Higher-Order GPS Noise Models, with Applications to Position Inference and Vehicle Navigation

Andrew Walden Ross Soundy

October, 2016

A thesis submitted for the
Masters of Science
in Electronics

University of Otago
New Zealand

Acknowledgements

Thanks Tim, this has been an incredible experience and there is no way that I would have learned as much or achieved as much if you were not my supervisor. Danny, thanks for always being around and happy to help; your feedback, help with papers, and advice has been invaluable. Brad, thanks for showing me the ropes, helping me with programming, and all your work on the data gathering. Finally, thank you to the army of friends and family who helped with editing, suggestions, and convincing me that I could actually finish, somehow you were right.

Contents

List of Figures

Abstract

This thesis presents an experimental and theoretical investigation of GPS errors. Data from stationary GPS units were gathered from several locations and an analysis of the noise present was undertaken. Several noise models were proposed and their relative performance compared using the Akaike Information Criterion. A vehicle navigation system was implemented using a dynamical vehicle model which combined GPS, accelerometer and gyroscope measurements via a nonlinear Kalman filter to enable superior position and velocity estimation, especially during short-term losses of GPS signal.

Chapter 1

Introduction

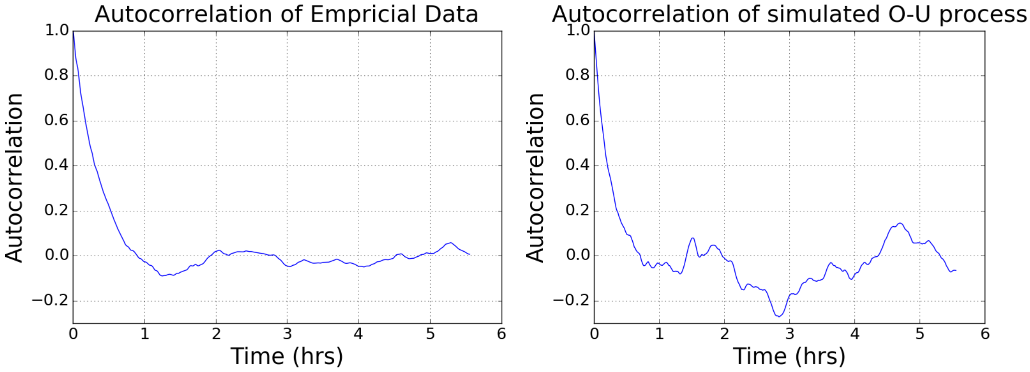
The Global Positioning System (GPS) and other global navigation satellite systems (e.g., GLONASS and Galileo) have become ubiquitous in everyday life. This thesis investigates how we can optimally use the data obtained from GPS devices. We investigate this question through two separate, but related cases, described in Sections 1.1 and 1.2 below.

1.1 GPS Noise Models

A commonly made assumption is that the noise from GPS devices is independent, identically distributed Gaussian noise [1, 2]. This thesis challenges this assumption, finds that it isn't valid, which agrees with some of the literature [3, 4]. We then propose improved noise models that fit observations better, and discusses how existing sensor fusion techniques can be extended to these new noise models. In addition, this thesis describes how dynamical vehicle models can be used to infer better position and velocity information for vehicle navigation.

Understanding the nature of the noise present in measurements helps in the extraction of signals from noisy data. In Chapter 2 we show that the independence assumption of GPS noise is fundamentally flawed and we present several alternative noise models that provide a better fit to the data and allow for time dependent noise. Figure 1.1 shows the autocorrelation function, a measure of the signal correlation after a given time delay, of empirical GPS data (a), and for one of the proposed time dependent noise models, the Ornstein-Uhlenbeck model (b).

A common reason to make the independence assumption is that filtering with independent noise is mathematically more tractable than filtering with time dependent noise. However, we demonstrate that filtering with a time dependent noise model is practical by implementing one of the proposed time dependent models, the Ornstein-Uhlenbeck noise model, in a Kalman filter. We believe that Kalman filtering single-frequency GPS data with an Ornstein-Uhlenbeck noise model is new to the literature, and could lead to improved devices from satellite position measurement [5].



(a) (b)

Figure 1.1: The autocorrelation of (a) empirical GPS position errors (latitude), and (b) a simulated Ornstein-Uhlenbeck process, with parameters estimated from the data. The GPS errors in position estimates are correlated for more than 30 minutes, i.e. if a position measurement is incorrect, then 30 minutes late it is still likely to be incorrect in the same direction.

1.2 Inference of Driver Behaviour using Vehicle Models, and GPS and Inertial Measurements

In this thesis we also describe a dynamical vehicle model that allows inference of driver behaviour, specifically acceleration and braking. By placing constraints on the vehicle dynamics of the model we are able to limit the degrees of freedom in the system and so extract more accurate state estimates from the available measurements.

The model is used by a sensor fusion algorithm to combine GPS, accelerometer, and gyroscope measurements. GPS measurements represent a cheap, accurate tool for determining position with an accuracy as low as 3 m for a standalone unit [6, 7]. GPS units also output velocity measurements which typically have an accuracy of less than 1ms^{-1} [8]. However, the sampling frequency of GPS devices is quite low compared to many other digital sensors.[1] Additionally, the noise for a standalone GPS unit is significant and GPS signals can be subject to obstruction [7, 9].

By contrast, inertial sensors typically have a much higher sampling frequency, in the order of hundreds of Hz [7, 10], and noise that is independent of GPS measurements. By combining these separate measurements of a vehicle using a nonlinear Kalman filter we can infer any change in the speed, heading, and driver inputs.

The system developed in this thesis can be deployed on a vehicle without any

---

[1] Most GPS devices only receive data at a rate of 1 Hz, although with reconfiguration devices can sample at a rate of 5 Hz or (with some difficulty) 10 Hz [2].

modification of hardware or interaction with the software of the vehicle and provides superior performance than either a standalone GPS unit, or a GPS and inertial navigation system that makes no assumptions on the possible vehicle dynamics. The system can also successfully bridge short-term GPS outages.

1.3 Satellite Navigation

There are several global navigation satellite systems (GNSSs) in varying degrees of operational capability. As of March 9, 2016 the only systems with global coverage are the American Global Positioning System (GPS) and the Russian GLObal NAvigation Satellite System (GLONASS), which have 31 and 29 satellites in orbit, respectively [11,12]. The Chinese BeiDou Navigation Satellite System currently has 21 satellites in orbit and is in a state of partial operation as it can provide positioning information between longitudes of 55°E and 180°E and between latitudes of 55°S and 55°N [13,14]. At its current stage the European Galileo system is half-way operational with 12 satellites in orbit [15].

1.3.1 GPS

GPS is still the most commonly used system and all GNSS measurements made in this thesis use the GPS constellation. The first GPS satellite was launched in 1978 and the constellation was declared fully operational in 1995 [16]. As of March 9, 2016 the GPS constellation has more satellites in orbit than any other system and the Federal Aviation Administration (FAA) has shown single frequency GPS to have a horizontal accuracy of less than 3.5 m across North America [11,17].

There are many sources that contribute to GPS errors. The dominant source of GPS error is ionospheric effects due to the dispersion of the microwave signal in the ionised upper atmosphere. Dual frequency GPS units are able to largely eliminate the errors from ionospheric effects by calculating the atmospheric dispersion from the different time delays on the L1 (1575.42 MHz) and L2 (1227.60 MHz) signals [7]. However, single frequency GPS units cannot make these corrections and dual frequency GPS receivers are considerably more expensive (∼\$2000+) than single frequency receivers (∼\$10+).

There are also errors due to the propagation through the troposphere, these are typically smaller than the ionospheric errors, however, the troposphere is non-dispersive and so cannot be corrected for by using the different time delay of dual frequency signals. Around 90% of the tropospheric error is due to dry gases, which have a relatively stable concentration and may be largely corrected for by models that do not require any extra information. The remaining 10% of the tropospheric error is due to water vapor and so is difficult to correct for without real-time weather data being taken into account [7].

Table 1.1: The User Equivalent Range Error budget for the Standard Positioning Service, with the contributions being within a 95% confidence interval.

Source	UERE Contribution (95%) metres [6]
Group Delay Stability	3.1
Other Space Segment Errors	1.0
Clock/Ephemeris Estimation	2.0
Clock/Ephemeris Curve Fit	0.8
Ionospheric Delay Terms	9.8–19.6
Group Delay Time Corrections	4.5
Other Control Segment Errors	1.0
Tropospheric Delay	3.9
Receiver Noise and Resolution	2.0
Mulitpath	0.2
Other User Segment Errors	1.0

The local geography may also affect the GPS error by reflected signals contributing to mulitpath effects. The reflections have a longer pathlength between the receiver and the satellite, this longer pathlength then affects the position reported by the device. Because of the longer pathlength and some attenuation by the reflecting surface, the reflected signals are necessarily delayed with respect to the true pathlength and of smaller amplitude. However, in the case where the true signal is also attenuated, e.g., by foliage or local geography, the reflected signal may be mistaken for the true signal, which can then result in a significant position error [11].

Table 1.1 shows the stated User Equivalent Range Error (UERE) budget for the Standard Positioning Service (SPS) provided by the US Department of Defense and GPS NAVSTAR groups. All errors provided are 95th percentile error bounds [6]. It is worth noting that the errors shown in Table 1.1 cover the stated tolerances for the performance of the GPS satellites and receivers. However frequency standards on board the satellites have consistently exceeded their specifications, and with the steady improvements to satellite software and receiver hardware, measured errors are often considerably smaller than what would be expected from the stated error budget [18].

If high-precision measurements are required then the user may be able to use a dual frequency GPS which can correct for the majority of the ionospheric effects. Also accurate weather data can be used to estimate the amount of moisture in the air and the effect of that on the tropospheric errors. While dual frequency measurements and real-time weather models can substantially reduce the errors in GPS measurements these methods are outside the means of everyday users of GPS.

For the purposes of this thesis we take an agnostic position to the source of the noise observed in the GPS signal. We treat the GPS unit as a ‘black box’, and investigate statistical models of the observed noise. Using this approach we filter GPS measurements in a way that does not require any extra information and that would be generally applicable to the large number of uses of satellite navigation.

1.4 The Kalman Filter

The Kalman filter is a recursive algorithm for sequential inference and sensor fusion pioneered by Rudolf E. Kalman and Richard S. Bucy [19]. The Kalman filter propagates both an estimate of the state of the system and an estimate of the uncertainty of that state. The Kalman filter can operate on systems with arbitrary additive noise models. However, in the specific case of Gaussian noise the Kalman filter yields the exact posterior probability density function [20, 21]. Consequently the Kalman filter provides optimal estimates for the current state and covariance for any linear system with Gaussian noise [22]. This, in combination with the central limit theorem, makes the Kalman filter a very powerful algorithm.

The Kalman filter assumes that the next state of the system, $\mathbf{x}(k+1)$, is a linear combination of the current state, $\mathbf{x}(k)$, and some additive noise, $\mathbf{w}(k)$. The measurement model of the Kalman filter then assumes that some measurement, $\mathbf{z}(k)$, is made on the system which is a linear projection of the current state onto the measurement space. The measurement process is also assumed to have some inherent noise, $\mathbf{v}(k)$. The entire process can be expressed mathematically as,

$$\begin{aligned} \mathbf{x}(k+1) &= F(k)\mathbf{x}(k) + \mathbf{w}(k), \\ \mathbf{z}(k) &= H(k)\mathbf{x}(k) + \mathbf{v}(k). \end{aligned} \tag{1.1}$$

Where the state transition matrix $F(k)$ is used to propagate the current state forwards in time and is derived from the dynamics of the system. The measurement matrix $H(k)$ projects the current state onto the measurement space.

The Kalman filter consists of two steps, prediction and measurement. Equation 1.2 begins with the $(k-1)^{\text{th}}$ state estimate given $(k-1)$ measurements, and it is denoted by $\hat{\mathbf{x}}(k-1|k-1)$. The covariance of this state estimate is denoted by $P(k-1|k-1)$. The current state estimate is then propagated forward in time by the state propagation matrix to give a prediction of the next state. The next measurement of the state $\mathbf{z}(k)$ is then made and the prediction and measurement are combined, weighted by the Kalman gain $K(k)$, a factor calculated from the state covariance, measurement covariance, and measurement matrices, as shown in Equation 1.3.

Prediction

$$\begin{aligned} \hat{\mathbf{x}}(k|k-1) &= F(k)\hat{\mathbf{x}}(k-1|k-1), \\ P(k|k-1) &= F(k)P(k-1|k-1)F(k)^T + Q(k), \end{aligned} \tag{1.2}$$

Measurement

$$\begin{aligned} K(k) &= P(k|k-1)H(k)^T\Big(H(k)P(k|k-1)H(k)^T + R(k)\Big)^{-1}, \\ \hat{\mathbf{x}}(k|k) &= \hat{\mathbf{x}}(k-1|k-1) + K(k)\Big(\mathbf{z}(k) - H(k)\hat{\mathbf{x}}(k|k-1)\Big), \\ P(k|k) &= \left(I - K(k)H(k)\right)P(k|k-1), \end{aligned} \tag{1.3}$$

The reader will note that the system and measurement noise, $\mathbf{w}(k)$ and $\mathbf{v}(k)$, respectively, are both missing in the above algorithm, since the expectation value of zero-mean noise is zero. Therefore this term is not included in the expectation of the state. However, the covariance matrices of the noise terms are included as the $Q(k)$ and $R(k)$ matrices for the system and measurement noise, respectively.

The Kalman gain of the system depends on the predicted state covariance, $P(k|k-1)$, as well as the covariance of the system noise, $Q(k)$, and the covariance of the measurement noise, $R(k)$. This automatically determines the weighting given to the predictions and measurements. If the measurement covariance is larger than the system covariance then the Kalman gain places a stronger emphasis on the predictions than the measurements and vice versa [19].

1.5 Nonlinear Extensions to the Kalman Filter

While the Kalman filter is an extremely useful algorithm, it assumes that the aforementioned measurement and propagation equations are linear. However, many real-world problems are nonlinear, for example, the problem of combining GPS and inertial measurements for navigation. Therefore we must look into the set of nonlinear state estimators.

1.5.1 Extended Kalman Filter (EKF)

The extended Kalman filter was the first attempt at using a Kalman filter-based algorithm to estimate a system with nonlinear state dynamics, and was developed soon after the Kalman filter. The EKF handles the nonlinear state dynamics by taking a first order Taylor series expansion of the state propagation and measurement equations around the current state estimate. In the following equations we denote the true state of the system at time $k\Delta t$ as $\mathbf{x}(k)$ and the state estimate, given measurements $\mathbf{z}(1)\ldots\mathbf{z}(k)$, as $\hat{\mathbf{x}}(k|k)$.

$$\begin{aligned} \mathbf{x}(k+1) &= f(\mathbf{x}(k)) + \mathbf{w}(k), \\ \mathbf{z}(k+1) &= h(\mathbf{x}(k)) + \mathbf{v}(k), \end{aligned} \tag{1.4}$$

$$\begin{aligned} f(\mathbf{x}(k)) &= f(\hat{\mathbf{x}}(k|k)) + J_f(\hat{\mathbf{x}}(k|k) - \mathbf{x}(k)) + \dots, \\ &\approx f(\hat{\mathbf{x}}(k|k)) + J_f(\hat{\mathbf{x}}(k|k) - \mathbf{x}(k)), \end{aligned} \tag{1.5}$$

$$\begin{aligned} h(\mathbf{x}(k)) &= h(\hat{\mathbf{x}}(k|k)) + J_h(\hat{\mathbf{x}}(k|k) - \mathbf{x}(k)) + \dots, \\ &\approx h(\hat{\mathbf{x}}(k|k)) + J_h(\hat{\mathbf{x}}(k|k) - \mathbf{x}(k)), \end{aligned} \tag{1.6}$$

$$J_f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}, \quad J_h = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial h_n}{\partial x_1} & \frac{\partial h_n}{\partial x_2} & \cdots & \frac{\partial h_n}{\partial x_n} \end{bmatrix}. \tag{1.7}$$

Equation 1.5 shows that the extended Kalman filter takes the nonlinear function $f(\mathbf{x}(k))$ and truncates the Taylor series at the first order, effectively linearising the state dynamics at every prediction step. This means that propagating a Gaussian probability density function (pdf) through the linearised system dynamics will allow the pdf to stay as a Gaussian, which enables the state estimate and state covariance matrix to uniquely characterise the uncertainty.

However, for highly nonlinear systems a first order expansion of the Taylor series may not be sufficient. In this case another nonlinear filter must be applied. A good candidate would be the unscented Kalman filter (UKF) which can be shown to be equivalent to a second order Taylor series expansion for any type of noise, or a third order expansion if the system and measurement noise are Gaussian [23].

1.5.2 Unscented Kalman Filter (UKF)

The extended Kalman filter deals with nonlinear dynamics by linearising the dynamics at each step and then propagating the pdf through the linearised dynamics. Rather than approximating the system dynamics as linear in order to be able to use a Gaussian pdf, the unscented Kalman filter retains the full nonlinear system dynamics and then approximates the resulting pdf as a Gaussian [24].

The UKF works by selecting a deterministic group of points, called sigma points. These sigma points are chosen to have the same mean and covariance as the current state estimate. Once the points have been chosen they are propagated through the nonlinear system dynamics $f(\mathbf{x}(k))$ directly [25],

$$\begin{aligned} \mathbf{x}(k+1) &= f(\mathbf{x}(k)) + \mathbf{w}(k), \\ \mathbf{z}(k+1) &= h(\mathbf{x}(k)) + \mathbf{v}(k). \end{aligned} \tag{1.8}$$

After the sigma points have been propagated the mean and covariance is taken from the propagated points and that is taken to be the predicted mean and predicted covariance of the state.

Because the UKF does not truncate the higher order nonlinearities of the system, we expect that it has superior performance to the EKF in highly nonlinear systems [23]. For systems that are not dominated by nonlinear terms the UKF and EKF have often been found to have comparable performances, both in terms of predictive power and computational cost [26, 27].

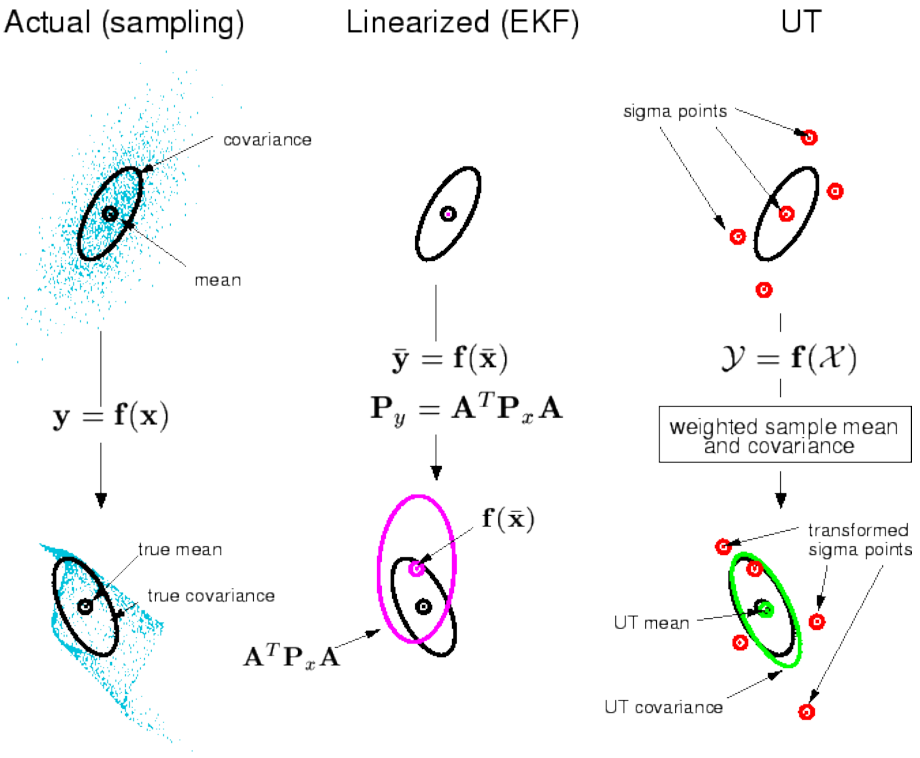


Figure 1.2: Some probability density function is propagated through the nonlinear function $f(\mathbf{x})$, with the propagated pdf shown, along with the mean and covariance of the pdf. In the centre we see the pdf propagated through the linearised dynamics of the EKF. On the right we see the unscented transform (UT) of the UKF, first generating the sigma points, then propagating them and finally transforming them back to the mean and covariance. Image taken from http://www.cslu.ogi.edu/nsel/ukf/img84.gif [24].

