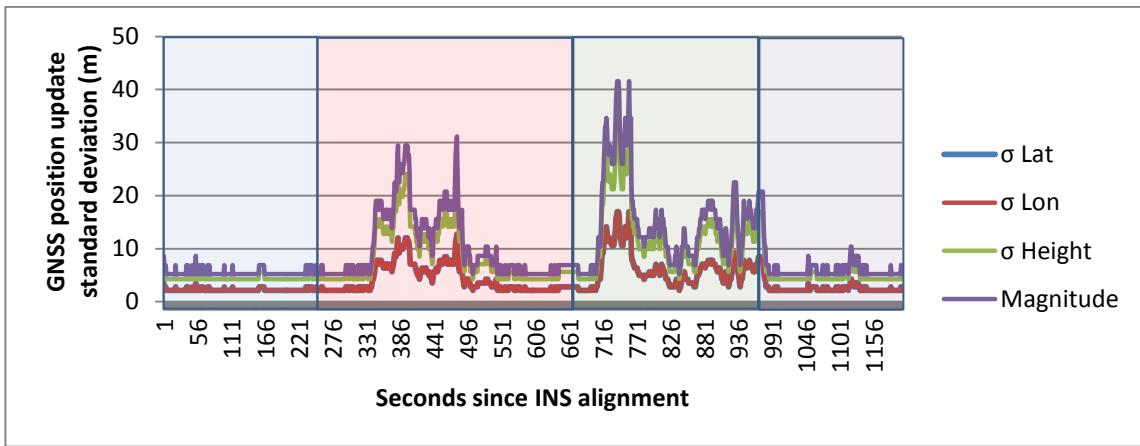


Figure 6-34: GNSS position update standard deviation (urban canyon)

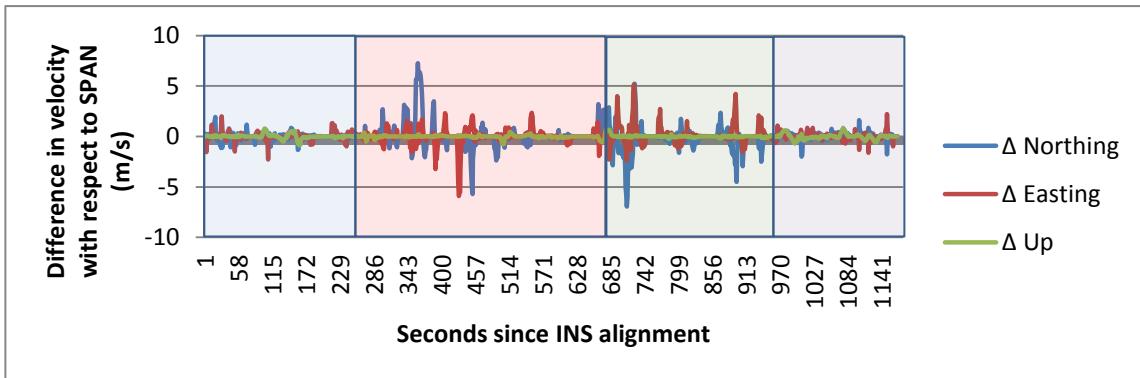


Figure 6-35: Difference in velocity with respect to SPAN (urban canyon)

Figure 6-35 above, shows the velocity differences compared to truth. The fact the up component difference was minimum shows the vehicle motion constraints worked well. The velocity differences in the open-sky dataset showed there were some spikes, which correlated to times at which the GNSS velocity updates were troublesome. It isn't any different in this case except we are now looking at much more pronounced peaks. Figure 6-36 below shows the decrease in velocity accuracy during the second and third sections.

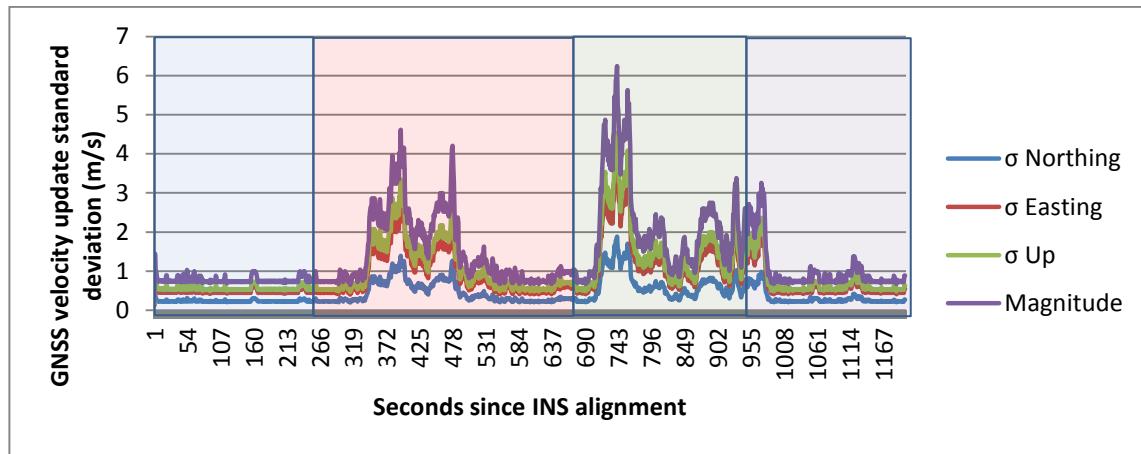


Figure 6-36: GNSS velocity update standard deviation (urban canyon)