Particle Filter

Particle filters, also known as sequential Monte Carlo methods, are a set of filters that represent the probability density function of a state by a series of particles, such that the distribution of particles approximates the pdf of the state. These particles are then propagated directly through the nonlinear dynamics of the system. The numerical representation of the pdf means that particle filters can, in principle, handle any pdf. By increasing the numbers of particles one may increase the accuracy of the particle filter, allowing the user to scale the accuracy to the specific application of the filter [28]. The cost of being able to handle arbitrary pdfs and nonlinear dynamics is the extra computational complexity of the particle filter [29].

While the UKF only requires $(2n+1)$ sigma points given an n-dimensional state space, the particle filter typically requires an order of magnitude more points [30]. This is because the UKF utilizes the unscented transform to deterministically choose the sigma points while the particle filter usually chooses the particles at random [28].

Because the particle filter scales so poorly with increasing dimensions we only looked at the EKF and UKF for real time navigation. With the explosion of computing power soon it may be practical to implement a particle filter in a multi-dimensional system such as integrated GPS and inertial navigation. A useful future work would be to use a particle filter off-line to evaluate the performance of both the EKF and UKF.

1.6 Sensor fusion with GPS data

Given the low cost and size of digital sensors, devices with many integrated sensors are now commonplace. The reader probably has, in his or her pocket, a computer with a GPS, accelerometer, gyroscope and magnetometer, capable of performing the kind of nonlinear state estimation used in Chapter 7. Such sensors are becoming smaller, faster, and less power-intensive. With this sudden boom in potential data sources comes the necessity of filtering and processing this data quickly and effectively.

Problems, such as losing satellite visibility due to geographic or man-made features, are common in GPS navigation [9]. By combining an inertial navigation unit (INU) with GPS data the problem of short-term losses of signal can be significantly reduced. We demonstrate this in Figure 1.3 by simulating a loss of GPS signal by removing fifteen seconds of GPS data from the data set. We can then see the results of this fifteen second loss of GPS signal on a combined GPS and inertial navigation system. The red path in Figure 1.3 shows the GPS measurements with a fifteen second loss of signal. The blue path shows the estimated position by relying on the inertial measurements and the algorithm developed for this thesis.

Figure 1.3: The red waypoints are the GPS signal that the combined navigation system was using. We simulated a loss of GPS signal by removing fifteen seconds of GPS data to see how the system would cope. The blue waypoints are the estimated position from the combined system.

1.7 Chapter Outlines

The chapters in this thesis are placed depending on their relation to the two separate, but related, aims of this thesis. Chapters 2-4 relate to the analysis, characterisation, and filtering of single-frequency GPS noise. Whereas, Chapters 5-7 relate to the design, implementation, and experimentation of a system for tracking road-based vehicles by combining GPS and inertial measurements using a dynamical vehicle model. The individual chapter outlines are given below.

- Chapter 2. A background to GPS noise is presented, as well as some problems with using an independent noise model, as demonstrated by empirical GPS errors with long-term autocorrelation functions. Different noise models are presented, as is the Akaike Information Criterion (AIC), by which the models will be compared.

- Chapter 3. Single-frequency GPS data was collected by two different types of single-frequency GPS units from four separate locations. The data gathered is then analysed and the relative performance of the models, proposed in Chapter 2, is presented.

- Chapter 4. A background to Kalman filtering with GPS measurements is presented. A derivation of how to filter GPS measurements with an Ornstein-Uhlenbeck (O-U) noise model is given, as well as a demonstration of such filtering.

- Chapter 5. An introduction of inertial navigation is presented, using a one-dimensional acceleration example. The benefits of using inertial navigation and sensor fusion is then demonstrated, using this example. The reference frames that will be used for the vehicle tracking system are also presented.

- Chapter 6. A dynamical vehicle model for road-based vehicles is presented. The system of implementing the model in a nonlinear Kalman filter is shown and the measurement and propagation equations given. A simulated two-dimensional test is also given, as proof of concept for the model.

- Chapter 7. The dynamical vehicle model from Chapter 6 is implemented and tested with real-world data. The hardware used and results of the vehicle tracking system are presented and analysed.

Chapter 2

Modelling GPS Noise

This chapter presents experimental data illustrating some of the characteristics of GPS noise. Different noise models are proposed as candidates for the noise observed in GPS measurements. Each of these models are fit to experimental data, and their relative performance analysed in Chapter 3. The selection criteria used for model comparison was the Akaike Information Criterion (AIC), which is explained in Section 2.3.

In Section 2.1 we present the autocorrelation function of empirical GPS data. The presence of a long-term autocorrelation function presents significant problems for the commonly used independent, identically distributed (i.i.d.) Gaussian model as a noise model for GPS noise [1, 31].

Candidate models are proposed in Section 2.4. For each model we derive the log likelihood functions that are used by the Akaike Information Criterion for the model selection performed in Chapter 3. Example autocorrelation functions of each noise model are also presented.

In this chapter and chapter 3 GPS position data from known positions are analysed. We then characterise the error signal N_t, and propose noise models that fit the data better than the independent, i.i.d. Gaussian model in common use. We expect that better noise models will lead to superior performance.

2.1 GPS Noise Background

Accurate uncertainty estimation is a crucial part of any filtering process. Position estimation, of course, is no exception. Any GPS measurement of position contain uncertainties in that measurement. Typically the measurement is modeled by assuming that the GPS device reports the true position X_0 with some additive Gaussian noise [1] i.e,

$$X_t = X_0 + N_t, \tag{2.1}$$

where N_t is usually taken to be independent and normally distributed with zero mean and variance of σ^2.

Previous research has identified the problems with fitting an i.i.d. Gaussian noise model to GPS measurements [5, 32]. Two models have been proposed to fit the measurement noise in GPS measurements more accurately. One is a moving-average (MA) process, the other is an autoregressive (AR) process [32, 33]. In this, and the following Chapter we will analyse both to determine which model fits the GPS errors best. We also propose the Ornstein-Uhlenbeck as a simple, continuous time analogue to the autoregressive process that will be tested alongside the AR and MA processes.

2.2 Autocorrelation of GPS Noise

The autocorrelation (or autocovariance) between X_t and X_s for some process $\{X_t\}$ is given by [34]

$$cov(X_t, X_s) = \mathrm{E}[(X_t - \mathrm{E}[X_t])\,(X_s - \mathrm{E}[X_s])] \tag{2.2}$$

where $\mathrm{E}[X]$ is the expectation operator. One of the advantages of i.i.d. Gaussian noise is that the theoretical autocorrelation function is zero after lag of dt, where dt is the sampling period.

However, Figure 2.1 clearly shows that the autocorrelation function of empirical GPS data does not reach zeros after the one second sampling period. Indeed, it takes approximately an hour for the autocorrelation to reach zero. This data was taken from a GlobalSat BU-353S4 GPS unit operating on the L1 frequency (1575.42 MHz) with sampling period of 1 second over a total time of around 100 hours, or 360,000 individual measurements.

The Kalman filter is an optimal linear state estimator and is described in Section 1.4. In order for a Kalman filter to function, the covariance matrices of the process and measurement noises needs to be known.

In the case of i.i.d. Gaussian noise the covariance matrix does not need to be updated for as long as the filter is running. Therefore a commonly made assumption in many Kalman filter-based systems is that the measurement noise term is independent from the previous noise terms [1, 7]. However the empirical data that we obtained implies that in the case of GPS measurements this assumption is flawed.

A common method of determining the independence of one measurement to the next is to look at the autocorrelation function of the measurements [22, 35]. The autocorrelation function ($R(\tau)$) of a signal indicates of how strongly correlated a signal is with itself after a time delay of τ seconds. If the autocorrelation function is well defined then it should be bounded by [1,-1], with 1 representing prefect correlation and -1 representing perfect anticorrelation. The autocorrelation function of discrete data can be estimated with the following equation [34].

$$R(\tau) = \frac{\sum_{t=\tau+1}^{n}(X_t - \mu)(X_{t-\tau} - \mu)}{\sum_{t=1}^{n}(X_t - \mu)^2}. \tag{2.3}$$

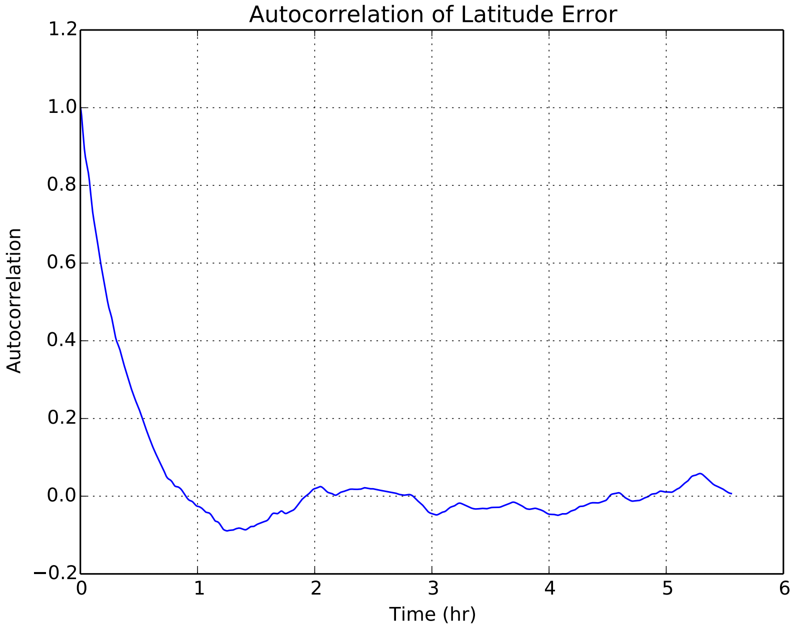
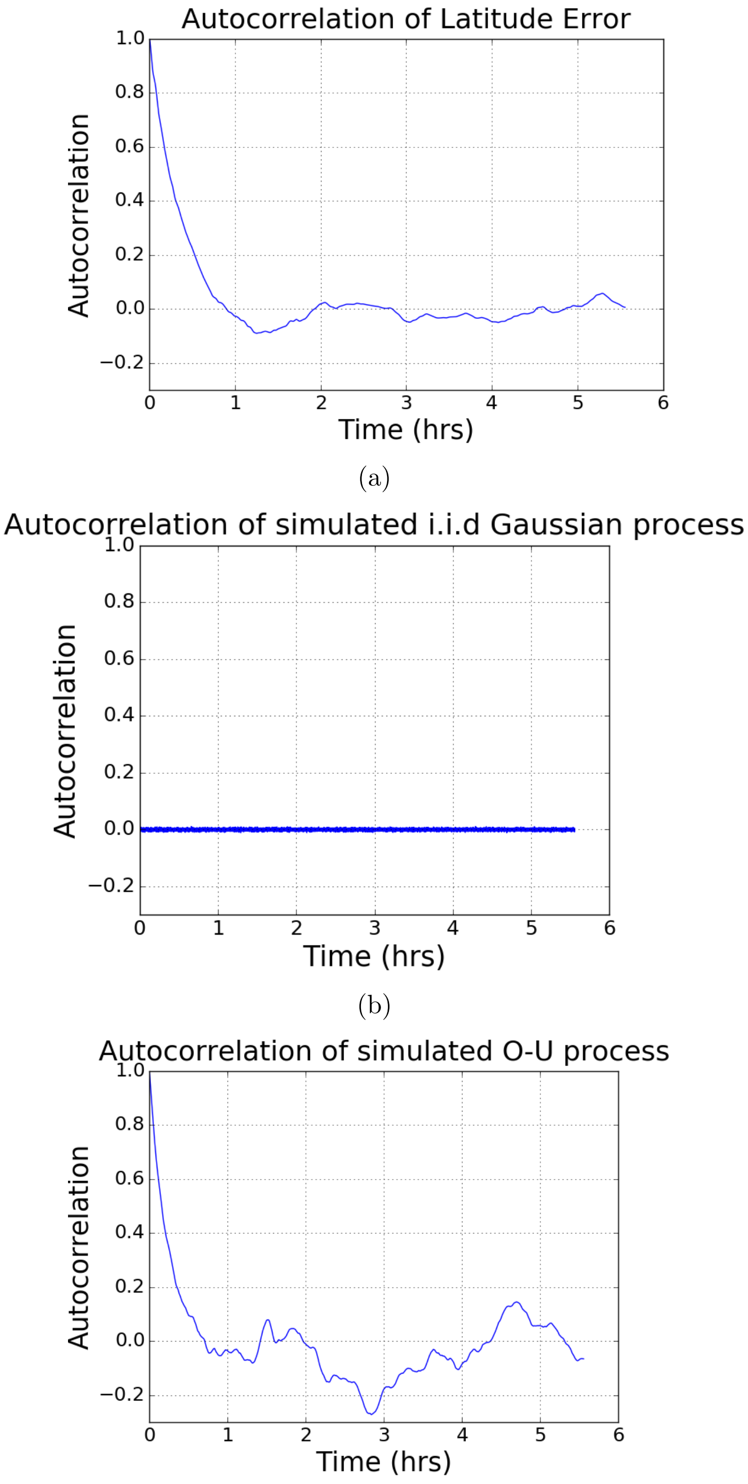


Figure 2.1: Autocorrelation of latitude errors taken from a GlobalSat BU-353S4 GPS unit with excellent sky view. The sampling period of the device is 1 second. This data is taken from a data set of $\sim$ 100 hours but the time axis has been cut off at 5.5 hours for readability.

In Equation 2.3 the correlation between measurement X_t and $X_{t+\tau}$ is given by $R(\tau)$ and the μ term represents the mean of the set of data $\{X_i\}$. As mentioned above the autocorrelation function of i.i.d. Gaussian noise is zero after a lag of dt. To illustrate the difference between this and the autocorrelation of GPS data, we generated some i.i.d. Gaussian noise with the same length, mean and variance as the GPS data seen in Figure 2.1. The autocorrelation function of the generated i.i.d. Gaussian noise is shown in Figure 2.2(b). It is clear that the autocorrelation function is, indeed, close to zero after a lag of one sampling period. For comparison the autocorrelation of the GPS data is shown in Figure 2.2(b), along with the autocorrelation function of a simulated Ornstein-Uhlenbeck process Figure 2.2(c).

However it is equally clear from Figure 2.1 that the autocorrelation function for the measured latitude error does not quickly decay (keeping in mind that the data has a sampling period of 1 second and the time scale shown is in hours). This time dependence implies that if a GPS position measurement is offset from the true position then it is statistically likely to be offset in that same direction for up to an hour.



(a)

(b)

(c)

Figure 2.2: The autocorrelation of empirical GPS data, (b) independent Gaussian noise with the same mean and variance as the observed data, and (c) a simulated Ornstein-Uhlenbeck process. For readability only the first 5.5 hrs of the autocorrelation function were shown.

We chose to treat the GPS measurements as a black box and simply attempt to model the noise observed and modify our filtering algorithms accordingly. This approach has some appeal as it allows us to remain vendor neutral. Also we wanted to propose ways of modeling GPS noise that are generally applicable across all GPS devices, rather than simply modeling the noise present in one specific device.

2.3 Model Comparison

In order to determine which model is the best out of a candidate set we require some selection criteria to order them. Ideally the criteria would reward models that fit the data well and penalise overly complex models. This balance between complexity and goodness of fit is sometimes called parsimony after William of Ockham's law of parsimony (commonly referred to as Ockham's razor). The law of parsimony states that "Plurality must not be posited without necessity." This notion is often restated as "Among competing ideas, favour the simplest one." [36].

In essence, model selection is about determining how likely a given model is relative to the other models in the candidate set. Before we attempt to do so it would be worth defining our terms. Below we have Bayes' Rule [37].

$$P(x|\theta) = \frac{P(\theta|x)P(x)}{P(\theta)}, \tag{2.4}$$

Where $P(x|\theta)$ is the probability of the data (x) given some model parameter (θ). This is proportional to the probability of the model parameter given the data ($P(\theta|x)$), multiplied by the probability of the data ($P(x)$). The probability of the model parameter given the data ($P(\theta|x)$), is known as the 'likelihood' of the model parameter given the data.

The right hand side of the equation is then normalized by dividing through by the probability of the model parameter ($P(\theta)$).

2.3.1 Log likelihood

Let $\{x_i\}$ be the set of measurements $x_1, x_2, ...x_N$, which are distributed according to some probability density function $f(x|\theta)$, where θ is the set of independently adjustable parameters that describe the probability density function (pdf). To obtain the joint pdf of the measurements $\{x_i\}$, we get.

$$\begin{aligned} f(\{x_i\}|\theta) &= f(x_1|\theta) \cdot f(x_2|\theta) \cdot ... \cdot f(x_N|\theta), \\ &= \prod_{i=1}^{N} f(x_i|\theta), \end{aligned} \tag{2.5}$$

Often when working with large data sets the likelihood functions are very small, even for relatively orderly predictions (imagine a system where each measurement has a likelihood of 0.9 and you have a million measurements, $0.9^{1000000}$ is a still very small number). Because of this it is common to work with the log of the likelihood functions, i.e.

$$\begin{aligned}\ln(L) &= \ln\left(\prod_{i=1}^{N} f(x_i|\theta)\right), \\ &= \sum_{i=1}^{N} \ln(f(x_i|\theta)),\end{aligned} \tag{2.6}$$

It is also worth noting that implicit in Equation (2.6) is that the measurement probabilities are independent of each other. This is not necessary, for non-independent probabilities we simply add in the conditional pdfs as required. For example if each measurement was dependent on the previous measurement then (2.6) would become.

$$\begin{aligned}\ln(L) &= \ln\left(f(x_1|\theta)\cdot\prod_{i=2}^{N} f(x_i|\theta, x_{i-1})\right), \\ &= \ln(f(x_1|\theta)) + \sum_{i=2}^{N} \ln(f(x_i|\theta, x_{i-1})),\end{aligned} \tag{2.7}$$

Because the logarithm is a monotonically increasing function the logarithm of a function will be maximised when the function itself is maximised.

2.3.2 Maximum Likelihood Estimation

Maximum likelihood estimation (MLE) is firmly based in the Bayesian framework of statistics and so it is worth defining Bayes' rule for the set of measurements $\{x_i\}$.

$$P(\theta|\{x_i\}) = \frac{f(\{x_i\}|\theta)P(\theta)}{P(\{x_i\})}, \tag{2.8}$$

Where $P(\theta|\{x_i\})$ is the probability of the parameters given the data, $P(\theta)$ is the probability of the parameters and $P(\{x_i\})$ is the probability of the data. Obviously for model comparison we want to choose the most likely model parameters given the data, therefore we want to maximise $P(\theta|\{x_i\})$. Note that on the right hand side of Bayes' rule the denominator is independent of θ. So if we assume a uniform prior distribution across the parameter space (i.e. choose $P(\theta)$ to be constant) then by maximising $f(\{x_i\}|\theta)$ we will maximise $P(\theta|\{x_i\})$.

Note that the constant terms from (2.8) are left out of the log likelihood equation, this is standard for both the likelihood and the log likelihood as, in general, we are not interested in the constant terms and only seek to maximise the joint pdf of the data given the parameters of the model.

To obtain the maximum likelihood we can maximise the log likelihood as this will return the maximum likelihood parameters [38]. To obtain the maximum likelihood we simply take the partial derivative of the log likelihood with respect to each of the parameters and select each maximal parameter accordingly, i.e.

$$\begin{aligned} \frac{\partial \ln(L)}{\partial \theta_1} &= \partial \ln(f(x_1|\theta)) + \sum_{i=2}^{N} \ln(f(x_i|\theta, x_{i-1}))\partial\theta_1 = 0, \\ \frac{\partial^2 \ln(L)}{\partial \theta_1^2} &\leq 0, \end{aligned} \tag{2.9}$$

If the equations cannot be solved analytically then there are many, well-known numerical ways of obtaining accurate approximations [39]. Once the maximum likelihood estimators of the parameters have been calculated obtaining the maximum likelihood is a simple matter of inputting the parameters into your likelihood function along with $\{x_i\}$.

We used Maximum Likelihood Estimation to determine the optimal parameters for each of the noise models discussed in Section 2.4.

2.3.3 Akaike Information Criterion

There are many ways to try and characterize the fit of a particular model to the data. For our noise model comparisons we will use the Akaike Information Criterion (AIC) [38]. The definition for the AIC is given below.

$$AIC = 2k - 2\ln(L), \tag{2.10}$$

Where k refers to the number of independently adjustable parameters within the model and L refers to the maximum likelihood of the parameters given the data [38]. The model which minimises the AIC value for the given data is said to be the preferred model [40]. However, it is important to note that the AIC value of a particular model for some given set of data says nothing about the absolute likelihood of that model's fit to the data. The AIC value of a model only provides relative information for comparison between models, i.e. if all of the candidate models fit the data poorly the AIC values won't reflect the poor fit.

A common application of the AIC is to use the AIC number of a model to determine how likely one model is relative to another. The process of obtaining relative likelihoods from AIC values is reasonably simple. Let M be the set of candidate models $M_1, M_2, \ldots, M_n$, each model has its associated AIC value $AIC_1, AIC_2, \ldots, AIC_n$.

Of the set of AIC values suppose model j has the lowest AIC value, which we denote AIC_{min}. One of the useful aspects of the AIC is that we can quantitatively determine the probability that model M_i is the best among the candidate set M. This is done by calculating the model's Akaike weight w_i [40].

$$w_i = \frac{\exp\left[\frac{(AIC_{min} - AIC_i)}{2}\right]}{\sum_{m=1}^{n} \exp\left[\frac{(AIC_{min} - AIC_m)}{2}\right]} \tag{2.11}$$

For example, imagine that model M_i's Akaike weight is 0.25. In that case there is a 25% chance that of the candidate models, M_i is the best. The Akaike weights also provide a convenient way of comparing two models and calculating quantitatively which is better. This is achieved by dividing the Akaike weights, which gives us the evidence ratio of the two models. For example let us compare model j against model i.

$$\text{Evidence ratio} = \frac{w_j}{w_i}, \tag{2.12}$$

Let's say the w_j is 0.35 and w_i is 0.25, this gives an evidence ratio of 1.4. This means that model M_j is 1.4 times more likely than model M_i. For ease of interpretation when we compare the performance of different models we will include both the AIC value of each model and its associated Akaike weight.

Now that we are versed in the AIC and MLE framework we can begin to use both to try and characterize the noise distributions that we observe in GPS errors.

2.4 Candidate Models for GPS noise

As mentioned above the Akaike Information Criterion is only useful for comparing candidate models against each other. The AIC gives no information as to the absolute fit of the model to the data. Therefore, it is important to consider models that have a reasonable basis for use as GPS noise models. We have selected models that are commonly used or have been proposed as GPS noise models. We took into consideration not only the model's fit to the data, but also the complexity involved in parameter estimation and filtering. The models chosen for comparison are the independent, identically distributed Gaussian model, the Ornstein-Uhlenbeck process, the autoregressive processes, the moving average process and the mixed autoregressive, moving average model.

2.4.1 Independent Identically Distributed Gaussian

The independent identically distributed Gaussian (i.i.d. Gaussian) noise model is familiar as a standard model in filtering and signal processing. It is defined below.

$$X_t = \mu + n_t, \tag{2.13}$$

Where X_t is the measured variable with true value μ and noise term (n_t) has a Gaussian distribution with zero mean and standard deviation of σ, i.e. $n_t \sim \mathcal{N}(0, \sigma^2)$. The current measurement is independent of all previous measurements and $n_t \sim \mathcal{N}(0, \sigma^2)\ \forall\ t$ [35].

The i.i.d. Gaussian model has many aspects which make it extremely useful for filtering. The central limit theorem tells us that the sum of N independent random variables, each with well-defined expected values and variance, will tend to a Gaussian distribution [41].

Also i.i.d. Gaussian models are extremely easy to filter with. A Gaussian distribution is uniquely defined by its mean and covariance [41]. Therefore a system with M variables can be completely described as an array of means with length M and an $M \times M$ covariance matrix.

The Kalman filter, for example, uses an array of means and a covariance matrix for describing the current state and uncertainty. This means that for a linear system with i.i.d. Gaussian noise the Kalman filter propagates the exact conditional probability estimates [20].

However, as noted previously the i.i.d. Gaussian model has no long-term autocorrelation function, unlike what is observed in the empirical data. Therefore we expect that other models will fit the data better and allow improved filtering of the data.

Maximum Likelihood Estimation of i.i.d Gaussian noise

The log likelihood of i.i.d Gaussian noise is,

$$\ln(L) = -\frac{N}{2}\ln(2\pi) - \frac{N}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{N}[X_i - \mu]^2, \tag{2.14}$$

where μ and σ^2 are the mean and variance, respectively [34]. Therefore by taking the derivatives with respect to μ and σ^2 we see that the maximum likelihood estimates for an i.i.d Gaussian process are simply the sample mean, i.e.

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i,\ \hat{\sigma^2} = \frac{1}{N}\sum_{i=1}^{N}[X_i - \hat{\mu}]^2, \tag{2.15}$$

2.4.2 Ornstein-Uhlenbeck process

The Ornstein-Uhlenbeck process is a mean-reverting stochastic process, originally developed to describe the velocities of particles undergoing Brownian mo-

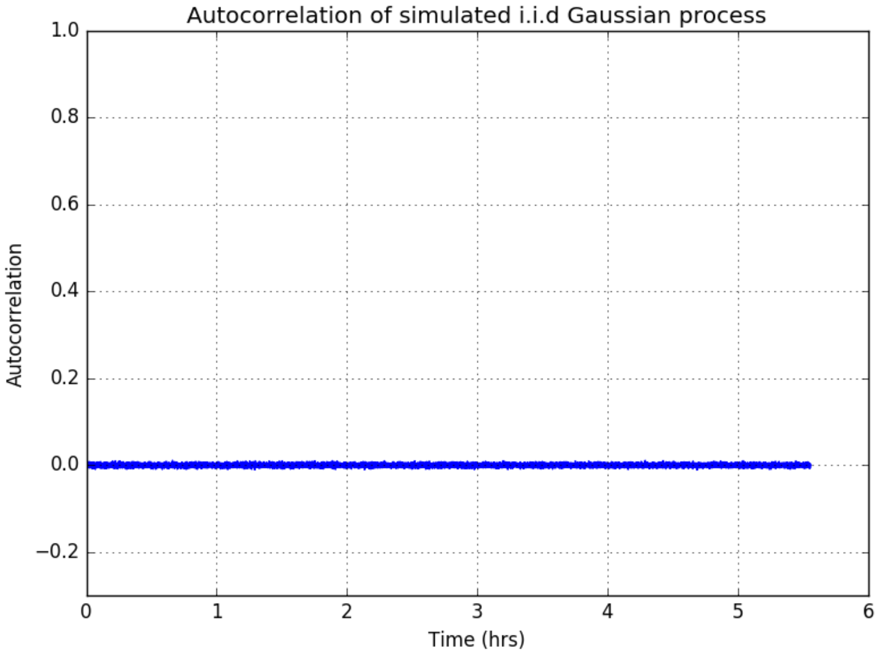


Figure 2.3: Autocorrelation of simulated independent, identically distributed Gaussian noise.

tion [42]. The Ornstein-Uhlenbeck process was originally proposed as a candidate noise model as it has a long-term autocorrelation function, similar to what we observed in the empirical data. Also, its mean-reverting behaviour is qualitatively to some of the behaviour that we observed.

An Ornstein-Uhlenbeck (O-U) process is defined by the stochastic differential equation

$$dx_t = \theta(\mu - x_t)dt + \sigma dW_t, \tag{2.16}$$

where θ, μ and σ are parameters of the process and W_t is the Wiener process [42].

$$W_t = W_t - W_0 \sim \mathcal{N}(0, t) \tag{2.17}$$

Therefore, is we transfer from the continuous case above, to the discrete case then Equation 2.16 becomes the following [42].

$$X_t = X_0 e^{-\theta t} + \mu \left(1 - e^{-\theta t}\right) + \sigma \int_0^t e^{\theta(s-t)} dW_s. \tag{2.18}$$

The maximum likelihood estimates for the Ornstein-Uhlenbeck process provides useful descriptions of the noise and its behaviour. The μ is the 'true' position of the device which we seek to find and which the process mean-reverts towards, the θ determines the autocorrelation function with the autocorrelation time proportional to $e^{-\theta dt}$ where dt is the sampling period of the GPS device. The σ parameter then determines the noise present between measurements.

As a candidate noise model for GPS noise the Ornstein-Uhlenbeck process has several desirable properties; it is mean-reverting, it can have a long term autocorrelation time and periodic structure in the autocorrelation function and its parameters are not determined by the sampling rate of the device.

We simulated an Ornstein-Uhlenbeck process with the Euler-Maruyama method [43] and then plotted its autocorrelation function, as seen in Figure 2.4. The similarities between autocorrelation functions of GPS data and a simulated Ornstein-Uhlenbeck process are easily identifiable in Figure 2.2.

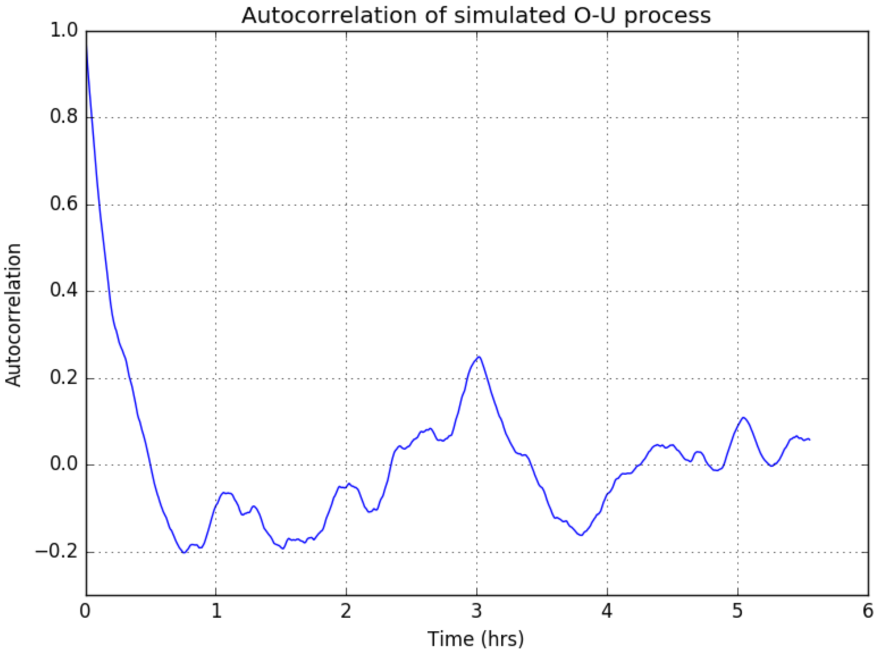


Figure 2.4: Autocorrelation of an Ornstein-Uhlenbeck process showing similar long-term structure to the GPS position errors.

Maximum Likelihood Estimation of O-U Process

The log likelihood function of an Ornstein-Uhlenbeck process is given below,

$$\ln(L) = \ln(p(X_0)) - \frac{(N-1)}{2}\ln(2\pi) - \frac{(N-1)}{2}\ln(\tilde{\sigma}^2) - \sum_{i=2}^{N}\frac{[x_i - \tilde{\mu}]^2}{2\tilde{\sigma}^2}, \tag{2.19}$$

where $\tilde{\sigma}^2$ and $\tilde{\mu}$ are defined as,

$$\tilde{\sigma}^2 = \frac{\sigma^2(1 - e^{-2\theta dt})}{2\theta} \quad \text{and} \quad \tilde{\mu} = X_{i-1}e^{-\theta dt} + \mu(1 - e^{-\theta dt}) \tag{2.20}$$

Ignoring the first component, which is only dependent on the probability of the first measurement, the parameters that maximize the log likelihood can be obtained explicitly by,

$\hat{\theta} = -dt^{-1}\log(\beta_1)$, $\hat{\mu} = \beta_2$ and $\hat{\sigma}^2 = 2\frac{\hat{\theta}\beta_3}{1-\beta_1^2}$. where

$$\beta_1 = \frac{\sum_{i=1}^{n} X_i X_{i-1} - n^{-1}\sum_{i=1}^{n} X_i \sum_{i=1}^{n} X_{i-1}}{\sum_{i=1}^{n} X_{i-1}^2 - n^{-1}\left(\sum_{i=1}^{n} X_{i-1}\right)^2} \tag{2.21}$$

$$\beta_2 = \frac{n^{-1}\sum_{i=1}^{n}(X_i - \beta_1 X_{i-1})}{1 - \beta_1} \tag{2.22}$$

$$\beta_3 = n^{-1}\sum_{i=1}^{n}[X_i - \beta_1 X_{i-1} - \beta_2(1 - \beta_1)]^2 \tag{2.23}$$

The O-U process is a continuous-time analogue of an autoregressive process of order 1 [44, 45]. We will now investigate the more general group of autoregressive processes.

2.4.3 Autoregressive processes

A Gaussian autoregressive process of order p (AR(p)) is defined as,

$$X_i = \phi_1 X_{i-1} + \phi_2 X_{i-2} + \cdots + \phi_p X_{i-p} + \epsilon_i, \tag{2.24}$$

where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, and the set $\{\phi_n\}$ are independent parameters. The process therefore has $p+2$ parameters. Due to the dependence on the previous measurement the process has an autocorrelation function that exhibits exponential decay and, like the O-U process, can exhibit periodic behaviour [34].

Because the autoregressive process is linear and only dependent on the previous p measurements, a system of linear equations with the measurements and parameters can be described and easily expressed as a matrix. The matrix may then be inverted to give the parameters of the AR processes. This process, known as solving the Yule-Walker equations, is a computationally efficient method of parameter estimation.

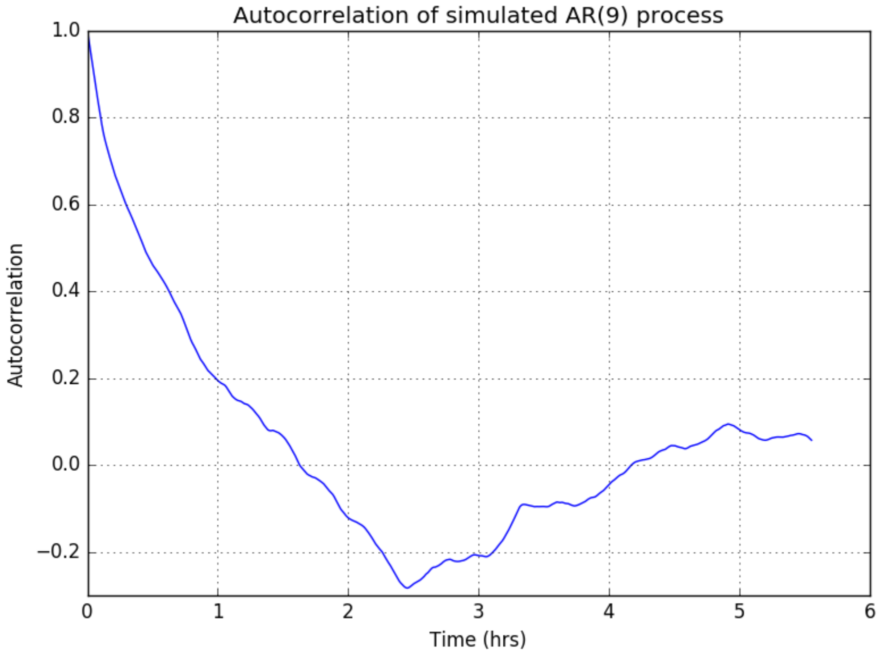


Figure 2.5: Autocorrelation of a simulated autoregressive process of order 9. The parameters used in this model were obtained by maximum likelihood estimation fit to empirical GPS data.

Maximum likelihood for autoregressive processes

The exact maximum likelihood function for the autoregressive process has some slight complications. However, if the maximum likelihood function is conditioned on the first p measurements the conditional likelihood is much easier to calculate.

The conditional likelihood of X_t given measurements $X_{t-1}, \ldots, X_{t-p}$ is given by,

$$f(X_t|X_{t-1}, \ldots, X_{t-p}, \phi's, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-(X_t - \mu - \sum_{i=t-p}^{t-1} \phi_i X_i)^2}{2\sigma^2}\right].$$

Therefore the conditional log likelihood of the set of data, ignoring measurements $1, 2, \ldots, p$ is given by:

$$\ln(L(\theta)) = \frac{-(N-p)}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=p}^{N} \left(X_t - \mu - \sum_{i=t-p}^{t-1} \phi_i X_i \right)^2 .$$

2.4.4 Moving-average processes

A Gaussian moving-average process of order q (MA(q)) is defined as,

$$X_i = \mu + \epsilon_i + \theta_1 \epsilon_{i-1} + \ldots + \theta_q \epsilon_{i-q},$$

where $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$ and the set $\{\theta_n\}$ are independent parameters. Thus the process has $q + 2$ parameters, and so a moving-average process of order zero reduces to a Gaussian process. Unfortunately because the lagged terms are not observable, maximum likelihood estimation for moving-average processes tends to be considerably more difficult than for autoregressive processes of comparable order, and analytical expressions for the MLE of MA parameters are not readily calculated and are usually performed by numerical optimization [34]. The simplest technique for parameter estimation of an MA(q) process is the Kalman filter [22]. While an impressive use of the Kalman filter, this approach is computationally slow relative to the solution of the Yule-Walker equations, seen in the autoregressive process.

It should be noted that unlike an O-U or AR process a moving-average process does not have a long term autocorrelation function. For a moving-average process of order q the autocorrelation function is zero after a lag of time qdt, where dt is the sampling period.

Therefore a pure moving-average process would require a cumbersomely large order to reproduce the autocorrelation functions that were observed in the data. However a combined autoregressive moving-average process would deal with this problem much more elegantly.

2.4.5 Mixed Autoregressive moving-average process

The mixed autoregressive moving-average (ARMA(p,q)) process is, as its name suggests, a linear combination of an autoregressive process of order p with a moving-average process of order q, shown below.

$$X_t = \mu + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \ldots + \theta_q \epsilon_q.$$

The ARMA process, predictably, has characteristics of both autoregressive and moving-average processes. Due to the autoregressive terms an ARMA process is able to have a non-zero autocorrelation time after lag pdt or qdt.

However, as it has the unobservable terms from the moving-average process, the MLE for the parameters must be done numerically. Therefore, we also need to run a Kalman filter through the data in order to maximise the likelihood function and so the estimation time is considerably longer than the autoregressive process alone.

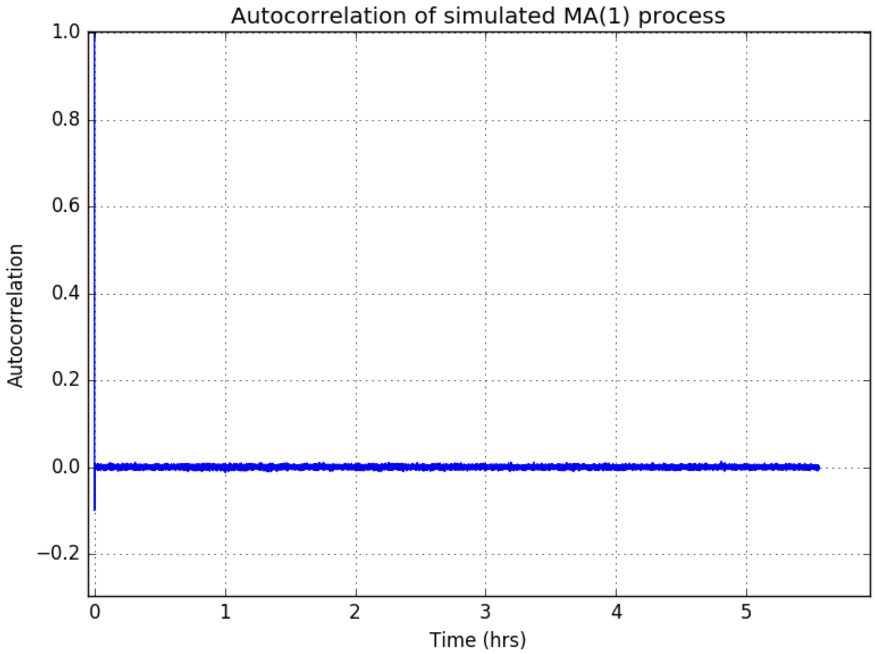


Figure 2.6: Autocorrelation of a simulated moving-average process of order 1. The parameters used in this model were obtained by maximum likelihood estimation fit to empirical GPS data.