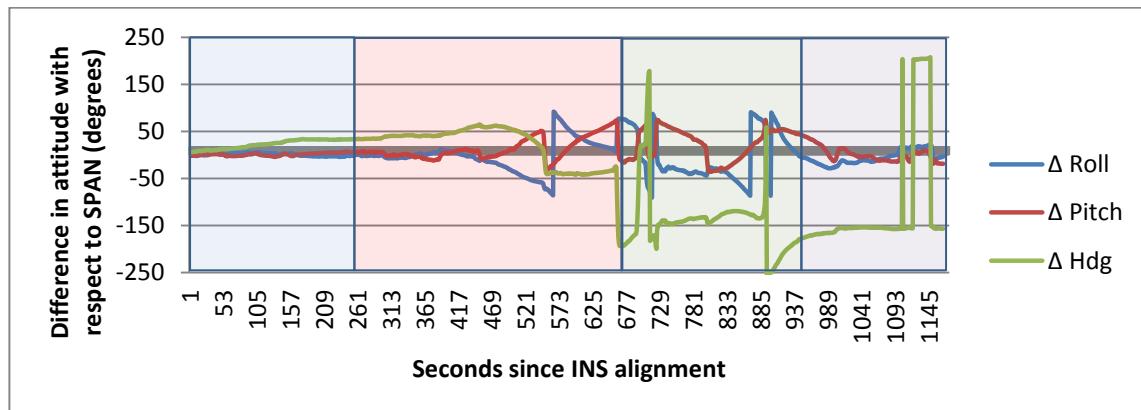


Figure 6-37: Difference in attitude with respect to SPAN (urban canyon)

This last plot above, Figure 6-37, shows the pitch and roll were fairly stable for the first section and the first half of the second section until the urban canyon environment was too much for the filter to handle causing it to blow out. The heading estimate didn't hold on as well, which indicates the turn-on biases and scale factor errors are not completely accurate. Nonetheless, the effect of the troublesome environment is the same on the heading component. Once this part of the dataset is reached, attitude accuracy completely collapses. An interesting fact is that pitch and roll start converging once the urban canyon is left behind in section four. The heading difference plateaus for a little bit until a sharp turn left turn is taken by the vehicle. The heading goes out in the complete opposite direction. Thus, while pitch and roll improve outside the urban canyon, heading did not.

Table 6-9 below shows the RMS of the differences for the different sections. As expected, position and velocity are least accurate in the middle sections when the vehicle is traversing through the heart of downtown Calgary. Pitch and roll were quite good in the first section, which did not contain a challenging environment. The heading estimate on the other hand was not as good as it was in the open-sky dataset. All three attitude components decrease considerably in quality within the middle sections with the last section seeing an improvement in pitch and roll once the vehicle has left downtown. However, the RMS for the differences in heading shows it was never able to recover from the urban canyon. This is expected as there isn't any direct or indirect heading measurement taking place. Once it diverges in downtown, there is nothing to help steer it back into place. Having a heading aiding sensor such as a secondary GNSS receiver/antenna pair would have definitely helped keeping the heading estimates in

check, at least once the vehicle left downtown. The reason why this wasn't an issue with pitch and roll estimates is that both are indirectly kept in check by gravity.

Table 6-9: RMS of differences between Moto-X and SPAN in urban canyon

	Section 1 epochs: [1 -259]	Section 2 epochs: [260 -674]	Section 3 epochs: 675 -953]	Section 4 epochs: [954 -1169]
Latitude (m)	2.593	11.983	12.192	2.284
Longitude (m)	1.502	8.042	12.595	3.339
Height (m)	3.137	15.635	19.037	5.062
Velocity N (m/s)	0.351	1.326	1.298	0.357
Velocity E (m/s)	0.398	0.809	0.989	0.344
Velocity U (m/s)	0.207	0.136	0.128	0.256
Roll (deg)	3.322	34.389	51.836	15.730
Pitch (deg)	2.993	25.573	40.412	14.560
Heading (deg)	24.811	52.360	155.966	165.717

CHAPTER 7: CONCLUSIONS AND RECOMMENDATIONS

The work presented in this thesis dealt with a thorough assessment of smartphone-level IMUs in a loosely-coupled INS. The motivation for the work stems from the fact smartphones have become quite normal nowadays, providing a large percentage of the population with a potentially compact INS that fits in a pocket. The primary equipment used was the Moto-X Android smartphone, which is equipped with MEMS accelerometers, gyroscopes, magnetometers and a high-sensitivity single frequency GNSS receiver capable of tracking both GPS and GLONASS. The magnetometers were assessed as to whether their heading estimate was accurate enough to be used for heading aiding in the filter. The other equipment that was used for truth trajectory purposes was the NovAtel Flex6 loaded with SPAN firmware, a GPS-702-GG survey-grade antenna, and a high-end IMU-LN200 motion sensor. The initial purpose of the thesis was to develop the INS filter inside the Android phone so that it would be completely stand-alone. However, as the difficulty of developing the loosely-coupled INS filter became apparent, the goal of the thesis ended up being geared towards developing the INS filter in C++ where the smartphone-logged raw data could be processed offline. There is nothing applied in this offline filter that cannot be applied in a real-time Android implementation provided there is an accurate a priori knowledge of turn-on biases and scale factor errors for the accelerometers and gyroscopes.

The author-generated loosely-coupled filter was first assessed by using the raw data from the IMU-LN200 and the single-point pseudorange measurements from the NovAtel position filter inside the Flexpak6. The truth testing was performed using an

open-sky dataset in NW Calgary and an urban canyon dataset in downtown Calgary. The loosely-coupled filter continually estimates 15 INS error states and uses GNSS position and velocity as updates. The filter contains vehicle motion constraints to control the estimated vehicle trajectory from accelerating upwards or moving sideways when a GNSS outage is encountered. It also applies zero velocity updates (ZUPTs) when there is a GNSS outage if it knows the vehicle is static (from looking at the accelerometer raw data). ZUPTs constrain the velocity error growth making them useful when outages occur. Once the filter was verified, an open-sky and urban canyon dataset was obtained with GNSS and raw IMU sensor data from the Moto-X smartphone, with the NovAtel SPAN solution being used as a truth for performance validation. The following are the conclusions from the work done and documented herein.