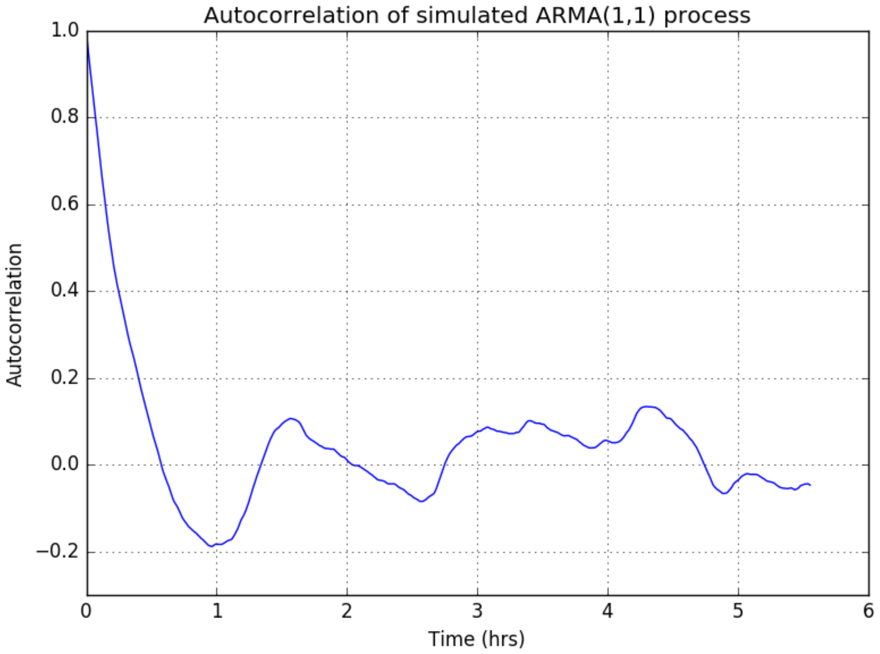


Figure 2.7: Autocorrelation of a simulated autoregressive, moving-average process of order (1,1). Due to the presence of the autoregressive terms the ARMA processes can display long term autocorrelation functions similar to those observed in the empirical data.

Chapter 3

Experimental Characterisation of Noise in GPS Positions

This chapter describes the collection of single frequency GPS data from several devices, at several different locations. The data gathered from the field was then analysed in the context of the noise models presented in Chapter 2. The candidate models were then ranked using the Akaike Information Criterion (AIC), a model selection criterion that provides a quantitative measure of the relative goodness of fit of a set of given statistical models.

3.1 Aims

We sought to investigate the relative performance of several statistical noise models by modelling the noise observed in single frequency GPS data. The most appropriate model was chosen from the set of proposed modes by its AIC value, as well as it general applicability towards GPS noise modelling.

To determine the most appropriate model for GPS noise, data was collected from GPS units positioned at a number of stationary points for an extended period of time. A commercial GlobalSat BU-353S4 GPS unit was used along with four tag-style GPS units to ensure that it was the errors in the GPS signal and not errors related to the device. The tag-style GPS units were developed by the Electronics Research Group at the University of Otago and have a different antenna to the commercially available GPS unit, which resulted in a significantly larger variance than the commercial device [46, 47].

3.2 Methods

One of the factors that causes degradation of GPS accuracy is the geometric distribution of satellites. For GPS to function effectively the satellites in the sky should be widely distributed rather than all satellites arranged in one section of the sky. This is why the altitude errors in GPS measurements are typically much larger than the errors in either the latitude or longitude directions [31]. Because GPS works on line of sight the satellites being used are all above the unit. Whereas for longitude there could be a broad arrangement of satellites to the east and west of the unit. However, the GPS constellation operates at an inclination of approximately 55°, i.e. at a latitude of more than 55° (north or south) there are only satellites in the direction of the Equator from your position.

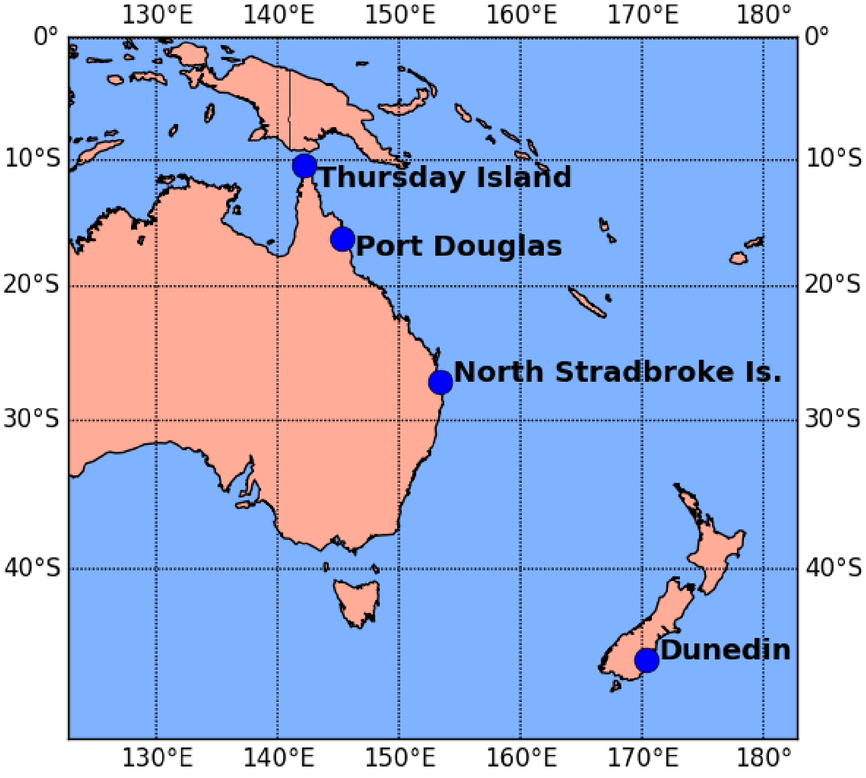


Figure 3.1: The locations where GPS measurements were made. The University of Otago, Dunedin (45.518°S, 170.308°E) (1). North Stradbroke Island (27.253°S, 153.308°E) (2). Port Douglas (16.297°S, 145.277°E) (3). Thursday Island (10.348°S, 142.133°E) (4).

Our initial measurements were made from the University of Otago, which is at a latitude of approximately 45.52° South. To ensure that our noise model was not biased by an imperfect satellite arrangement, separate measurements were made at latitudes of 27.25°S (North Stradbroke Island), 16.30°S (Port Douglas) and 10.35°S (Thursday Island).

In Figure 3.2 we can see the GPS devices used for this experiment. The tag style GPS device is shown in Figure 3.2 (a) prior to having the antenna attached, with a New Zealand 10 cent coin used for scale. The commercial GlobalSat GPS device is also shown in Figure 3.2 (b).

(a) (b)

Figure 3.2: Figure (a) shows one of the 4 tag style GPS units developed by the Electronics Research Group at the University of Otago, prior to having the antenna attached [46]. Figure (b) shows the GlobalSat BU-353S4 GPS unit with the USB that was used to communicate with the Raspberry Pi. Image taken from [48].

Each measurement period was for approximately 48 hours. Measurements were made with one GlobalSat BU-353S4 GPS unit on the L1 frequency, sampling once a second and with four tag-style GPS units with lower-quality aerials, also using the L1 frequency, sampling at an average of 18 seconds per fix. However the tag-style GPS units take fixes from a warm start instead of a hot start and so while the average sampling period is around 18 seconds it can be considerably longer, as shown in Figure 3.5.

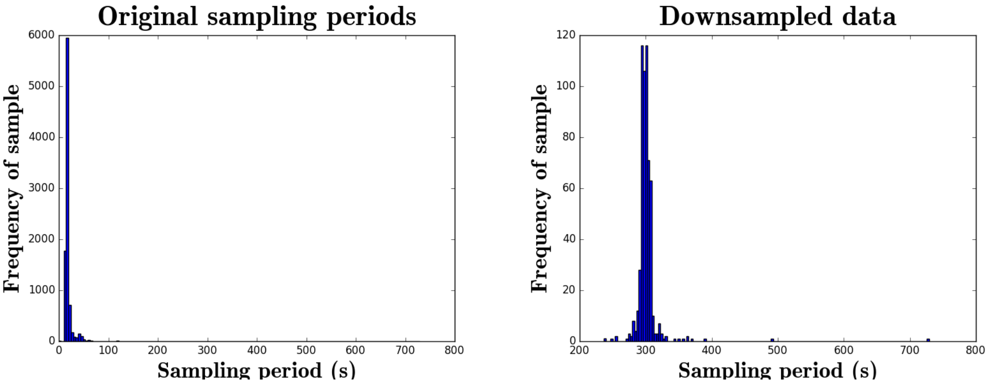
The O-U process and i.i.d. Gaussian processes can easily handle differing sampling periods. The O-U parameters are independent of the sampling period, as are the i.i.d. Gaussian parameters. However, the AR, MA, and ARMA process parameters have an implicit dependence on the sampling period. Given this, the tag-style devices were downsampled to a 5 minute sampling period so that any deviation in sampling period would be small in comparison to the sampling period. The downsampling was achieved by choosing the n^{th} data point to be the closest to 5 minutes from the $(n-1)^{th}$ data point, results demonstrating this downsampling can be seen in Figure 3.5.

In Dunedin the GPS units were mounted on the roof of the Physics Department of the University of Otago. The GlobalSat BU-353S4 GPS unit has a magnetised base which allowed easy mounting on the metal plate shown in Figure 3.6. This position allowed excellent sky view while ensuring that the equipment was protected

(a) (b)

Figure 3.3: Figure (a) shows the stand for the GlobalSat BU-353S4 GPS unit with the higher-quality aerial. The base is magnetised and a steel bolt threaded in the stand allows it to stay securely mounted. The case seen at the bottom of the stand contains a Raspberry Pi for data collection and storage. The aerial seen on the case is from a board attached to the Pi to receive telemetry signals from the tag style units. Figure (b) shows the set of 4 tag style GPS units developed by the Electronics Research Group at the University of Otago.

Figure 3.4: For the Australian locations two-man tent was used to protect the GPS equipment from the elements. This photograph was taken from the site of the North Stradbroke Island test.



(a) (b)

Figure 3.5: Sampling periods from one of the tag-style GPS units. Data was downsampled to reduce the relative error in sampling periods present within data sets.

from the mercurial Dunedin weather. Every data set taken by the GlobalSat unit in Dunedin was taken from the plate shown in Figure 3.6 and so the positions for all are the same ±5 cm.

Figure 3.6: All GPS measurements from the GlobalSat BU-353S4 GPS unit were taken from this mount beneath a large skylight on top of the Physics Building at the University of Otago. The GPS unit pictured is not the GlobalSat BU-353S4 but another GPS unit for a separate experiment.

On North Stradbroke Island and in Port Douglas measurements were made from local campsites. On Thursday Island the local primary school principal kindly let us set up our equipment on their basketball court, as this was during the school holidays. In Figure 3.4 we see the setup for the field experiments carried out at all three Australian test sites.

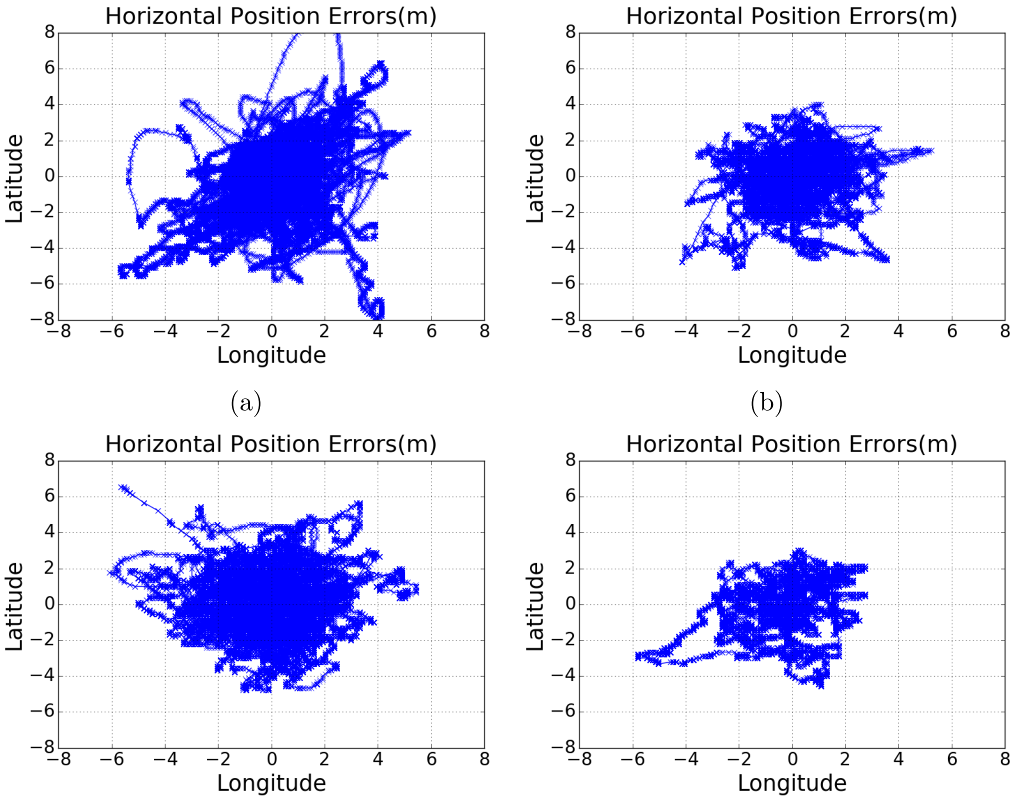
The experimental setup for the GPS devices is shown in Figure 3.3. A Raspberry Pi B+ was used for data collection and storage. The GlobalSat BU-353S4 GPS unit communicated directly with the Raspberry Pi via USB, the code for which was written by both the author and supervisor, Tim Molteno. The tag-style GPS units communicated via radio to a telemetry board that was mounted on the Raspberry Pi. The stands were based around simple PVC piping and 3D printed brackets which allowed for a lightweight and easily transportable experimental setup.

Unfortunately the base bracket for the tag-style GPS units broke while being assembled on North Stradbroke Island and so an improvised setup had to be constructed. This was achieved by hanging the top of the tag-stand from the ceiling of the tent, as shown in Figure 3.7.

Figure 3.7: Unfortunately the stand for the four tag-style GPS units broke while on North Stradbroke Island. In order to be able to store the equipment required and not risk accidental damage to the tags it was decided to suspend them as shown.

3.3 Results

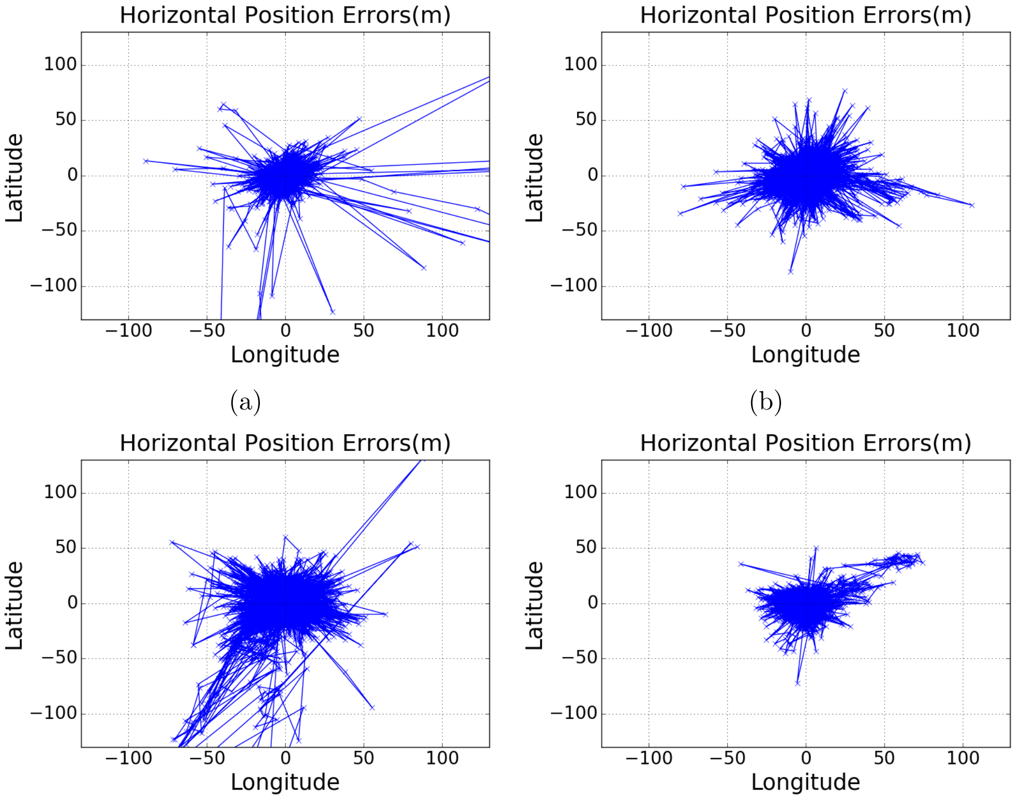
In Figure 3.8 we see the horizontal position errors reported by the GlobalSat commercial GPS unit, from Dunedin (a), North Stradbroke Island (b), Port Douglas (c), and Thursday Island (d). Then in Figure 3.9 we see the horizontal position errors reported by one of the tag style GPS units from Dunedin (a), North Stradbroke Island (b), Port Douglas (c), and Thursday Island (d).



(a) (b) (c) (d)

Figure 3.8: Horizontal positions from Dundedin (a), North Stradbroke Island (b), Port Douglas (c) and Thursday Island (d). Note how the position outliers tend to go on ‘walks’, where the current position estimate does not immediately go back towards the mean, but instead stays considerably removed from the ‘true’ position.

In Figure 3.9 we see the horizontal errors measured by one of the tag style GPS units from all four locations. As you can see the errors reported by the tag style GPS units is considerably larger than that reported by the commercial GlobalSat device. The tag style GPS devices were developed by the Electronics Research Group to be used on birds. Therefore weight was prioritised over accuracy, sacrificing a better quality aerial for weight savings.



(a)

(b)

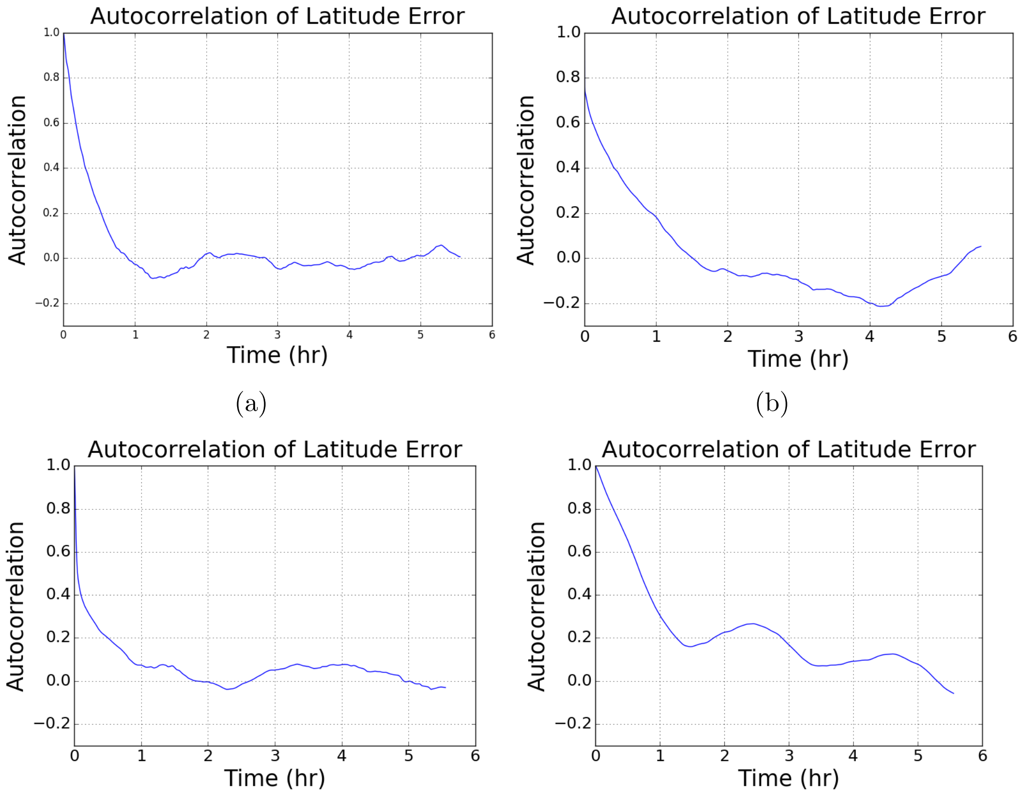
(c)

(d)

Figure 3.9: Horizontal position errors reported by the same tag style GPS unit from Dundedin (a), North Stradbroke Island (b), Port Douglas (c) and Thursday Island (d). A similar error behaviour is observed, as measurements that are far removed from the long term mean often stay outside the central point for several measurements in a row.

3.4 Analysis

The following tables are laid out with the AIC values for the respective positions (latitude, longitude, altitude) for the data taken from the various sites. The data taken from the GlobalSat BU-S353 GPS unit with the higher quality aerial is titled with ‘GlobalSat’ while the data gathered from the tag-style GPS units with lower quality aerials is title ‘Tags’ for simplicity. The top acronyms denote the noise model being dealt with, i.e. i.i.d. Gaussian (Gauss), Ornstein-Uhlenbeck process (O-U), autoregressive process of order p (AR(p)), moving-average process of order q (MA(q))



(a) (b)

(c) (d)

Figure 3.10: Autocorrelations for latitude data from Dundedin (a), North Stradbroke Island (b), Port Douglas (c) and Thursday Island (d).

and mixed autoregressive moving-average process of order p, q (ARMA(p, q)). Due to the often large differences in AIC values instead of the Akaike weights w_i the difference between the minimum AIC value and model AIC value is given as ΔAIC.

$$\Delta AIC_i = AIC_i - AIC_{min}, \tag{3.1}$$

We made use of the *statsmodels* module in Python to calculate the parameters for the AR, MA, and ARMA models. Occasionally the statistics package was unable to model either the moving-average (MA) or mixed autoregressive moving-average (ARMA) processes due to the moving-average parameters not being invertible. In these cases we have given the highest order processes available and denoted non-invertible processes as DNC.

For the sake of brevity the AIC values of the tag-style GPS units have been averaged over all of the tags for that location. See Appendix A.5 for the individual tag-style AIC values. For readability the GlobalSat AIC values have been rounded to the nearest integer.

Dunedin (45.52°S) data set

GlobalSat	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	1275060	-1697123	-1718883	789357	DNC	-1724844
Lat ΔAIC	2999904	27721	5961	2514201	DNC	0
Lon AIC	1099902	-1879884	-1895003	614421	DNC	-1902929
Lon ΔAIC	3002832	23045	7926	2517350	DNC	0
Alt AIC	1901656	-1462782	-1535844	1412690	DNC	-1542135
Alt ΔAIC	3443791	79353	6290	2954825	DNC	0

Tags	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	11189.73	11003.24	10912.11	11052.71	10981.89	10969.12
Lat ΔAIC	277.62	91.13	0	140.60	69.78	57.01
Lon AIC	11003.80	10827.14	10724.15	10882.31	10798.22	10793.47 (2,2)
Lon ΔAIC	279.65	102.99	0	158.16	74.07	69.32
Alt AIC	12219.62	11998.43	11901.11	12066.52	11990.96	11821.81
Alt ΔAIC	397.81	176.62	79.3	244.71	169.15	0

North Stradbroke Island (27.25°S) data set

GlobalSat	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(2,2)
Lat AIC	616991	4099	-846881	417772	160425	-54584
Lat ΔAIC	1463872	850980	0	1264653	1007306	792297
Lon AIC	552469	5140	-847589	358963	138410	-96669
Lon ΔAIC	1400058	852730	0	1206552	985999	750920
Alt AIC	991469	401359	-638245	783784	549365	(3,3) 272511
Alt ΔAIC	1629714	1039604	0	1422030	1187610	(3,3) 910756

Below is the North Stradbroke Island data set from GPS units with lower quality aerials. Recall that the model with the lowest AIC values is the best fit to the data from the candidate set. Due to fewer data points and consequently smaller AIC values the tag-style data is kept as a decimal.

Tags	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	4524.70	4454.52	4343.39	4487.33	4426.08	4342.60
Lat ΔAIC	182.10	111.92	0.80	144.74	83.48	0
Lon AIC	4337.74	4295.58	4225.97	4312.76	4292.28	4271.81
Lon ΔAIC	111.77	69.61	0	86.79	66.31	45.83
Alt AIC	5085.63	5045.22	4961.13	5062.49	5045.76	5025.82
Alt ΔAIC	124.50	84.10	0	101.36	84.63	64.69

Port Douglas (16.30°S) data set

Note that the higher order MA and ARMA models were not invertible for the GlobalSat GPS unit and so the highest terms were 1 and 1,1, respectively.

GlobalSat	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(1,1)
Lat AIC	1080721	-1031940	-1104629	749275	DNC	-1048908
Lat ΔAIC	2185349	72688	0	1853903	DNC	55721
Lon AIC	1254838	-1016929	(8) -1112235	927860	352311	-924883
Lon ΔAIC	2367073	95307	0	2040095	1464547	187352
Alt AIC	1445244	-659902	-844480	1112076	DNC	-722318
Alt ΔAIC	2289725	184578	0	1956556	DNC	122162

Tags	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	6322.39	6138.42	6016.16	6199.74	6116.34	6305.13
Lat ΔAIC	306.23	122.26	0	183.58	100.18	288.97
Lon AIC	6196.31	6085.74	6010.90	6120.54	6090.74	5952.81
Lon ΔAIC	243.50	132.93	58.09	167.73	137.93	0
Alt AIC	6998.93	6949.83	6864.24	6969.55	6953.60	6928.78
Alt ΔAIC	134.69	85.60	0	105.31	89.37	64.55

Thursday Island (10.35°S) data set

Unfortunately power was lost for around three minutes on Thursday Island and so the data set for the GlobalSat GPS unit from Thursday Island has been split into two separate data sets. The tag-style GPS units appear to have retained enough charge in their capacitors to be unaffected by this short loss of power. This is the first data set for the GlobalSat GPS unit.

GlobalSat	Gauss	O-U	AR(9)	MA(1)	ARMA(2,2)
Lat AIC	292988	-436143	-436616	184240	-436768
Lat ΔAIC	729757	625	152	621008	0
Lon AIC	284169	-414991	-415461	175645	(3,3) -415917
Lon ΔAIC	700085	925	456	591561	(3,3) 0
Alt AIC	471291	-329711	-332829	362145	-333476
Alt ΔAIC	804767	3765	647	695621	0

And the second data set from Thursday Island. As in the data set from Port Douglas the parameters for higher order MA and ARMA processes were not invertible and so the highest order terms obtained were 1 and 1,1, respectively.

GlobalSat	Gauss	O-U	AR(9)	MA(1)	ARMA(1,1)
Lat AIC	368278	-390082	-419119	240593	-391015
Lat ΔAIC	787397	29037	0	659712	28104
Lon AIC	490408	-212528	-358983	369403	-201441
Lon ΔAIC	849391	146455	0	728386	157541
Alt AIC	616549	-107027	-151204	488436	-106982
Alt ΔAIC	767753	44177	0	639640	44222

Data set from low-quality aerial GPS units. One of the tags was damaged in transport and so the data set from Thursday Island is averaged over three GPS units, not four as is the case for the other data sets.

Tags	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	3730.57	3559.66	3489.74	3616.87	3562.08	3610.96
Lat ΔAIC	240.84	69.92	0	127.12	72.34	121.21
Lon AIC	3729.26	3579.87	3495.36	3640.82	3573.10	3296.72
Lon ΔAIC	432.53	283.15	198.64	344.10	276.38	0
Alt AIC	4782.49	4630.38	4522.52	4690.94	4631.02	4586.51
Alt ΔAIC	259.96	107.86	0	168.42	108.49	63.99

3.5 Conclusions of GPS noise experiment

It is clear from Figure 3.10 that the long-term autocorrelation times observed in Dunedin are not a result of the poor satellite arrangement at higher latitudes. Of these the autocorrelation time of the data gathered at Thursday Island has a slightly different pattern to those observed at the other locations. This could be because of a temporary loss of power experienced at Thursday Island which essentially forced the data to be split into two sets of data of around 24 hours each. Or, alternatively it could be an affect of the local geometry that led to reflected signals which would affect the GPS data.

What we see in the data is that there is a general trend that appears across the regions where the noise model with the highest AIC (and so the worst fit to the data) is the i.i.d. Gaussian. After that typically is the lower-order moving-average processes, followed by the Ornstein-Uhlenbeck model and then either the higher order autoregressive models or the mixed autoregressive moving-average model.

It is worth noting that getting the parameters that maximise the likelihoods for the Ornstein-Uhlenbeck and autoregressive models is a trivial matter where either the analytical solution is known or it is simply a case of inverting a matrix which can be done numerically and very quickly. Conversely estimating the parameters for the moving-average and mixed autoregressive moving-average models is considerably more involved and typically requires running a Kalman filter over the data set with a given set of parameters, calculating the log likelihood of that particular case and then doing the same thing again with a slightly different set of parameters until the log likelihood is maximised. This approach is much more computationally expensive. Also if the process is non-stationary then the moving-average parameters are not invertible and further measures must be taken to solve for the MLE parameters. This is why the autoregressive models are taken up to ninth order whereas, at most the MA and ARMA models are only taken up to third order and even then we experienced invertibility problems.

3.6 Proposed Noise Models

In this experiment we looked at data from two different types of GPS devices. The data gathered from these two devices was qualitatively similar. Across both GPS types and all locations the i.i.d. Gaussian model was the worst fit to the data. The moving average models tended to be the next worst fit, followed by either the autoregressive processes (including the discretized O-U process) or the mixed autoregressive moving average processes.

If we just look at the data from the commercial GlobalSat GPS device we see that the autoregressive processes are a much better fit to the data than the moving average processes. Parameter estimation for the autoregressive processes is also considerably more computationally efficient than for the moving-average processes.

The higher order AR processes have lower AIC values, indicating that, from the point of view of modeling the data precisely it is better to have a higher order AR process as the likelihood maximization outweighs the increased complexity due to the extra terms.

However, for the less accurate tag-style GPS units, the higher order AR processes are only slightly more likely than the AR(1) process. If one is considering using the AR(1) process we would advise using the discretized form of the Ornstein-Uhlenbeck process rather than a generalized AR(1) process. This has several advantages over simply estimating the autoregressive parameter ϕ. For one, the sampling period of the device is explicit in the formulation of the Ornstein-Uhlenbeck process and so for cases when there is a non-constant sampling period this is clearly more useful. Also the other parameters of the O-U process give clear insight into the characteristics of the model. The θ parameter describes the rate at which random measurements will revert towards the mean μ while the σ parameter indicates the amount of noise present between each measurement.

Chapter 4

GPS Filtering

In the previous chapter we analysed GPS data and compared the performance of several different noise models. From the data we determined that the worst model in the candidate set was the independent, identically distributed Gaussian (i.i.d. Gaussian) noise model. The higher order autoregressive and mixed autoregressive, moving-average processes tended to provide the best fit to the data. Of the autoregressive and mixed autoregressive, moving-average processes the solely autoregressive processes provide a much simple parameter estimation problem, and so for this chapter we will only focus on filtering with i.i.d. Gaussian and autoregressive noise models.

However one of the benefits of using the i.i.d. Gaussian noise model is that it is easily integrated into filtering algorithms such as the Kalman filter. In this chapter we derive the equations required to integrate a discretized Ornstein-Uhlenbeck noise model into the Kalman filter algorithm. We demonstrate the filtering with the O-U noise model and demonstrate its advantages over simply using a generalized AR(1) noise model.

4.1 Filtering with independent Gaussian noise

While the i.i.d. Gaussian noise model has been shown to fit the data less well than other noise models it is a very simple model to filter with. Importantly, if using a linear model with i.i.d. Gaussian noise then the standard Kalman filter will provide the exact posterior to the distribution, i.e. it will filter as well as possible given the amount of noise present.

We will begin by defining exactly what we mean when we say an i.i.d. Gaussian noise model.

$$x_t = \mu + \sigma \int_0^t dW_s. \tag{4.1}$$

Equation (4.1) gives the continuous version of the i.i.d. Gaussian noise model where x_t is the measurement at time t, μ is the true value of the variable, W_t is the Wiener process and σ is the standard deviation of the noise. This simplifies to the commonly given form,

$$x_t \sim \mathcal{N}\left(\mu, \sigma^2\right), \tag{4.2}$$

Indicating that the measurement x_t is normally distributed with mean μ and variance σ^2.

As discussed in Chapter 1 the Kalman filter is a very computationally efficient, recursive algorithm and so real time state estimation with a linear model and i.i.d. Gaussian noise is trivial to implement.

All that is required for a Kalman filter is a linear system model, measurement model, and expressions for the noise and uncertainty in the initial estimate. In the case of a three-dimensional i.i.d. Gaussian model this is simply,

$$F(k) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad H(k) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.3}$$

$$Q(k) = \begin{bmatrix} \sigma_{v_1}^2 & 0 & 0 \\ 0 & \sigma_{v_2}^2 & 0 \\ 0 & 0 & \sigma_{v_3}^2 \end{bmatrix}, \qquad R(k) = \begin{bmatrix} \sigma_{w_1}^2 & 0 & 0 \\ 0 & \sigma_{w_2}^2 & 0 \\ 0 & 0 & \sigma_{w_3}^2 \end{bmatrix}, \tag{4.4}$$

where $F(k)$ is the state transition matrix, $H(k)$ is the measurement matrix, and $Q(k)$ and $R(k)$ are the system and measurement noise covariances, respectively. In Figure 4.1 we see the results of filtering GPS data with an i.i.d. Gaussian noise model.

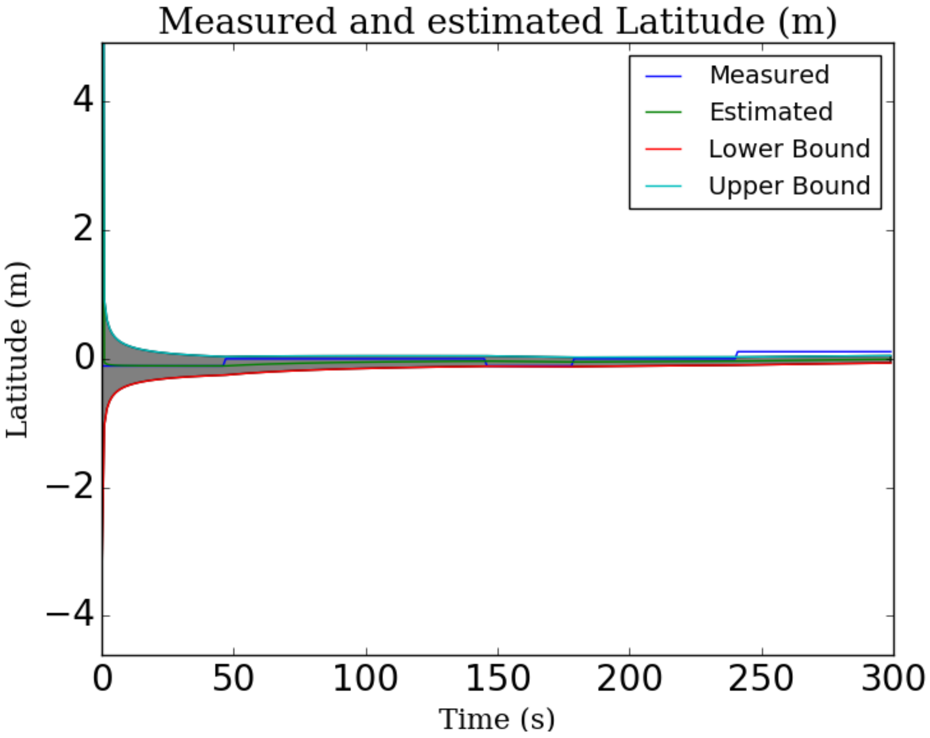


Figure 4.1: Above shows the results of filtering latitude measurements with an i.i.d. Gaussian noise model. The estimate is plotted in green and shown within a 95% confidence interval.

4.2 Filtering with Ornstein-Uhlenbeck noise

As mentioned previously an Ornstein-Uhlenbeck process is defined by the stochastic differential equation,

$$dx_t = \theta(\mu - x_t)dt + \sigma dW_t, \tag{4.5}$$

where μ is the long-term average of x_t, θ is a parameter of the model, and W_t is the Wiener process.

This then implies that σ from above is the standard deviation of the normally distributed noise in the O-U process.

The solution of the Ornstein-Uhlenbeck process is as follows [42],

$$x_t = x_0 e^{-\theta t} + \mu \left(1 - e^{-\theta t}\right) + \sigma \int_0^t e^{\theta(s-t)} dW_s, \tag{4.6}$$

where x_0 is the previous state, x_t is the state at time t, and s is the time index for the integral (t having already been used to represent the final time. From Equation

4.6 we see that the process is normally distributed with expected value and variance as given below.

$$\mathrm{E}[x_t] = x_0 e^{-\theta t} + \mu \left(1 - e^{-\theta t}\right), \quad \mathrm{var}[x_t] = \frac{\sigma^2}{2\theta} \left(1 - e^{-2\theta t}\right), \tag{4.7}$$

i.e.

$$x_t \sim \mathcal{N}\left(x_0 e^{-\theta t} + \mu(1 - e^{-\theta t}), \frac{\sigma^2}{2\theta}(1 - e^{-2\theta t})\right), \tag{4.8}$$

We can then easily move from the continuous case to the discrete case. If we let dt be the sampling period of the discrete case, then we get the following,

$$\mathrm{E}[x_i] = x_{i-1} e^{-\theta dt} + \mu \left(1 - e^{-\theta dt}\right), \quad \mathrm{var}[x_i] = \frac{\sigma^2}{2\theta} \left(1 - e^{-2\theta dt}\right), \tag{4.9}$$

In the case of modeling GPS noise as an Ornstein-Uhlenbeck process μ, the long-term mean is obviously the most important property. If we assume that the device is stationary i.e. μ does not change in time then we can re-write our measurement in terms of 'true' position, μ, and a noise term n_t which has O-U noise, i.e.

$$x_i = \mu + n_i, \tag{4.10}$$

where,

$$n_i \sim \mathcal{N}\left(e^{-\theta dt}(x_{i-1} - \mu), \frac{\sigma^2}{2\theta}\left(1 - e^{-2\theta dt}\right)\right), \tag{4.11}$$

If we substitute $(x_{i-1} - \mu)$ for n_{i-1}, then we get the following pleasant expression for the distribution of the i^{th} noise variable term.

$$n_i \sim \mathcal{N}\left(n_{i-1} e^{-\theta dt}, \frac{\sigma^2}{2\theta}\left(1 - e^{-2\theta dt}\right)\right), \tag{4.12}$$

The Kalman filter is a versatile algorithm that combines sequential measurements and provides asymptotically optimal estimates of the state while minimizing the state covariance. One of the convenient aspects of the Kalman filter is that it is a recursive algorithm and so all of the information from the previous measurements is contained in the current state estimate, therefore significantly easing the computational load over each iteration.

While filtering with the O-U model will require expanding the state vector for Kalman filtering we need only include a single extra noise variable for each O-U filtered variable. We also have a measurement model and propagation model ready from Equation 4.12. The state propagation matrix ($F(k)$) to filter an O-U variable is given below, along with the measurement matrix ($H(k)$).

$$F(k) = \begin{bmatrix} 1 & 0 \\ 0 & \exp(-\theta dt) \end{bmatrix}, \quad H(k) = \begin{bmatrix} 1 & 1 \end{bmatrix}, \tag{4.13}$$

To implement a standard Kalman filter we now only need a state vector ($\hat{\mathbf{x}}(k)$) and covariance matrices for the measurement ($R(k)$) and process noises($Q(k)$). The only slight complication is that for a system where the measurement device is stationary there is virtually no process noise present in the system. However to construct a functioning Kalman filter there exists some computational reasons against having the matrix $Q(k)$ set to exactly zero. Fortunately setting $Q(k)$ to some small value, e.g. 10^{-6}, deals with these satisfactorily. This then gives us the state vector and covariance matrices as follows.

$$\hat{\mathbf{x}}(k) = \begin{bmatrix} \hat{\mu} \\ n(k) \end{bmatrix}, \quad R(k) = \begin{bmatrix} \frac{\sigma^2}{2\theta}\left(1 - e^{-2\theta dt}\right) \end{bmatrix}, \quad Q(k) \approx [0], \tag{4.14}$$

Note that the dependence on the sampling period is explicit in the descriptions of the state propagation and measurement matrices ($F(k)$ and $R(k)$, respectively). Therefore the sampling period need not be constant. As we saw in the tag-style GPS units some GPS devices have difficulty with constant sampling periods when taking data from a warm start. This would present difficulties with some noise models as the parameters involved can have an implicit time dependence, as is the case for the autoregressive and moving average models [34]. By having an explicitly defined time dependence the Ornstein-Uhlenbeck allows this to take into account the effect of longer or shorter time periods and allows the filter to adjust automatically.