7.1 Conclusions

1. Tightly-coupled INS filters such as NovAtel SPAN perform better than loosely-coupled INS filters when in challenging conditions such as urban canyons. The reason for this is that a tightly-coupled integration can make use of a complete filter update with as few as two satellites whereas a loosely-coupled integration requires a minimum of 4 satellites.
2. INS navigation consists of two steps: the mechanization equation and an extended Kalman filter. The mechanization equations integrate the raw IMU data in order to get position, velocity, and attitude estimates. The EKF uses

knowledge of the sensor errors and current filter inputs in order to steer the mechanization equations. Inaccurate modelling of system process noise will cause the INS to diverge.

3. A 15 error state EKF was used in this project where 9 of the states correspond to three position, velocity, and attitude error estimates. The remaining 6 states correspond to error states for sensor bias drifts in the accelerometers and gyroscopes. The added error sources associated with the Moto-X smartphone sensors (e.g. axes misalignment, turn-on bias, scale factor errors) can be considered by adding states to the filter or by increasing the modelled error for the sensor bias drifts.
4. The author-generated loosely-coupled filter performed within expectations when using the NovAtel pseudorange measurements as updates to the filter running raw IMU-LN200 data in both open-sky and urban canyon datasets. The latter required use of IMU dead reckoning when a GNSS outage was encountered. Vehicle motion constraints were also added to minimize vertical and lateral movement. These constraints can only be applied to land vehicles.
5. The removal of scale factor errors for high-end IMUs such as the IMU-LN200 did not modify the output of the filter. On the other hand, the scale factor errors in the Moto-X sensors are required in order to achieve a

relatively accurate INS solution. The scale factor errors proved to be very sensitive; modifying them ever-so-slightly breaks the filter altogether.

6. Turn-on biases can be accurately determined for high-end IMUs such as the IMU-LN200 through a six-position calibration. These turn-on biases proved to be very repeatable from turn-on to turn-on. On the other hand, the turn-on biases on the Moto-X smartphone sensors change with every system reset meaning their biases have to be estimated with a 30-second in-field calibration.
7. The Hall effect magnetometers inside the Moto-X smartphone proved to be inaccurate even after calibrating them for soft-iron and hard-iron effects. The scale factor errors were estimated empirically but even then, results show the magnetometers are not as sensitive as they need to be. Previous research has shown accurate results with static heading estimates with AMR magnetometers. Hall effect magnetometers are more suited to be used as switches due to comparatively low sensitivity and stability when compared to other types of MEMS magnetometers.
8. The loosely-coupled INS using Moto-X raw data performed better than expected in open-sky conditions. The RMS difference of the heading was 13.215 degrees which means the Moto-X can be used for civilian enthusiast-level navigation. The big issue is in removing turn-on biases and scale factor errors accurately.

9. As technology improves, it is expected for INS sensors inside smartphones to get better making them more suitable for INS navigation. If Android devices logged raw GNSS measurements, it would even be possible to do a tightly-coupled GNSS/INS integration in the smartphone.
10. The loosely-coupled INS using Moto-X raw data performed as expected in the urban canyon scenario. The high-sensitivity GNSS receiver in the smartphone will provide position and velocity estimates in tougher environments when compared to a high-accuracy GNSS receiver (e.g. NovAtel FlexPak6), but these estimates won't be accurate enough for high-grade navigation. The low-accuracy updates cause the attitude to completely blow out in the urban canyon. While pitch and roll recover somewhat after the vehicle leaves the urban canyon, heading is not able to recover. The reason for this is that pitch and roll are indirectly kept in check through gravity and accurate GNSS position/velocity updates whereas heading is directly dependent on vehicle motion and error estimates in the filter.
11. Three assumptions related to accuracy of input filter updates were made when running the INS with the raw data from the Moto-X smartphone. The first is that the vertical component of position is two times less accurate than the horizontal accuracy. This assumption is partly to blame for the poor vertical position in the Moto-X. The second assumption is that both easting and northing components have the same accuracy. The third assumption derived accuracies for the input velocity updates from changes in position.

This assumption might work when the vehicle travels at fast speeds but is not applicable when the vehicle is stationary or moving slowly.

7.2 Recommendations

Based on the results and conclusions presented in this thesis, the following is recommended for future research:

1. Implement the author-generated loosely-coupled filter in an Android device as was originally intended in this thesis research. Rewriting code that works is easier than writing it from scratch, especially in Android which is not the most user-friendly operating system for coding. As technology improves, so will the sensors inside smartphones. Having a released loosely-coupled INS filter freely available to everyone is a great way to provide tested inertial navigation to smartphone application developers so they can spend more time on the more creative side of their application.
2. Using the loosely-coupled filter as a starting point, develop a tightly-coupled integration using smartphone sensors to assess the level of improvement for urban canyon scenarios. If smartphones are still unable to log raw measurements when the project is undertaken, any other GNSS receiver capable of doing so can be used instead.

3. Add an in-field turn-on bias calibration to the Android application following the procedure used in this thesis.
4. Develop a method to estimate scale factor errors while in the field in real time, similar to that of the turn-on bias calibration.
5. Come up with a more accurate way of determining the accuracy of velocity updates. Current estimates are derived from the accuracy in the changes in position; this is inaccurate when moving at slow velocities.
6. Implement and assess a 27-error-state filter where the scale factor errors and turn-on biases are directly modelled in the INS. Further states could be added to more accurately estimate the misalignment of the axes.
7. The current filter implementation assumes the IMU enclosure frame matches that of the vehicle frame. The application needs to be modified so that the necessary rotations are applied so that the IMU can be mounted any way in the vehicle.
8. Allow filter to use aided heading from external sensors in order to allow instantaneous alignments and constrain heading drifts. Filter should be smart enough to determine if sensors are accurate enough to be used or not through vehicle calibration by assessing the difference of the external sensor's heading with that of the vehicle's course made good. Users should be required to move at a pre-determined speed given that course made good will be more accurate at higher speeds.