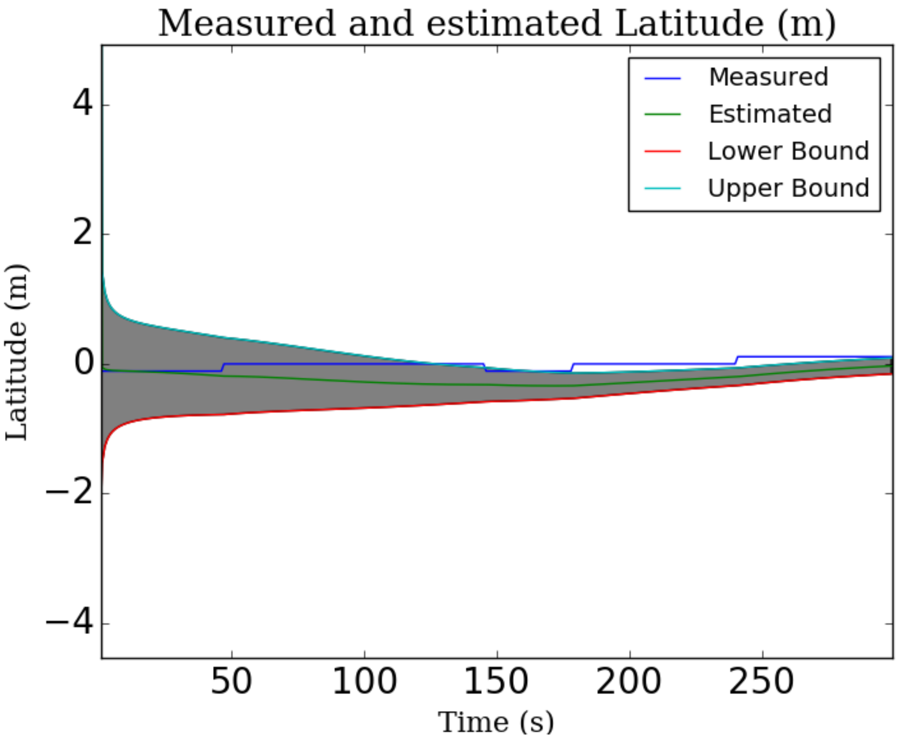


Figure 4.2: Above shows the results of filtering latitude measurements with an Ornstein-Uhlenbeck noise model. As in Figure 4.1 the estimate is plotted in green and shown within a 95% confidence interval.

4.3 Time-varying mean

Our derivation for a stationary mean is useful when looking at long time series of stationary GPS data. Inconveniently most devices that use GPS do so precisely because the position of those devices is *not* stationary. Fortunately we can extend our derivation from Section 4.2 to a system with a time-varying mean.

Let μ_i be the long-term mean of position i. Substituting this back into Equation 4.10 we then get the following.

$$x_i = \mu_i + n_i. \tag{4.15}$$

To make things easier on ourselves we'll assume that the μ_i term is constant over the sampling period so we still have the same solution as seen in Equation 4.6, just with μ replaced by μ_i, i.e.

$$x_i = x_{i-1}e^{-\theta dt} + \mu_i\left(1 - e^{-\theta dt}\right) + \sigma\int_0^{dt} e^{\theta(s-t)}dW_s, \tag{4.16}$$

just as with the stationary case our n_i is the noise variable which is normally distributed as shown below.

$$n_i \sim \mathcal{N}\left(\left(x_{i-1}-\mu_i\right)e^{-\theta dt}, \frac{\sigma^2}{2\theta}\left(1-e^{-2\theta dt}\right)\right). \tag{4.17}$$

Substituting in $x_{i-1} = \mu_{i-1} + n_{i-1}$ we then get the following.

$$n_i \sim \mathcal{N}\left(\left(n_{i-1}-\left(\mu_i-\mu_{i-1}\right)\right)e^{-\theta dt}, \frac{\sigma^2}{2\theta}\left(1-e^{-2\theta dt}\right)\right). \tag{4.18}$$

Most GPS devices are able to report velocity and heading data as well as position data. Therefore if we assume that the velocity v_i is approximately constant between data points then we have,

$$v_i dt \approx \mu_{i+1} - \mu_i, \tag{4.19}$$

which we can then substitute back into Equation 4.18 to get an expression for the distribution of the noise term which does not depend on any unknown information.

$$n_i \sim \mathcal{N}\left(\left(n_{i-1}-v_{i-1}dt\right)e^{-\theta dt}, \frac{\sigma^2}{2\theta}\left(1-e^{-2\theta dt}\right)\right). \tag{4.20}$$

Therefore our noise term distribution is only dependent on the previous state, so our state vector need not grow any larger (i.e. augment it further with other noise terms). So we would have the following state propagation ($F(k)$) and measurement ($H(k)$) matrices.

$$F(k) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \exp(-\theta dt) & -dt\exp(-\theta dt) \\ 0 & 0 & 1 \end{bmatrix}, \quad H(k) = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.21}$$

Which then, correspond to the state vector ($\hat{\mathbf{x}}(k)$), and the process ($Q(k)$) and measurement ($R(k)$) covariance matrices.

$$\hat{\mathbf{x}}(k) = \begin{bmatrix} \hat{\mu} \\ n(k) \\ v \end{bmatrix}, \quad R(k) = \begin{bmatrix} \frac{\sigma^2}{2\theta}\left(1-e^{-2\theta dt}\right) & 0 \\ 0 & \sigma_v^2 \end{bmatrix}, \quad Q(k) \approx \begin{bmatrix} 0 & 0 \\ 0 & \sigma_a^2 dt \end{bmatrix}, \tag{4.22}$$

Where v is the velocity, σ_v^2 is the variance in the velocity measurements, and σ_a^2 is the variance in the acceleration of the object.

Chapter 5

Inertial Navigation

This chapter describes the process of inertial navigation and how it is related to this project. A one-dimensional example is introduced in Section 5.1, wherein we demonstrate the general properties of inertial navigation and perform a one dimensional example of sensor fusion using position and accelerometer measurements. In Section 5.2 we introduce the reference frames to be used in subsequent chapters. We also address the rotation formalisms for transforming between those reference frames in Section 5.3.

Firstly we explain what we mean by inertial navigation. The term incorporates many different techniques and sensors, however, the fundamental principle is obtaining a state estimate by integrating the output of a sensor, often referred to as an inertial measurement unit (IMU) [49]. An example could be estimating the distance travelled by integrating the speed. Another example could be estimating the angle by integrating gyroscopic measurements. Fundamentally though, all of these processes distill to,

$$x(t) = x_0 + \int_0^t v(t')\mathrm{d}t', \tag{5.1}$$

where $x(t)$ is the desired estimate, obtained by integrating some known variable $v(t)$. Below we assume that $v(t)$ is a linear combination of the true value, $v_{\text{true}}(t)$, and some error, $\epsilon(t)$.

$$\begin{aligned} v(t) &= v_{\text{true}}(t) + \epsilon(t), \\ x(t) &= x_0 + \int_0^t v_{\text{true}}(s)\mathrm{d}s + \int_0^t \epsilon(s)\mathrm{d}s, \\ x(t) &= x_{\text{true}}(t) + \int_0^t \epsilon(s)\mathrm{d}s. \end{aligned} \tag{5.2}$$

It is clear from the equations above that as the integration time (t) increases our error in the desired variable ($x(t)$) necessarily accumulates due to the integrated error term ($\epsilon(t)$). Therefore any system that exclusively used inertial navigation would require highly accurate data in order to accurately estimate the desired variable. This is not to say that such a system is impossible, indeed the inertial measurement units on commercial aircraft can be accurate to within 0.6 nautical miles/hour [50].

5.1 One-dimensional Inertial Navigation

As an illustrative example of inertial navigation we consider the problem of height estimation for an elevator given some accelerometer measurements. If we assume that the accelerometer is fixed in the elevator's frame of reference then we can consider the problem of height estimation as a simple, one-dimensional inertial navigation problem. If the distances between floors are known then an easily accessible test-bench is available for testing the performance of different inertial navigation algorithms.

While there may appear to be little practical use in this problem it serves as a useful test-bench for inertial navigation systems, and clearly demonstrates some of the fundamental concepts of inertial navigation. Such a system could be easily implemented into a standard undergraduate course to provide a simple real-world navigation problem with aspects of data gathering, processing and analysis.

We begin by defining the system. If we assume that the elevator goes directly up and down (relative to local gravity), then our state vector need only incorporate height (h), velocity (v), and acceleration (a). It is worth noting that an accelerometer measures proper acceleration and so will measure the acceleration due to gravity ($\mathbf{g}$) as an acceleration upwards with magnitude $|\mathbf{g}|$ in addition to the net acceleration of the device ($\mathbf{a}_{net}$), i.e.

$$\mathbf{a}_{meas} = R\left(\mathbf{a}_{net} + \mathbf{g}\right), \tag{5.3}$$

where R is the rotation into the measurement frame. We assume that any acceleration from the elevator is applied along the same line as local gravity and therefore to obtain the net acceleration we need simply subtract $|\mathbf{g}|$ from our total measured acceleration. For higher dimensional systems we will need a more mathematically complete transformation, see subsection 5.3.

We begin with the following second order differential equation which relates the height of the elevator with the net acceleration of the elevator.

$$\frac{\mathrm{d}^2 h(t)}{\mathrm{d}t^2} = a(t). \tag{5.4}$$

Which can then be separated into the following two, first order differential equations,

$$\dot{v}(t) = a(t), \tag{5.5}$$

$$\dot{h}(t) = v(t). \tag{5.6}$$

From the terms in the equations above we can form a state vector ($\mathbf{x}$) like so,

$$\mathbf{x} = \begin{bmatrix} h \\ v \\ a \end{bmatrix}, \tag{5.7}$$

which gives us the state differential equation below.

$$\mathbf{x} = \begin{bmatrix} h \\ v \\ a \end{bmatrix}, \quad \dot{\mathbf{x}} = \begin{bmatrix} v \\ a \\ \dot{a} \end{bmatrix}. \tag{5.8}$$

Now that we have a differential equation for the state vector, and we know how the measurement of acceleration refers to our state, we can write the state propagation matrix ($F(k)$) and state measurement matrix ($H(k)$), as defined in Equation 1.1.

$$\begin{aligned} F(k) &= \begin{bmatrix} 1 & dt & \frac{1}{2}dt^2 \\ 0 & 1 & dt \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{w}(k) = \begin{bmatrix} 0 \\ 0 \\ w(k) \end{bmatrix}, \\ H(k) &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}, \mathbf{v}(k) = \begin{bmatrix} 0 \\ 0 \\ v(k) \end{bmatrix}. \end{aligned} \tag{5.9}$$

The system noise from the elevator and the measurement noise from the accelerometer are denoted by $\mathbf{w}(k)$ and $\mathbf{v}(k)$, respectively. We will assume that both the system and measurement noises are zero mean and normally distributed.

In Figure 5.1 we see the measured accelerations in an elevator. Initially the elevator is waiting, then around 15 seconds after the start of the measurements, the elevator begins to accelerate upwards, reaching a peak upwards acceleration of around 0.6 ms^{-2}. Then at around 20 seconds the elevator begins to accelerate downwards before stopping, two floors above its starting position. At around 37 second the elevator then goes back down to the original floor.

We can simply integrate the measured acceleration (subtracting $|\mathbf{g}|$ first) to estimate the velocity; integrating over the same time period as shown in Figure 5.1, we obtain an estimate of the velocity, which is presented in Figure 5.2.

We can clearly see in the latter half of Figure 5.3 that the integrated errors in the acceleration measurement accumulate in the velocity and position estimates, leading to accurate short-term estimates but increasingly inaccurate long-term estimates.

A key part of state estimation is uncertainty estimation, without which a state estimate is almost meaningless. As mentioned in Chapter 1 the Kalman filter simultaneously propagates the uncertainty of the state as well as the state estimate. Therefore if we run the same data from Figures 5.1 - 5.3 through a Kalman filter

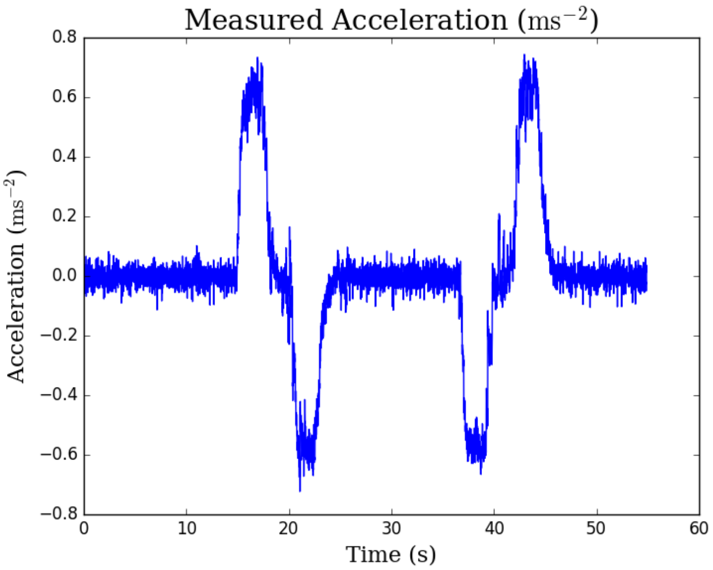


Figure 5.1: Measured acceleration in an elevator going up and down two floors.

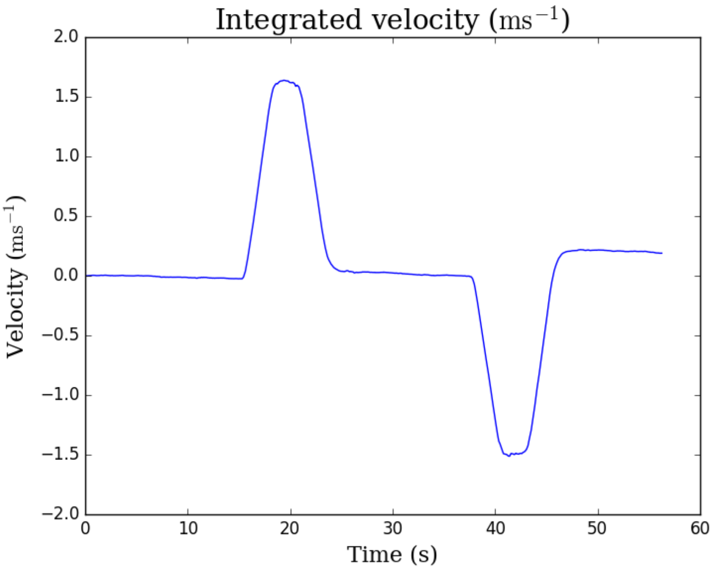


Figure 5.2: By integrating the acceleration over the same time period as shown in Figure 5.1 we get an estimate of the velocity.

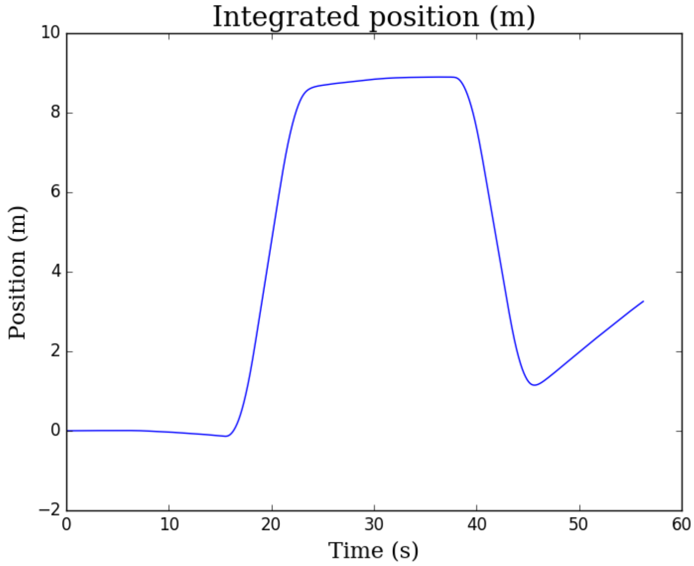


Figure 5.3: By integrating the velocity shown in Figure 5.2 we obtain an estimate of the change in position. However it is important to remember that the true final position of the elevator is the same as the initial position. The difference seen above is a result of the integrated errors of the measured acceleration.

we should be able to see the effect of these accumulated errors on the velocity and position estimates.

As we can see in Figure 5.4 the uncertainty in the velocity estimate starts to wander from the true values, this is clearly seen when the elevator is stationary but the estimate is changing. These small errors in the velocity estimate integrate to become increasing errors in the position estimate, seen in Figure 5.5.

This example demonstrates the main pitfalls of inertial navigation. An inertial navigation system gives accurate and useful short-term estimates, however the integration of errors means that without extremely accurate navigational or tactical grade IMUs, only using inertial navigation can lead to significant errors in the velocity and position estimates. Early on in maritime navigation inertial navigation was referred to as ‘dead reckoning’, possibly hinting at the macabre consequences of solely relying on inertial navigation [1].

---

[1] Some have proposed that dead reckoning might be from nautical abbreviation ded. (“deduced) in log books. However, according to the Oxford English Dictionary this proposal has “no justification” and the true etymology is probably from dead (adj.) in the sense of “unrelieved, absolute.

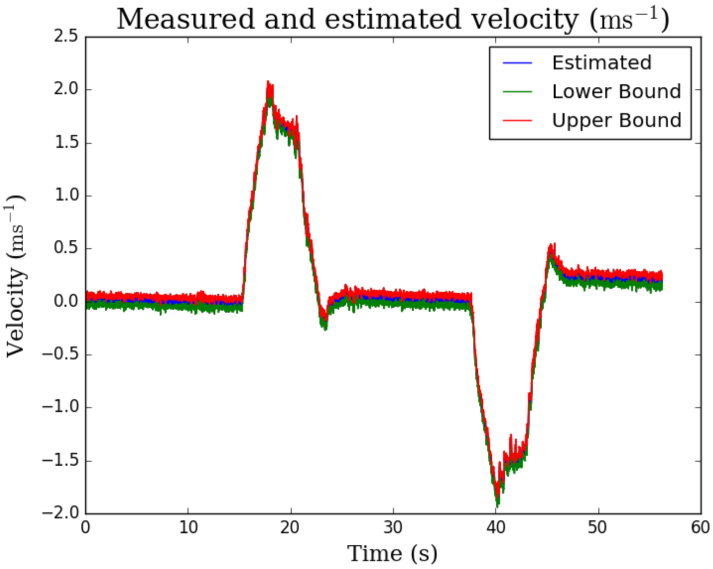


Figure 5.4: The Kalman filtered velocity estimate of the same time period as shown in Fig 5.1. The confidence interval shown is 95%.

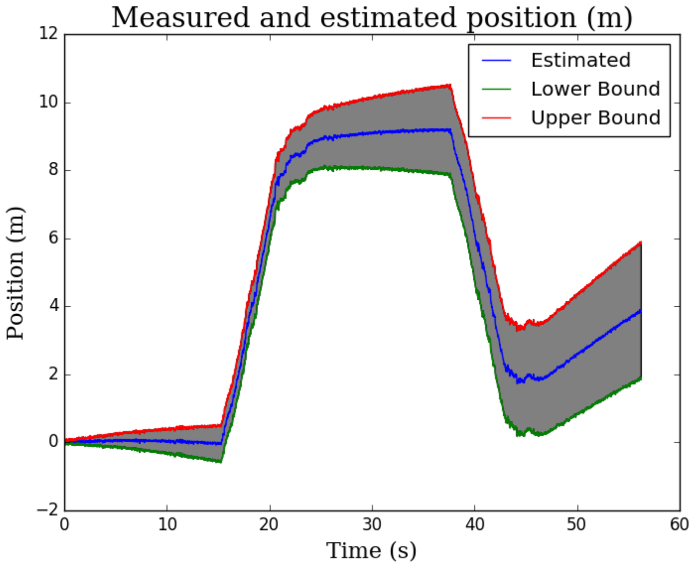


Figure 5.5: The Kalman filtered position estimate of the elevator height. The confidence interval shown is 95%.

5.1.1 One-dimensional Sensor Fusion

The purpose of this one-dimensional example of inertial navigation is to illustrate the general ideas that will be used in Chapter 7 to improve vehicle navigation when combined with GPS measurements. We can further illustrate these ideas by adding a position measurement once the elevator has stopped at a new floor. This measurement is simply done through timing and a record of which floors the elevator stopped at and when.

Observe in Figure 5.6 how the uncertainty of the position estimate shrinks and then continues to grow as the inertial navigation errors propagate. This is a very similar process to the combination of GPS and inertial measurements and is a simple process when operating in the Kalman filter framework. In this model we can illustrate the advantages of sensor fusion that will be implemented in the combined GPS, inertial navigation model described in Chapter 6.