9. Current initial alignment is hard coded in the filter from knowledge of the truth system (SPAN). The filter should default to align kinematically and allow an aided heading alignment if point eight above is implemented.

REFERENCES

- Bahan, K. (2014). Personal Communication. Geomatics Technical Lead, NovAtel Inc, Calgary, AB, Canada, July.
- Bektas, S. (2015). “Least Squares Fitting of Ellipsoid Using Orthogonal Distances”. *Boletim de Ciências Geodésicas*, Vol 21, No 2, Curitiba, Brazil.
- Bilich, A. and Larson, K. (2009). “It’s not All Bad: Understanding and Using GNSS Multipath”. *GPS World*, October 2009.
- Blum, R., Greencorn, D., and Cooperstock J. (2012). “Smartphone sensor reliability for augmented reality applications”. McGill University, Montreal, QC, Canada.
<http://srl.mcgill.ca/publications/2012-MOBIQUITOUS.pdf>. Last accessed on August 10th, 2016.
- Bobye, M. (2014). Personal Communication. Principal Geomatics Engineer, NovAtel Inc, Calgary, AB, Canada, December.
- Braun, A. (2006). “ENGO421 Coordinate Transformations”. Department of Geomatics Engineering lecture notes, University of Calgary, Calgary, AB, Canada.
- Brodin, G., and Daly, P., (1997). “GNSS Code and Carrier Tracking in Presence of Multipath”. *International Journal of Satellite Communication*, Vol 15, pp-25-34.
- Brown, R. G., and Hwang, P. (1997) *Introduction to Random Signals and Applied Kalman Filtering*. John Wiley & Sons, University of Michigan, Ann Arbor.
- Cai, Y., Zhao, Y., Ding, X., and Fennelly, J. (2012). “Magnetometer Basics for Mobile Phone Applications”. Electronic Products.
- Dixon, R. (2014). Personal Communication. NovAtel SPAN Chief Engineer, NovAtel Inc, Calgary, AB, Canada, March – September.
- Driscoll, C., Lachapelle, G., and Tamazin, M. (2011). “Dynamic Duo. Combined GPS/GLONASS Receivers in Urban Environments”. *GPS World*, January 2011.
- El-Sheimy, N., Osman A., Nassar, S., and Noureldin, A. (2003). “A New Way to Integrate GPS and INS Wavelet Multiresolution Analysis”. *GPS World*, October 2003.
- El-Sheimy, N. (2008). “ENGO361 Least Squares Adjustment”. Department of Geomatics Engineering lecture notes, University of Calgary, Calgary, AB, Canada.

- El-Sheimy,N. (2012). “Inertial Techniques and INS/DGPS Integration”. Department of Geomatics Engineering lecture notes, University of Calgary, Calgary, AB, Canada.
- Farrell, J. A. and M. Barth (2001) *The Global Positioning System & Inertial Navigation*, McGraw Hills, New York.
- Gao, J. (2007). *Development of a Precise GPS/INS/On-Board Vehicle Sensors Integrated Vehicular Positioning System*. Ph.D. thesis, Department of Geomatics Engineering, University of Calgary, Calgary, AB, Canada.
- Gao, Y. and Sideris, M. (2007). “Data Analysis in Engineering”. Department of Geomatics Engineering lecture notes, University of Calgary, Calgary, AB, Canada.
- Gelb, A. (1974) *Applied Optimal Estimation*. The Massachusetts Institute of Technology Press, Cambridge.
- Godha, S. (2006). *Performance Evaluation of Low Cost MEMS-Based IMU Integrated with GPS for Land Vehicle Navigation Application*. M.Sc.E. thesis, Department of Geomatics Engineering, University of Calgary, Calgary, AB, Canada.
- Godha, S. (2014). Personal Communication. Sr. Team Lead, Positioning Algorithms, NovAtel Inc, Calgary, AB, Canada, June.
- Google Inc. Google Earth. 2015.
- GPS World (2016). “Google to provide raw GNSS measurements”. <http://gpsworld.com/google-to-provide-raw-gnss-measurements/>. Last accessed on August 14th, 2016.
- Grewal, M. and Andrews, A. (2001) *Kalman Filtering: Theory and Practice Using Matlab*, John Wiley and Sons Inc., Hoboken.
- Guo, H.H., Xiong, J., Yu, M., Fang, S., and Guo, J. (2015). “Low-cost Sensor Information Fusion Technology for iPhone Aided Navigation”. *Proceedings of the 2015 Internatioal Technical Meeting*, ION ITM 2015, Dana Point, California, January 26-28, 2015.
- Hadrel, T., Phocas, M., and Riquier, N. (2010). “Mobile-Phone GPS Antennas”. *GPS World*, February 2010.
- iFixit (n.d.). “Motorola Moto X Teardown”. <https://www.ifixit.com/teardown/motorola+moto+x+teardown/16867>. Last accessed on August 9th, 2016.
- Kay, H. (2013). “Smartphone IMU GPS”, Downloaded from Google Play Store.

- Klug, B. (2013). "The State of Qualcomm's Modems – WTR1605 and MDM9x25". <http://www.anandtech.com/show/6541/the-state-of-qualcomms-modems-wtr1605-and-mdm9x25/2>. Last Accessed on August 9th, 2016.
- Kong, X. (2000). *Inertial Navigation System Algorithms for Low Cost IMU*. Ph.D. thesis, Australian Centre for Field Robotics, Department of Mechanical and Mechatronic Engineering, The University of Sydney, Sydney, NSW, Australia.
- Langley, R.B. (1997). "GPS Receiver System Noise". *GPS World*, pp 40-45.
- Langley, R.B. (2003). "Getting Your Bearings: The Magnetic Compass and GPS". *GPS World*. <http://www2.unb.ca/gge/Resources/gpsworld.september03.pdf>. Last accessed on April 29th, 2016.
- Langley, R.B. (2016). Personal Communication. Professor at Geodesy and Geomatics Engineering, University of New Brunswick, Fredericton, NB, Canada, January – August.
- Maron, K. (2014). "Introducing NovAtel Correct". NovAtel Inc, Calgary, AB, Canada.
- MacGougan, G. (2003). *High Sensitivity GPS Performance in Degraded Signal Environments*. M.Sc.E. thesis, Department of Geomatics Engineering, University of Calgary, Calgary, AB, Canada.
- Misra, P. and Enge, P. (2011). *Globas Positioning System: Signals Measurements, and Performance*. 2nd Ed. Ganga-Jamuna press, Lincoln, Massachusetts.
- Mohamed, A. and Mamatas, A. (2012). "Fundamentals of GNSS-Aided Inertial Navigation". Automatic Flight Control Systems – Latest Developments, Dr. Thomas Lombaerts (Ed.). University of Florida. ISBN: 978-953-307-816-8, 35 pp.
- Mourcou, Q., Fluery, A., Franco, C., Klopcic, F., and Vuillerme, N. (2015). "Performance Evaluation of Smartphone Inertial Sensors Measurements for Range of Motion". *Open Access Sensors Journal*. <http://www.mdpi.com/1424-8220/15/9/23168/pdf>. Last accessed on August 10th, 2016.
- Nassar, S. (2003). *Improving the Inertial Navigation System (INS) Error Model for INS and INS/DGPS Applications*. Ph.D. thesis, Department of Geomatics Engineering, University of Calgary, Calgary, AB, Canada.
- Nayak, R.A. (2000). *Reliable and Continuous Urban Navigation using Multiple GPS Antennas and a Low Cost IMU*. . M.Sc.E. thesis, Department of Geomatics Engineering, University of Calgary, Calgary, AB, Canada.

NovAtel Inc. (2015a). “ALIGN Heading and ALIGN Relative Positioning Application Note”. <http://www.novatel.com/assets/Documents/Bulletins/apn048.pdf>. Last accessed on January 12th, 2016.

NovAtel Inc. (2015b). “OEM6 Installation and OperationUser Manual”. <http://www.novatel.com/assets/Documents/Manuals/om-20000128.pdf>. Last accessed on January 12th, 2016.

NovAtel Inc. (2015c). “OEM6 Firmware Reference Manual”. <http://www.novatel.com/assets/Documents/Manuals/om-20000129.pdf>. Last accessed on January 12th, 2016.

NovAtel Inc. (2015d). “Positioning Modes of Operation”. <http://www.novatel.com/assets/Documents/Bulletins/apn051.pdf>. Last accessed on January 12th, 2016.

NovAtel Inc. (2015e). “SPAN Brochure”. <http://www.novatel.com/assets/Documents/Papers/SPANBrochure.pdf>. Last accessed on January 12th, 2016.

NovAtel Inc. (2015f). “SPAN on OEM6 Firmware Reference Manual”. <http://www.novatel.com/assets/Documents/Manuals/OM-20000144UM.pdf>. Last accessed on January 12th, 2016.

NovAtel Inc. (2015g). “SPAN on OEM6 User Manual”. <http://www.novatel.com/assets/Documents/Manuals/om-20000139.pdf>. Last accessed on January 12th, 2016.

NovAtel Inc. (2015h). “OEM628 Product Sheet”. <http://www.novatel.com/assets/Documents/Papers/OEM628.pdf>. Last accessed on January 12th, 2016.

O’Keefe, K. (2007). “ENGO465 Satellite Positioning”. Department of Geomatics Engineering, lecture notes, University of Calgary, Calgary, AB, Canada.

O’Keefe, K. (2008). “ENGO585 Wireless Positioning”. Department of Geomatics Engineering, lecture notes University of Calgary, Calgary, AB, Canada.

Olshausen, B. (2000). “Aliasing, PSC129- Sensory Processes, Lecture Notes, University of California, Berkeley. <http://redwood.berkeley.edu/bruno/npb261/aliasing.pdf>. Last accessed on August 11th, 2016.

Ozyagcilar, T. (2012a). “Implementing a Tilt-comensated eCompass using Accelerometer and Magnetometer Sensors”. Freescale Semiconductor Inc.

Ozyagcilar, T. (2012b). “Layout Recommendations for PCBs using a Magnetometer Sensor”. Freescale Semiconductor Inc.

- Ozyagcilar, T. (2013). "Calibrating an eCompass in the Presence of Hard and Soft-Iron Interference". Freescale Semiconductor Inc.
- Park, K., Nerem, R.S., Schenewerk, M.S., and Davis, L.J. (2004). "Site-specific multipath characteristics of global IGS and CORS GPS sites". *Journal of Geodesy*, 77, pp. 799-803.
- Petovello, M. (2003). *Real-Time Integration of a Tactical-Grade IMU and GPS for High-Accuracy Positioning and Navigation*. Ph.D. thesis, Department of Geomatics Engineering, University of Calgary, Calgary, AB, Canada.
- Renaudin, V., Afzal, and M., Lachapelle, G. (2010). "Complete Triaxis Magnetometer Calibration in the Magnetic Domain". *Journal of Sensors*, Volume 2010, Article ID 967325, 10 pp.
- Savage, P. G. (2000). *Strapdown Analytics*, vol. 1. Strapdown Associates, Inc., Maple Plain.
- Schwarz K.P., and Wei, M. (2000). "ENGO 623 INS/GPS Integration for Geodetic Applications". Department of Geomatics Engineering lecture notes, University of Calgary, Calgary, AB, Canada.
- Shin, E. (2001). *Accuracy Improvement of Low Cost INS/GPS for Land Applications*. M.Sc.E. thesis, Department of Geomatics Engineering, University of Calgary, Calgary, AB, Canada.
- Sukkarieh, S. (2000) *Low Cost, High Integrity, Aided Inertial Navigation Systems for Autonomous Land Vehicles*, PhD Thesis, Department of Mechanical and Mechatronic Engineering, University of Sydney, Australia
- Tang, H. (2014). *Autonomous Mobile Robot Indoor Navigation using Multi-sensor Integration*, PhD Thesis, Department of Geodesy and Geomatics Engineering, University of New Brunswick, Fredericton, N.B., Canada
- Weill, L.R. (1997). "Conquering Multipath: The GPS Accuracy Battle". *GPS World*, pp 59 -66.
- Wells, D., Beck, N., Delikaraouglo, D., Kelusberg, A., Krakiwsky, E., LaChapelle, G., Langley, R.B., Nakiboglu, M., Schwarz, K., Tranquilla, J., and Vanicek, P. (1999). "Guide to GPS Positioning". Department of Geodesy and Geomatics Engineering Technical Report No. 58, University of New Brunswick, Fredericton, N.B., Canada.
- Zhang, X. (2003). *Integration of GPS with a Medium Accuracy IMU for Metre-Level Positioning*, MSc Eng. Thesis, Department of Geomatics Engineering, University of Calgary, Canada.