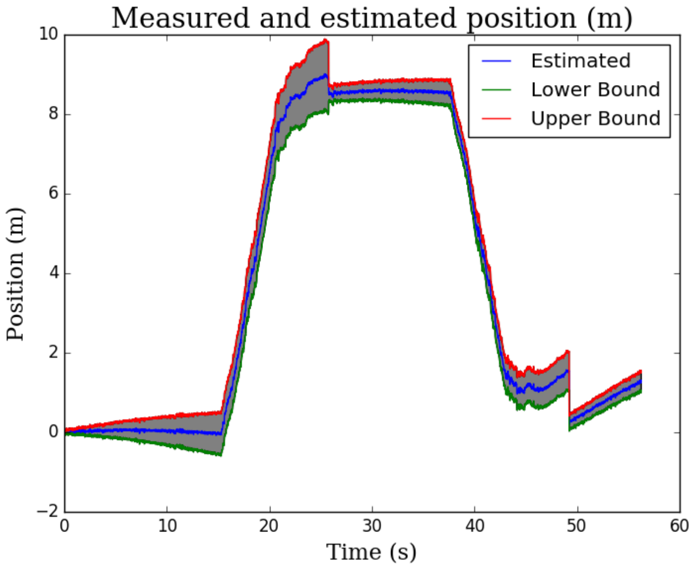
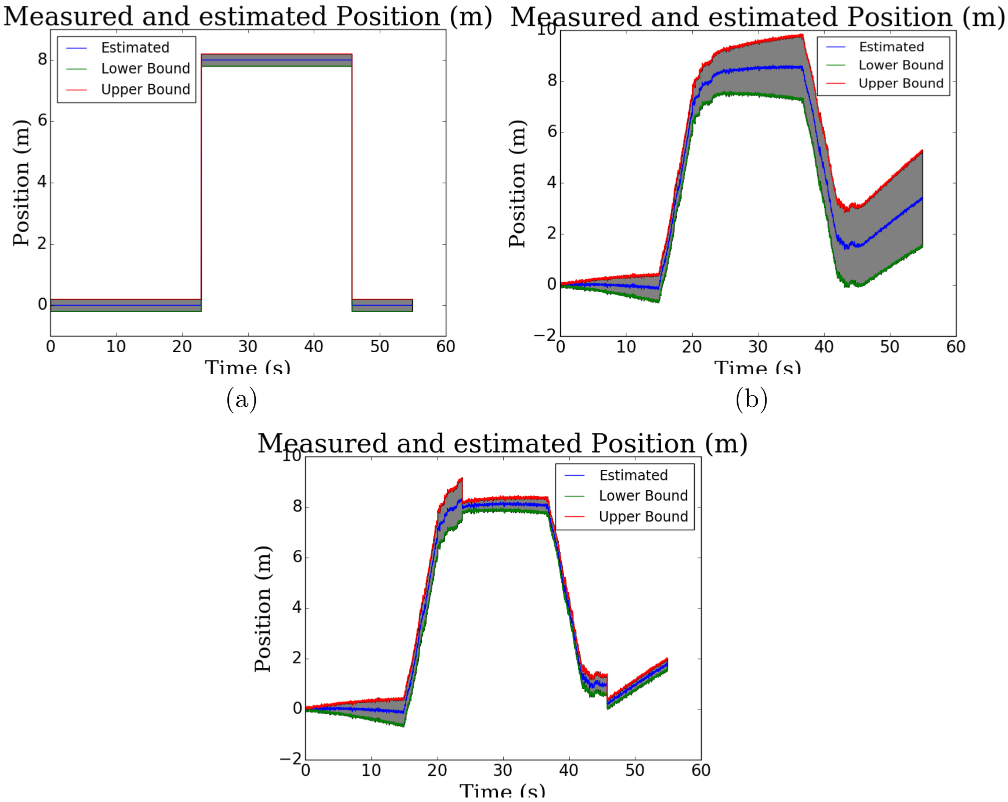


Figure 5.6: The Kalman filtered position estimate, with two position measurements at $\sim$ 25 and 48 seconds. The acceleration data is the same as used previously and the confidence interval shown is 95%.

Figure 5.6 clearly shows the two-fold effects of incorporating different measurement units. Firstly the position estimate adjusts, this prevents the filter from diverging from the true state due to the integrated inertial errors. Secondly the confidence interval shrinks markedly after the new measurement.

In Figure 5.7 we see the position, velocity, and acceleration estimates of a position-only measurement system, an acceleration-only measurement system and a



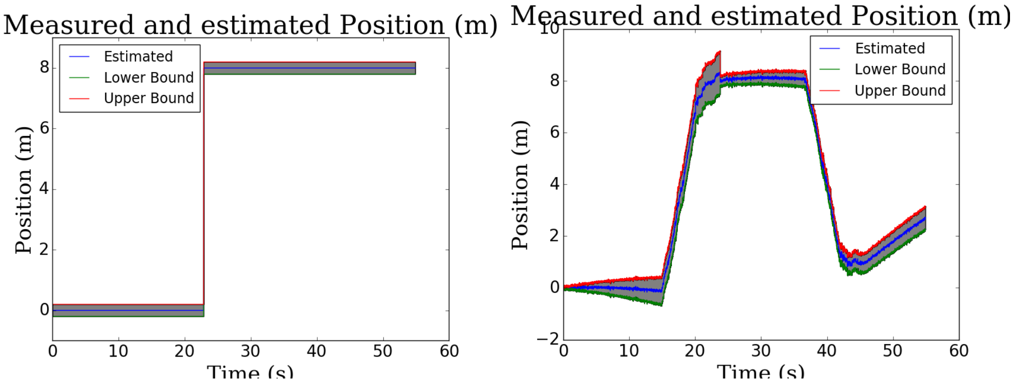
(a)

(b)

(c)

Figure 5.7: Here we have systems with position only (a), acceleration only (b), and combined measurement systems (c). It is clear that by combining different measurement sources the overall state estimate is considerably improved over the individual measurements alone.

system that uses both measurement types.



(a) (b)

Figure 5.8: This shows the effect of losing the ability to make position measurements on: a system that only uses position measurements (a), and a system that combines measurement systems (b). In the absence of position measurements the position-only system can't know that the elevator has changed height without more measurements. However, in that case the combined system simply navigates by inertial navigation.

Finally in Figure 5.8(b) we illustrate the effect of losing the ability to measure position on the position estimate of a combined measurement system. As you can see, without position measurements the system is navigating solely by inertial navigation just as in Figure 5.3.

By comparison, Figure 5.8(a) illustrates what would happen if a position-only measurement system loses the ability to measure position. The filter cannot update its state estimate and so must wait until the system can make measurements again.

This loss of position measurements is obviously less than optimal. However, such measurement outages are common when making GPS position measurements; especially if the measurement device has an obstructed sky-view. For example when the device is in a steep valley or an 'urban canyon', an artificial valley formed by tall buildings on either side.

In the absence of a GPS signal a GPS-only system cannot tell if a vehicle has turned, stopped, or accelerated without first regaining the signal. In Chapter 7 we will demonstrate this idea again and simulate GPS signal losses in empirical data. In the absence of GPS measurements the combined system will simply navigate by inertial navigation. We will then compare these results against the original data and against a system that only uses GPS measurements to navigate.

5.2 Multi-Dimensional Inertial Navigation

By only considering a one-dimensional system in the previous section we were able to investigate the general ideas of inertial navigation without worrying about the specific details. The former example was only ever designed as a stepping stone towards implementing an inertial navigation system for terrestrial vehicle navigation. Consequently we must now be much more explicit in our terminology, especially regarding reference frames and measured forces.

5.2.1 Reference Frames

For the purposes of vehicle tracking we are primarily interested in three, related reference frames. The Earth frame, the local navigation frame and the body frame.

- For the Earth frame we will be using an Earth Centered Earth Fixed (ECEF) reference frame, which is stationary with respect to the surface of the Earth; it is rotating around its axis with a period of one sidereal day. The coordinates that we use to describe navigation in the ECEF reference frame are given by the World Geodetic System WGS84 ellipsoid. WGS84 approximates the Earth as an ellipsoid centered at the Earth's center of mass and has an Equatorial semi-major axis of 6 378 137.0 m and a semi-minor axis of approximately 6 356 752.314 m [51]. WGS84 is the standard model used by GPS and so the natural model to use when navigating by GPS.

  For the local navigation frame we used the East, North, Up (ENU) convention. The origin of the ENU reference frame is defined to be the vehicle's center of mass. The axes of the ENU frame point, predictably, in the East, North, and Upwards directions. This does mean that as the vehicle travels across the surface of the Earth the ENU frame will rotate, albeit quite slowly for most terrestrial navigation purposes, we will go into further detail on this point in Section 5.5.

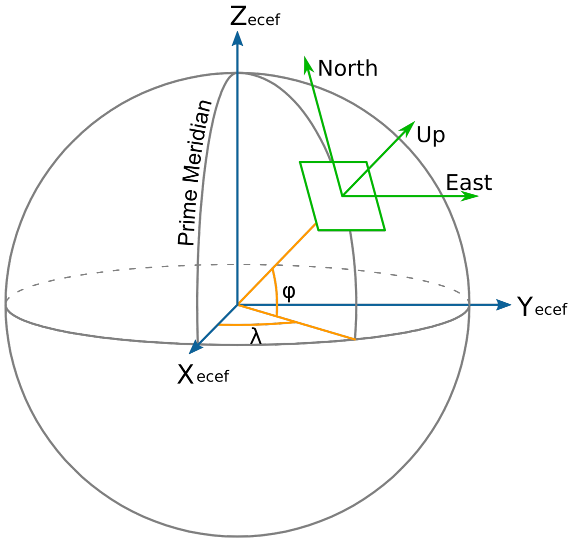


Figure 5.9: The relation between the Earth frame (ECEF) and the local navigation frame. The prime meridian is clearly shown and latitude and longitude are represented by ϕ and λ, respectively. Image from Wikipedia [52].

- The origin of the body frame is coincident with the origin of the local navigation frame at the vehicle's center of mass. The axes of the body frame are rightwards, forwards, and upwards with respect to the centre of mass of the vehicle.

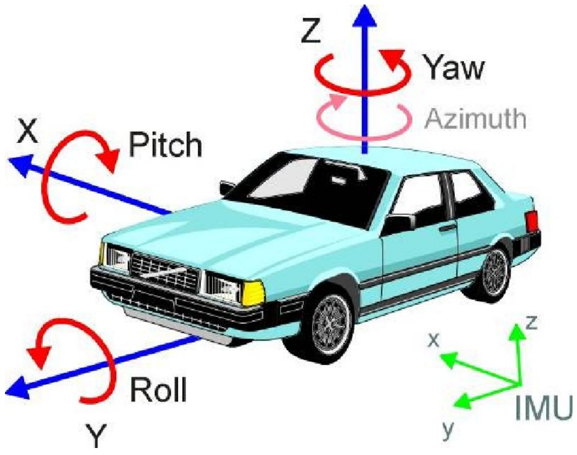


Figure 5.10: The three axes of the body frame of a vehicle.

Each of these reference frames are suited to different descriptions of the navigation of the vehicle. It is in the Earth frame that the position of the vehicle is defined, and the estimation of this is the most fundamental purpose of any navigational system.

However, when discussing the direction and speed that a vehicle is travelling, it is often more convenient to discuss directions such as North-South, East-West or Up-Down. This also allows us to use familiar units such as metres per second or kilometers per hour, as opposed to the rates of change of latitude and longitude.

Finally it is in the body frame that the attitude and acceleration of the vehicle is calculated before being transformed into the local navigation frame. It is also important to remember that any measurements made on the vehicle will, naturally, be made in the body frame. Therefore if the vehicle is changing orientation or accelerating, the associated non-inertial effects should be corrected for to ensure the most accurate navigation solutions. These corrections will be discussed in more depth in Section 5.5.

5.3 Transforming Between Reference Frames

An important part of any inertial navigation system is the rotation formalism adopted in that system. Euler's rotation theorem says that any rotation of a rigid body can be described by a single rotation about a single axis [53]. Therefore the minimum number of parameters to describe such a rotation is three, however most rotation formalisms have extra, redundant parameters for various reasons.

For this project three different rotation formalisms were investigated: the rotation matrices, quaternions and Euler angles. The rotation matrix will be familiar to many readers. It simply a 3×3 matrix with a determinant of 1. The vector is rotated by ordinary matrix multiplication with the rotation matrix. Successive rotations are performed by simply multiplying the two rotation matrices together. The ease of use and familiarity are strong reasons to adopt the rotation matrix formalism. However, being a 3×3 matrix the rotation matrix has six redundant parameters.

Quaternions are a four dimensional hypercomplex number system introduced by William Rowan Hamilton in 1843 [53]. It can be shown that the group of unit quaternions can span the hypersphere of the 3D rotation group SO(3) [49]. Having only four parameters the quaternions considerably reduce the number of redundant parameters when compared to the rotation matrices.

Euler angles are a three dimensional parameterisation of the rotation space, where each angle represents a rotation of that size around one of the axes of the coordinate system. These angles are often colloquially known as pitch, roll, and yaw. Being a three dimensional parameterisation the Euler angles have no redundant parameters, also the pitch, roll, yaw system is intuitive and the resulting equations are simple to derive and simple to check. However the Euler angles system is susceptible to a problem known as gimbal lock. Gimbal lock occurs when the navigation system approaches 90° pitch, at which point the differential equations that update the system become indeterminate [49].

Initially this project intended to use the quaternion formalism to describe system rotations. With only one redundant parameter the quaternion system requires a smaller state vector than the rotation matrix formalism and still avoids the problem of gimbal lock. A quaternion class for this purpose was written in Python and underwent unit testing.

However, eventually we decided that the three dimensional Euler angles formalism was sufficient for our purposes. The angles themselves are intuitive and understandable, and it is a simple task to assign different uncertainties to each angle, this is useful as GPS units can report heading information with reasonable accuracy and we wanted to be sure that the uncertainty in the other rotations of the system was limited so that we could best make use of this information. Because our system is intended for terrestrial, road-going vehicles only we need not concern ourselves overly with the problem of gimbal lock which can occur at high pitch angles. Simply put, if a road-going vehicle is approaching a pitch angle of 90° navigation may not be the primary point of concern.

We have given a brief overview of the rotation formalisms analysed for this project. A more in-depth description of the rotation formalisms discussed is available in Appendix A.3.

5.3.1 Euler Angles

As mentioned above the Euler angles rotation formalism represents the rotation as three separate rotations, about the three axes of the coordinate system. A positive rotation follows the right-hand rule. A rotation around the upward or z axis is denoted (γ), the forwards or y axis is denoted (ϕ) and around the rightwards or x axis is denoted(θ).

We can write these rotations in the familiar form of the rotation matrices given below.

$$\begin{aligned}
R_x &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}, \\
R_y &= \begin{bmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{bmatrix}, \\
R_z &= \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}.
\end{aligned} \tag{5.10}$$

Since rotations are not commutative, we must specify the order or rotations. In our case we adopt the common z-y-x order, (that is, first a rotation about the z axis, then the y axis, then the x axis). This gives us the following composite rotation matrix.

$$R_E^B = R_x R_y R_z. \tag{5.11}$$

$$R_E^B = \begin{bmatrix} \cos\phi\cos\gamma & \cos\phi\sin\gamma & -\sin\phi \\ -\cos\theta\sin\gamma + \sin\theta\sin\phi\cos\gamma & \cos\theta\cos\gamma + \sin\theta\sin\phi\sin\gamma & \sin\theta\cos\phi \\ \sin\theta\sin\gamma + \cos\theta\sin\phi\cos\theta & -\sin\theta\cos\gamma + \cos\theta\sin\phi\sin\gamma & \cos\theta\cos\phi \end{bmatrix}. \tag{5.12}$$

In Equation 5.12 the matrix rotates some vector from the E (ENU) frame to the B (Body) frame. The inverse rotation of a rotation matrix is simply the transpose of the original rotation matrix, i.e.

$$R_B^E = (R_x R_y R_z)^T = R_z^T R_y^T R_x^T. \tag{5.13}$$

$$R_B^E = \begin{bmatrix} \cos\phi\cos\gamma & -\cos\theta\sin\gamma + \sin\theta\sin\phi\cos\gamma & \sin\theta\sin\gamma + \cos\theta\sin\phi\cos\theta \\ \cos\phi\sin\gamma & \cos\theta\cos\gamma + \sin\theta\sin\phi\sin\gamma & -\sin\theta\cos\gamma + \cos\theta\sin\phi\sin\gamma \\ -\sin\phi & \sin\theta\cos\phi & \cos\theta\cos\phi \end{bmatrix}. \tag{5.14}$$

For our purposes the system of Euler angles is an intuitive and convenient description of the rotation of the body frame with respect to the the ENU frame. In our situation where we have a highly constrained system, the pitch and roll angles have much less freedom than the yaw angle.

5.3.2 Euler Angles and Rotation

An important part of any transformation formalism is the process of updating the transformation given some frame rotation $\boldsymbol{\omega}$. We can calculate the rate of change of each of the Euler angles by using the appropriate rotation matrices and the angular velocity. The rates of change of the Euler angles given below will be denoted by dot notation.

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ 0 \\ 0 \end{bmatrix} + R_x \begin{bmatrix} 0 \\ \dot{\phi} \\ 0 \end{bmatrix} + R_x R_y \begin{bmatrix} 0 \\ 0 \\ \dot{\gamma} \end{bmatrix}. \tag{5.15}$$

The equation given above can then be rearranged to give the following [49].

$$\begin{aligned}\dot{\theta} &= (\omega_y \sin\theta + \omega_z \cos\theta)\tan\phi + \omega_x,\\ \dot{\phi} &= \omega_y \cos\theta - \omega_z \sin\theta,\\ \dot{\gamma} &= (\omega_y \sin\theta + \omega_z \cos\theta)\sec\phi.\end{aligned} \tag{5.16}$$

5.3.3 Conversion from ECEF(°) to ENU(m)

The ECEF reference frame (using coordinates from WGS84) is the global navigation frame and the most convenient for navigating over large distances. However we must be able to transform to and from other reference frames. To transform to the ENU frame we must know the semi-major (a) and semi-minor (b) axis lengths of the ellipsoid and the inverse flattening ($1/f$) of the ellipsoid. In the case of the WGS84 ellipsoid these are 6,378,137.0 m (a), 6,356,752.3142 m (b), and 298.257223563 ($1/f$).

We will begin by calculating the distance between two points, specified by WGS84 latitude and longitude. In this case the distance between the respective latitudes is given by [54],

$$s(\phi) = a(1-e^2)\int_{\phi_1}^{\phi_2} \left(1 - e^2 \sin^2\phi\right)^{-3/2} d\phi, \tag{5.17}$$

where ϕ_1 and ϕ_2 are the two latitudes, a is the semi-major axis and e is the eccentricity of the WGS84 ellipsoid. The eccentricity e and flattening f of an ellipsoid are related by the following.

$$e^2 = 2f - f^2. \tag{5.18}$$

In order to use Equation 5.17 the latitudes must be expressed in radians. If we assume that the distance between the two points is “small” then we can use the Euler approximation, i.e.

$$\begin{gathered}\Delta\phi = \phi_2 - \phi_1,\\ s(\phi) \approx a(1-e^2)\left(1 - e^2\sin^2\phi\right)^{-3/2}\Delta\phi.\end{gathered} \tag{5.19}$$

Whenever an assumption is made we should be aware of the range where that assumption is valid, in this case it is the assumption of a “small” difference in latitudes. Equation 5.17 is derived from the meridian radius of curvature of an ellipsoid. To test the validity of the “small” assumption we calculated the distance by numerically integrating the full distance expression and then compared it against the simplification in Equation 5.19. For a difference of one degree in latitude (around 111 km) the simplification is correct to within 0.00015%.

The calculation of the difference between longitudes with the same latitude is simpler than the same case for latitudes. This is because the polar flattening means that the Earth is non-circular along meridian arcs (lines of constant longitude). However when using the ellipsoid model of the Earth lines of constant latitude are circular, this allows us to easily calculate the distance between two points of longitude:

$$s(\lambda) = \frac{a \cos\phi \Delta\lambda}{\sqrt{1 - e^2 \sin^2\phi}}. \tag{5.20}$$

where $\Delta\lambda$ is the difference in longitude, just like with the latitude calculations the longitude here is expressed in radians instead of degrees to avoid extra terms in the expression.

5.4 Non-Inertial Reference Frames

Ideally, all measurements would be made in inertial reference frames. We would then use Newton's Laws to solve the equation in one reference frame and it would be a trivial task to transform the solution to whatever reference frame was most convenient using one of the methods mentioned in section 5.3. However even the surface of the Earth is not an inertial reference frame, let alone a vehicle travelling on the surface of the Earth. We, therefore are forced to work in non-inertial reference frames.