Zhang, J., Li, B., Dempster, A., Rizos, C. (2011). “Evaluation of High Sensitivity GPS Receivers”. *Coordinates: A Monthly Magazine on Positioning, Navigation, and Beyond*, March 2011.

APPENDIX A: DYNAMICS AND MEASUREMENT MODELS FOR INS ERROR STATES IN THE E-FRAME

Dynamics model for 15-State Filter

$$\begin{aligned}
 \begin{pmatrix} \dot{\delta r^e} \\ \dot{\delta v^e} \\ \dot{\varepsilon^e} \\ \dot{\delta b_a} \\ \dot{\delta b_g} \end{pmatrix} &= \underbrace{\begin{pmatrix} 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ N_{3x3}^e & -2(\Omega_{ie}^e)_{3x3} & -F_{3x3}^e & (R_b^e)_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & -(\Omega_{ie}^e)_{3x3} & 0_{3x3} & (R_b^e)_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & -\beta_{a3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & -\beta_{g3x3} \end{pmatrix}}_F \begin{pmatrix} \delta r^e \\ \delta v^e \\ \varepsilon^e \\ \delta b_a \\ \delta b_g \end{pmatrix} \\
 + & \underbrace{\begin{pmatrix} 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ (R_b^e)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & (R_b^e)_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & I_{3x3} \end{pmatrix}}_G \begin{pmatrix} \eta_a \\ \eta_g \\ \eta_{ba} \\ \eta_{bg} \end{pmatrix}, \quad Q(t)
 \end{aligned}$$

The dot on top of the variable represents the rate of change, e corresponds to the e-frame, b corresponds to the b-frame, and i corresponds to the i-frame. The remaining symbols are:

δr is the position error state vector ($\delta r_x, \delta r_y, \delta r_z$)

δv is the velocity error state vector ($\delta v_x, \delta v_y, \delta v_z$)

ε is the misalignment error state vector ($\varepsilon_x, \varepsilon_y, \varepsilon_z$)

δb_a is the accelerometer bias drift error

δb_g is the gyroscope bias drift error

R_b^e is the rotation matrix from the b-frame to the e-frame

δf is the raw accelerometer measurement errors ($\delta f_x, \delta f_y, \delta f_z$)

δw is the raw gyroscope measurement errors ($\delta w_x, \delta w_y, \delta w_z$)

I is the identity matrix

β_a is the inverse time constant for accelerometer bias drift

β_g is the inverse time constant for gyroscope bias drift

F^e is the skew symmetric matrix of the specific force (f_x, f_y, f_z) in the e-frame. Its form is as follows:

$$F^e = \begin{pmatrix} 0 & -f_z & f_y \\ f_z & 0 & -f_x \\ -f_y & f_x & 0 \end{pmatrix}$$

Ω_{ie} is the skew symmetric matrix of the Earth's rotation rate from the i-frame to the e-frame, expressed in the e-frame. Its form is as follows:

$$\Omega_{ie}^e = \begin{pmatrix} 0 & -\omega_e & 0 \\ \omega_e & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

where

ω_e is the Earth's mean rotation rate of 15.041 °/h

N^e is the gravitational gradient coefficient matrix in the e-frame. Its form is as follows:

$$\mathbf{N}^e = \begin{pmatrix} \frac{km}{R^3} \left(\frac{3r_x^2}{R^2} - 1 \right) + \omega_e^2 & \frac{km}{R^3} \frac{3r_x r_y}{R^2} & \frac{km}{R^3} \frac{3r_x r_z}{R^2} \\ \frac{km}{R^3} \frac{3r_x r_y}{R^2} & \frac{km}{R^3} \left(\frac{3r_y^2}{R^2} - 1 \right) + \omega_e^2 & \frac{km}{R^3} \frac{3r_y r_z}{R^2} \\ \frac{km}{R^3} \frac{3r_x r_z}{R^2} & \frac{km}{R^3} \frac{3r_y r_z}{R^2} & \frac{km}{R^3} \left(\frac{3r_z^2}{R^2} - 1 \right) \end{pmatrix}$$

where

k is the gravitational constant, $6.674 \times 10^{-11} \frac{m^3}{kg s^2}$

M is the mass of the Earth, $5.972 \times 10^{24} kg$

R is the radius of the Earth, $\sqrt{r_x^2 + r_y^2 + r_z^2} m$

ω_e is the rotation rate of the Earth

η_a is the accelerometer sensor noise with a spectral density of q_a

η_b is the gyroscope sensor noise with a spectral density of q_b

η_{ba} is the accelerometer bias drift noise with a spectral density of q_{ba}

η_{bg} is the gyroscope bias drift noise with a spectral density of q_{bg}

$\mathbf{Q}(t)$ is the spectral noise density matrix

$$\mathbf{Q}(t) = \begin{pmatrix} (q_a)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & (q_b)_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & (q_{ba})_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & (q_{bg})_{3x3} \end{pmatrix}$$

Dynamics model for 27-State Filter

$$\begin{aligned}
& \left(\begin{array}{c} \delta \dot{r}^e \\ \delta \dot{v}^e \\ \dot{\varepsilon}^e \\ \delta \dot{b}_a \\ \delta \dot{b}_g \\ b_{a,turn-on} \\ b_{g,turn-on} \\ \dot{S}_a \\ \dot{S}_b \end{array} \right) \\
& = \underbrace{\begin{pmatrix} 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ N_{3x3}^e & -2(\Omega_{ie}^e)_{3x3} & -F_{3x3}^e & (R_b^e)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & -(\Omega_{ie}^e)_{3x3} & 0_{3x3} & (R_b^e)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & -\beta_{a,3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & -\beta_{g,3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} \end{pmatrix}}_F \left(\begin{array}{c} \delta r^e \\ \delta v^e \\ \varepsilon^e \\ \delta b_a \\ \delta b_g \\ b_{a,turn-on} \\ b_{g,turn-on} \\ S_a \\ S_b \end{array} \right) \\
& + \underbrace{\begin{pmatrix} 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ (R_b^e)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & (R_b^e)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & I_{3x3} \end{pmatrix}}_G \left(\begin{array}{c} \eta_a \\ \eta_g \\ \eta_{ba} \\ \eta_{bg} \\ \eta_{sa} \\ \eta_{sg} \\ w \end{array} \right), \quad Q(t) \\
Q(t) & = \begin{pmatrix} (q_a)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & (q_b)_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & (q_{ba})_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & (q_{bg})_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & (q_{sa})_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & (q_{sb})_{3x3} \end{pmatrix}
\end{aligned}$$

where the additional symbols with respect to the 15-state filter are:

$\mathbf{b}_{a,turn-on}$ is the accelerometer turn-on bias

$\mathbf{b}_{g,turn-on}$ is the gyroscope turn-on bias

S_a is the accelerometer scale factor error

S_g is the gyroscope scale factor error

β_{sa} is the inverse time constant for the accelerometer scale factor

β_{sg} is the inverse time constant for the gyroscope scale factor

η_{sa} is the accelerometer scale factor error sensor noise with a spectral density of q_{sa}

η_{sg} is the gyroscope scale factor error sensor noise with a spectral density of q_{sg}

Loosely Coupled Measurement Model

$$\underbrace{\begin{pmatrix} r_{INS}^e - r_{GNSS}^e \\ v_{INS}^e - v_{GNSS}^e \end{pmatrix}}_{\mathbf{z}} = \underbrace{\begin{pmatrix} I_{3x3} & 0_{3x3} & 0_{3x(n-6)} \\ 0_{3x3} & I_{3x3} & 0_{3x(n-6)} \end{pmatrix}}_{\mathbf{H}} \underbrace{\begin{pmatrix} \delta r \\ \delta v \end{pmatrix}}_{\mathbf{x}} + \underbrace{\begin{pmatrix} \eta_{r_{GNSS}} \\ \eta_{v_{GNSS}} \end{pmatrix}}_{\boldsymbol{\nu}}, R(t)$$

$$\mathbf{R}(t) = \begin{pmatrix} \sigma^2 r_{GNSS}^e & 0_{3x3} \\ 0_{3x3} & \sigma^2 v_{GNSS}^e \end{pmatrix}$$

where,

r_{INS}^e is the INS position prediction at current epoch

v_{INS}^e is the INS velocity prediction at current epoch

r_{GNSS}^e is the GNSS position observation at current epoch

v_{GNSS}^e is the GNSS velocity observation at current epoch

$\eta_{r_{GNSS}}$ is the GNSS position measurement noise with variance $\sigma^2 r_{GNSS}^e$

$\eta_{v_{GNSS}}$ is the GNSS velocity measurement noise with variance $\sigma^2 v_{GNSS}^e$

APPENDIX B: SMARTPHONE MAGNETOMETER CALIBRATION

Chapter 4 indicated it is theoretically possible to use magnetometers as external heading sources in INS. It was also mentioned the measurement model specific to aided heading is the same regardless of whether the aided heading source is from GNSS or magnetometers. As per Chapter 5, the magnetometers inside the Moto-X Android smartphone provided very bad estimates that proved to be unusable for INS navigation. Even so, for the sake of completeness, the calibration procedure applied is shown below.

A smartphone magnetometer measures the sum of geomagnetic field plus magnetic interference from ferromagnetic components present in the smartphone circuit board (Langley, 2003). It is suggested for the circuit board manufacturers to place the magnetometers far from sources of magnetic interference, as well minimize use of ferromagnetic components. However, even after minimizing this as much as possible in the design stage, there smartphone magnetometers will experience hard iron and soft iron interference (*ibid*). Hard iron interference is caused by permanently magnetized ferromagnetic components in the circuit board and will be present as a bias in the magnetometer reading (*ibid*). Soft iron interference is caused by the induced magnetic field acting on normally unmagnetized components in the circuit board (*ibid*). The smartphone magnetometer reading is shown below:

$$\mathbf{B}^b = \mathbf{W} \mathbf{R}_l^b \mathbf{B}^l + \mathbf{V} \quad \text{B.1}$$

where,

\mathbf{W} is the matrix representing the soft-iron coefficients

\mathbf{V} is the hard iron vector

\mathbf{B}^b is the raw magnetometer vector in the b-frame

\mathbf{B}^l is the raw magnetometer vector in the l-frame defined by

$$\mathbf{B}^l = \begin{pmatrix} 0 \\ B \cos \delta \\ B \sin \delta \end{pmatrix} \quad \text{B.2}$$

where B is the magnitude of the observed geomagnetic field vector and δ is the magnetic inclination. The horizontal component of the geomagnetic field always points to the magnetic north pole which is why the raw magnetometer measurements need to be rotated to the local frame using \mathbf{R}_l^b before being used (Langley, 2003; Ozyagcilar, 2013).

The calibration is performed in this thesis research was carried out by logging raw magnetometer data at different orientations over 10 minutes to ensure enough samples are taken. For the results presented in this thesis research, 10 minutes at ~ 40 Hz were observed thus providing around 24000 samples. An ellipsoid is then best fitted over the raw data. As per Langley (2003), in principle, only four measurements are required but redundancy is important to accomplish a more accurate calibration.

An ellipsoid can be expressed in general algebraic form as follows:

$$\mathbf{A}x^2 + \mathbf{B}y^2 + \mathbf{C}z^2 + 2\mathbf{D}xy + 2\mathbf{E}xz + 2\mathbf{F}yz + 2\mathbf{G}x + 2\mathbf{H}y + 2\mathbf{I}z = 1 \quad \text{B.3}$$

where the bold values represent the 9-coefficients that define the shape of the ellipsoid.

The $\mathbf{G}, \mathbf{H}, \mathbf{I}$ coefficients represent the coordinates of the ellipsoid centre in the x,y,z axes respectively.

We can formulate the fitting of the magnetometer raw data to an ellipsoid as:

$$\mathbf{H}\hat{\mathbf{x}} = \mathbf{l} \quad \text{B.4}$$

where,

\mathbf{H} is the design matrix of B.3 with respect to the unknown ellipsoidal coefficients

$\hat{\mathbf{x}}$ is the vector of unknown coefficients $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{H}, \mathbf{I})^T$

\mathbf{l} is a vector of ones

The least squares solution for x takes the following form:

$$\hat{\mathbf{x}} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \mathbf{l} \quad \text{B.5}$$

$\hat{\mathbf{x}}$ can be used to create the square symmetric matrix, $\tilde{\mathbf{A}}$, as follows

$$\tilde{\mathbf{A}} = \begin{pmatrix} \hat{x}_{(1,1)} & \hat{x}_{(4,1)} & \hat{x}_{(5,1)} & \hat{x}_{(7,1)} \\ \hat{x}_{(4,1)} & \hat{x}_{(2,1)} & \hat{x}_{(6,1)} & \hat{x}_{(8,1)} \\ \hat{x}_{(5,1)} & \hat{x}_{(6,1)} & \hat{x}_{(3,1)} & \hat{x}_{(9,1)} \\ \hat{x}_{(7,1)} & \hat{x}_{(8,1)} & \hat{x}_{(9,1)} & -1 \end{pmatrix} = \begin{pmatrix} A & D & E & G \\ D & B & F & H \\ E & F & C & I \\ G & H & I & -1 \end{pmatrix} \quad \text{B.6}$$

The ellipsoid centre, which represents the hard iron effect, V , can be computed from elements of $\tilde{\mathbf{A}}$:

$$\begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix} = \begin{pmatrix} A & D & E \\ D & B & F \\ E & F & C \end{pmatrix}^{-1} \begin{pmatrix} G \\ H \\ I \end{pmatrix} \quad \text{B.7}$$

Applying the hard-iron effect to the measurements we get the bias-free measurements in the body frame as shown below. Note we still need to account for the soft-iron effect.

$$\mathbf{B}_{bias-free}^b = \mathbf{B}^b - \mathbf{V} = \mathbf{W} \mathbf{R}_l^b \mathbf{B}^l \quad B.8$$

The bias-free ellipsoid now takes the following form:

$$\mathbf{A}_{BF}x^2 + \mathbf{B}_{BF}y^2 + \mathbf{C}_{BF}z^2 + 2\mathbf{D}_{BF}xy + 2\mathbf{E}_{BF}xz + 2\mathbf{F}_{BF}yz = 0 \quad B.9$$

Performing a new least squares adjustment on the bias-free measurements gives us \hat{x}_{BF} , which can be used to build the new bias-free ellipsoid coefficient matrix:

$$\tilde{\mathbf{A}}_{BF} = \begin{pmatrix} \hat{x}_{BF(1,1)} & \hat{x}_{BF(4,1)} & \hat{x}_{BF(5,1)} \\ \hat{x}_{BF(4,1)} & \hat{x}_{BF(2,1)} & \hat{x}_{BF(6,1)} \\ \hat{x}_{BF(5,1)} & \hat{x}_{BF(6,1)} & \hat{x}_{BF(3,1)} \end{pmatrix} = \begin{pmatrix} A_{BF} & D_{BF} & E_{BF} \\ D_{BF} & B_{BF} & F_{BF} \\ E_{BF} & F_{BF} & C_{BF} \end{pmatrix} \quad B.10$$

The eigenvalues, $\lambda = (\lambda_1, \lambda_2, \lambda_3)^T$, and eigenvectors, \bar{v} , of \tilde{A}_{BF} are computed. These values allow us to compute the inverse soft-iron matrix.

$$\mathbf{W}^{-1} = \bar{v} \lambda \bar{v}^{-1} \quad B.11$$

Now that the hard-iron and soft-iron effects have now been found, the corrected magnetometer readings in the l-frame can be computed from the incoming raw measurements by re-ordering Eq. B.1:

$$\mathbf{B}^l = \mathbf{W}^{-1} \mathbf{R}_b^l (\mathbf{B}^b - \mathbf{V}) \quad B.12$$

Finally, the estimated heading from the corrected magnetometer readings is found through:

$$\psi = \tan^{-1} \left(-\frac{B_{ly}}{B_{lx}} \right) \quad B.13$$

CURRICULUM VITAE

Candidate's Full Name: Eduardo Infante

Universities Attended:

- 2005-2009 Bachelor of Science in Engineering, University of Calgary

Publications:

Wong Ken, N., Infante, E., Dixon, R. (2014). *Marine Performance of NovAtel SPAN GNSS/INS System*. CHC2014. Canadian Hydrographic Service.