Let us begin by considering two reference frames O and O′. Reference frame O is an inertial reference frame while reference frame O′ is both accelerating and rotating relative to O.

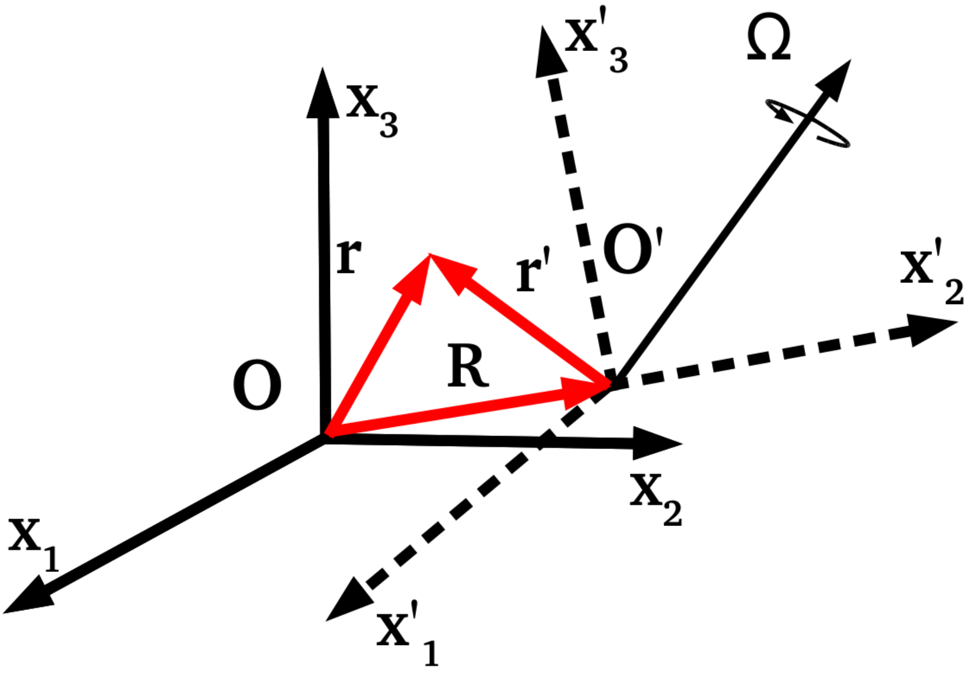


Figure 5.11: Above are two reference frames, one (O) is an inertial frame, the other (O′) is displaced by vector $\mathbf{R}$ and rotating with respect to O with angular velocity $\boldsymbol{\Omega}$.

From Figure 5.11 it is clear that the position vectors are related as follows.

$$\mathbf{r} = \mathbf{R} + \mathbf{r}'. \tag{5.21}$$

We want to know how $\mathbf{r}'$ changes over time. Let's start by taking the time derivative of $\mathbf{r} - \mathbf{R}$, i.e. the position vector ($\mathbf{r}'$) as seen in the inertial frame O,

$$\begin{aligned} \left(\frac{\mathrm{d}(\mathbf{r}-\mathbf{R})}{\mathrm{d}t}\right) &= \left(\frac{\mathrm{d}(r_1-R_1)}{\mathrm{d}t}\right)\hat{\mathbf{x}}_1 + \left(\frac{\mathrm{d}(r_2-R_2)}{\mathrm{d}t}\right)\hat{\mathbf{x}}_2 + \left(\frac{\mathrm{d}(r_3-R_3)}{\mathrm{d}t}\right)\hat{\mathbf{x}}_3, \\ \left(\frac{\mathrm{d}\mathbf{r}'}{\mathrm{d}t}\right) &= \sum_{j=1}^{3}\left(\frac{\mathrm{d}(r_j-R_j)}{\mathrm{d}t}\right)\hat{\mathbf{x}}_j = \sum_{j=1}^{3}\left(\frac{\mathrm{d}r_j}{\mathrm{d}t} - \frac{\mathrm{d}R_j}{\mathrm{d}t}\right)\hat{\mathbf{x}}_j, \end{aligned} \tag{5.22}$$

where $\hat{\mathbf{x}}_j$ is the unit vector of the j^{th} axis of O and r_j denotes the j^{th} coefficient of the vector as seen in O. In our case our measurements are made in the non-inertial frame O′, therefore it is convenient to be able to expand the time derivatives of $\mathbf{r}'$ so that the effects of making measurements in a non-inertial frame are explicit and therefore more easily compensated for.

As seen in the inertial frame (O) the axes of O′ are rotating with angular velocity $\boldsymbol{\Omega}$. Therefore if we want to relate the time derivatives in the different terms we will have to incorporate the product rule, and take into account the acceleration of $\mathbf{R}$.

$$\begin{aligned}\left(\frac{\mathrm{d}\mathbf{r}'}{\mathrm{d}t}\right)_i &= \left(\frac{\mathrm{d}r_1'}{\mathrm{d}t}\right)\hat{\mathbf{x}}_1' + \left(\frac{\mathrm{d}r_2'}{\mathrm{d}t}\right)\hat{\mathbf{x}}_2' + \left(\frac{\mathrm{d}r_3'}{\mathrm{d}t}\right)\hat{\mathbf{x}}_3' \\ &\quad + r_1'\left(\frac{\mathrm{d}\hat{\mathbf{x}}_1'}{\mathrm{d}t}\right) + r_2'\left(\frac{\mathrm{d}\hat{\mathbf{x}}_2'}{\mathrm{d}t}\right) + r_3'\left(\frac{\mathrm{d}\hat{\mathbf{x}}_3'}{\mathrm{d}t}\right), \\ &= \sum_{j=1}^{3}\left(\frac{\mathrm{d}r_j'}{\mathrm{d}t}\right)\hat{\mathbf{x}}_j' + \sum_{j=1}^{3} r_j'\left(\frac{\mathrm{d}\hat{\mathbf{x}}_j'}{\mathrm{d}t}\right), \\ &= \left(\frac{\mathrm{d}\mathbf{r}'}{\mathrm{d}t}\right)_n + \boldsymbol{\Omega}\times\mathbf{r}'.\end{aligned} \tag{5.23}$$

In an effort to avoid confusion the i subscript has been added when the differentiation is performed with respect to the inertial frame (O) and an n subscript when it is with respect to the non-inertial frame (O′).

Using the same process as seen in Equation 5.23 we can take the time derivative again to get the following.

$$m\left(\frac{\mathrm{d}^2\mathbf{r}'}{\mathrm{d}t^2}\right)_i = \left(\frac{\mathrm{d}^2\mathbf{r}'}{\mathrm{d}t^2}\right)_n + 2\boldsymbol{\Omega}\times\left(\frac{\mathrm{d}\mathbf{r}'}{\mathrm{d}t}\right)_n + \left(\frac{\mathrm{d}\boldsymbol{\Omega}}{\mathrm{d}t}\right)\times\mathbf{r}' + \boldsymbol{\Omega}\times\left(\boldsymbol{\Omega}\times\mathbf{r}'\right). \tag{5.24}$$

Substituting in Equation 5.22 and rearranging we may then get the acceleration as seen in O′.

$$\left(\frac{\mathrm{d}^2\mathbf{r}'}{\mathrm{d}t^2}\right)_n = \left(\frac{\mathrm{d}^2\mathbf{r}}{\mathrm{d}t^2}\right)_i - \left(\frac{\mathrm{d}^2\mathbf{R}}{\mathrm{d}t^2}\right)_i - 2\boldsymbol{\Omega}\times\left(\frac{\mathrm{d}\mathbf{r}'}{\mathrm{d}t}\right)_n - \left(\frac{\mathrm{d}\boldsymbol{\Omega}}{\mathrm{d}t}\right)\times\mathbf{r}' - \boldsymbol{\Omega}\times\left(\boldsymbol{\Omega}\times\mathbf{r}'\right). \tag{5.25}$$

We can then multiply both sides by the mass of some object located at $\mathbf{r}'$ to obtain the familiar expressions for the so-called ‘fictitious forces’.

$$\left(\frac{\mathrm{d}^2\mathbf{r}'}{\mathrm{d}t^2}\right)_n = \mathbf{F} - m\left(\frac{\mathrm{d}^2\mathbf{R}}{\mathrm{d}t^2}\right)_i - 2m\boldsymbol{\Omega}\times\left(\frac{\mathrm{d}\mathbf{r}'}{\mathrm{d}t}\right)_n - m\left(\frac{\mathrm{d}\boldsymbol{\Omega}}{\mathrm{d}t}\right)\times\mathbf{r}' - m\boldsymbol{\Omega}\times\left(\boldsymbol{\Omega}\times\mathbf{r}'\right). \tag{5.26}$$

The first term on the RHS of Equation 5.26 is any external force $\mathbf{F}$, the second term is the translational force due to the linear acceleration of O′, the third term is the Coriolis force, followed the Euler Force, and finally the centrifugal force. Because all of our measurements are made in non-inertial reference frames the ‘fictitious forces’ are measured just the same as true forces, e.g. gravity.

5.5 Reference Frame Corrections and Approximations

As mentioned previously, none of our reference frames are inertial frames. In this section we examine the magnitude of the non-inertial effects on our frames and correct for them.

We will begin with the Earth Centered Earth Fixed reference frame. Since this frame is stationary relative to the Earth, it is rotating around the polar axis at a rate of one rotation per sidereal day. This equates to an angular rate $\omega_e = 7.292115 \times 10^{-5}$ rad s^{-1} [51]. Transforming this rotation into the ENU reference frame we have the following angular rate,

$$\omega_e^E = \begin{bmatrix} 0 \\ \omega_e \cos(\lambda) \\ \omega_e \sin(\lambda) \end{bmatrix}, \tag{5.27}$$

where λ is the latitude of the vehicle. High accuracy inertial navigation systems will therefore have to correct for this angular rate, and update these corrections as the latitude changes. For our purposes an angular rate of 7.292115×10^{-5} rad s^{-1} is negligibly small as the least significant bit for the FXAS21002c gyroscope used in this project is 1.332×10^{-4} rad s^{-1}, when operating over the lowest possible range. Therefore, in order to simplify the calculations the rotation of the Earth has been omitted from future navigation equations. This simplification allows us to approximate the surface of the Earth as an inertial reference frame.

As a vehicle travels across the surface of the Earth the axes which point East, North, and Upwards will rotate at an angular rate of v/r, where v is the speed of the vehicle on the surface and r is the local radius of the Earth. For a vehicle travelling north at the Equator at 100 km/hr this angular rate of rotation is 4.355×10^{-6} rad s^{-1}. While this is even smaller than the effect of the rotation of the Earth, it is important to remember that this effect will accumulate as the vehicle travels away from its initial position. So while this effect may be ignored for shorter trips, the author advises that for longer trips e.g. traversing more than 50 km, the ENU frame be rotated accordingly.

As seen in Equation 5.26 there are several fictitious forces that emerge as a consequence of making measurements in a non-inertial reference frame. The first assumption we will make is that the sensors are fixed within the body frame, thus we can ignore the Coriolis force

5.5.1 Measurement Corrections in Non-inertial Frames

Now that we have an explicit description of the effects of non-inertial reference frames we want to use these to obtain a better description of our system. For this it is useful to examine the effects of the non-inertial frame on the different measurement types individually.

For the position element of the GPS we are unconcerned with the time derivatives of the position and so we simply rotate the sensor position into the ENU frame. GPS units also report velocity, which will be affected by the non-inertial frame, recalling Equation 5.23.

$$\left(\frac{\mathrm{d}\mathbf{r}'}{\mathrm{d}t}\right)_i = \left(\frac{\mathrm{d}\mathbf{r}'}{\mathrm{d}t}\right)_n + \boldsymbol{\Omega} \times \mathbf{r}'. \tag{5.28}$$

Therefore if we approximate the ECEF frame as an inertial frame the speed of the centre of mass of the vehicle is the reported speed from the GPS unit subtracted by the cross product of the angular velocity (reported by the gyroscope) and the position of the GPS unit with respect to the centre of mass of the vehicle. An important question now is, 'is this effect negligible?' Imagine that the vehicle is turning at a traffic light and takes 2 seconds to turn 90°, and that the GPS is 0.5m horizontally from the vehicle's centre of mass (assuming a flat road and no lean on the vehicle any vertical distance from the centre of mass would not contribute to a centripetal speed). This would provide a speed error of 0.39ms^{-1}, well above the quoted accuracy of 0.1ms^{-1} of the GlobalSat BU-353S4 GPS unit used in this thesis [48].

Therefore to correct for the centripetal velocity to obtain the velocity of the centre of mass we simply use the following equation,

$$\mathbf{v}^E = \mathbf{v}^B - \boldsymbol{\Omega} \times \mathbf{r}, \tag{5.29}$$

where $\mathbf{v}^B$ is the velocity reported from the GPS, $\mathbf{v}^E$ is the velocity of the centre of mass of the vehicle, and $\boldsymbol{\Omega} \times \mathbf{r}$ is the cross product of the rate of rotation and the position of the GPS in the body reference frame.

With the output from the GPS corrected for non-inertial effects we must also correct the output of the accelerometer. Recall that by assuming that the sensor is fixed in the body reference frame we are able to ignore the Coriolis force, and so we are left with the following equations,

$$\begin{aligned}
\left(\frac{\mathrm{d}^2\mathbf{r}'}{\mathrm{d}t^2}\right)_n &\approx \left(\frac{\mathrm{d}^2\mathbf{r}}{\mathrm{d}t^2}\right)_i - \left(\frac{\mathrm{d}^2\mathbf{R}}{\mathrm{d}t^2}\right)_i - \left(\frac{\mathrm{d}\boldsymbol{\Omega}}{\mathrm{d}t}\right) \times \mathbf{r}' - \boldsymbol{\Omega} \times (\boldsymbol{\Omega} \times \mathbf{r}') \,, \\
\left(\frac{\mathrm{d}^2\mathbf{r}'}{\mathrm{d}t^2}\right)_n &\approx \frac{\mathbf{F}_i}{m} - \left(\frac{\mathrm{d}^2\mathbf{R}}{\mathrm{d}t^2}\right)_i - \left(\frac{\mathrm{d}\boldsymbol{\Omega}}{\mathrm{d}t}\right) \times \mathbf{r}' - \boldsymbol{\Omega} \times (\boldsymbol{\Omega} \times \mathbf{r}') \,,
\end{aligned} \tag{5.30}$$

where $\mathbf{F}_i/m$ is the external force divided by the mass of the vehicle, including the normal force of the ground in response to gravity. Substituting gravity into the our measured acceleration equation we obtain.

$$\begin{aligned}\left(\frac{\mathrm{d}^2\mathbf{r}'}{\mathrm{d}t^2}\right)_n &\approx \mathbf{g}_i - \left(\frac{\mathrm{d}^2\mathbf{R}}{\mathrm{d}t^2}\right)_i - \left(\frac{\mathrm{d}\mathbf{\Omega}}{\mathrm{d}t}\right) \times \mathbf{r}' - \mathbf{\Omega} \times (\mathbf{\Omega} \times \mathbf{r}')\,, \\ \left(\frac{\mathrm{d}^2\mathbf{r}'}{\mathrm{d}t^2}\right)_n &\approx \mathbf{g}_i - \left(\frac{\mathrm{d}^2\mathbf{R}}{\mathrm{d}t^2}\right)_i - \left(\frac{\mathrm{d}\mathbf{\Omega}}{\mathrm{d}t}\right) \times \mathbf{r}' - \mathbf{\Omega} \times \left(\mathbf{v}^E\right).\end{aligned} \tag{5.31}$$

In Equation 5.31, we replaced $\mathbf{\Omega} \times \mathbf{r}'$ with $\mathbf{v}^E$, because we have assumed that the dominant term in the centrifugal acceleration will be due to rotation about the upwards axis, i.e. the vehicle turning a corner. Preliminary data gathered from the gyroscope and accelerometer appears to corroborate this assumption, the upwards facing gyroscope reported rates of rotation far higher than those obtained from the other two axes, as shown in Figure 5.12.

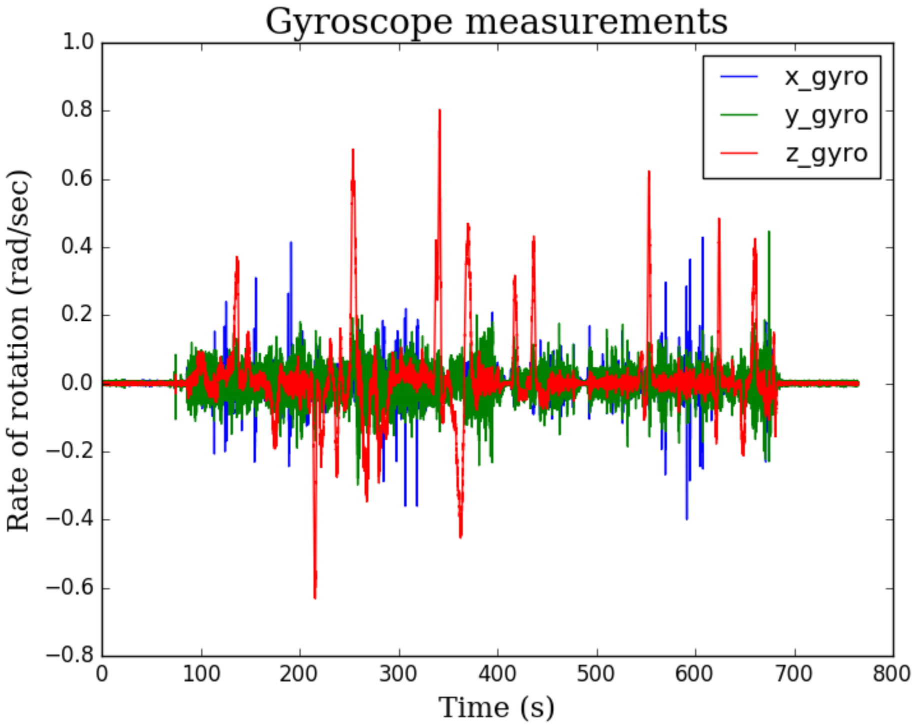


Figure 5.12: Gyroscope data obtained from driving a vehicle around Dunedin, along the circuit shown in Chapter 7. The x, y, z gyroscope measurements correspond to rotations about the rightwards, forwards, and upwards axes, respectively.

The substitution in Equation 5.31 takes into account the centrifugal acceleration terms due to rotation about the upwards and rightwards axes. However, the substitution explicitly ignores the centrifugal acceleration due to a rotation about the forwards axis of the vehicle. This term is treated as noise in the signal in the implementation in Chapters 6,7. This imperfect substitution is proposed because the centrifugal force term depends on the rate of rotation and either the radius of curvature of the path or the velocity. Without real-time access to the radii of curvature the first term would have to be estimated only from the accelerometer measurements, further increasing the degrees of freedom in the system. However the system already measures and estimates the forwards speed of the vehicle. So rather than estimate another unknown variable we could simply use the information already measured and available to estimate the centrifugal acceleration term. We felt that this simplification justified treating the last term of the centrifugal acceleration as noise. This assumption will be further discussed in the following chapter.

Chapter 6

Sensor Fusion in Vehicle Navigation

Now that we are familiar with the problems of transforming between reference frames and inertial navigation we are in a position to define a reasonable state space for terrestrial vehicle navigation. Before we do so, we begin with a few simple assumptions in order to simplify the equations later on in this chapter.

Firstly we assume that the tyres of the vehicle do not allow for lateral motion, i.e. while the vehicle is able to turn it is unable to slide laterally. This allows us to assume that all of the velocity of the vehicle is in the forwards direction of the vehicle, thereby allowing us to eliminate the lateral velocity term from the state vector.

Secondly we assume that the suspension of the vehicle is infinitely rigid, and so any measured roll angle is assumed to be due to the road surface. If we were to take into account the spring constants of the vehicle we could look up the spring constants and enter those into the model, this would necessarily be vehicle dependent, which goes counter to this project's aim of developing a vehicle independent device that can be easily transferred from one vehicle to another. Alternatively we could estimate the spring constants on the fly as part of the larger vehicle state estimation problem. This would be a valuable addition to the research herein however we believed that the additional benefits of spring constant estimation would be outweighed by the added complexity in the model. Also while in reality, the suspension of a vehicle does affect the attitude of the vehicle, especially during cornering we believe that these terms are small enough to safely ignore.

It was our goal to estimate the driver input into the vehicle. Is the driver braking while driving down the hill or accelerating? Recall that accelerometers measure proper acceleration, i.e. a stationary accelerometer on the Earth's surface would measure the acceleration due to gravity as a vector with magnitude $|g|$ in the *upwards* direction. Therefore a forwards facing accelerometer would measure the net forwards acceleration (a_F) and the projection of a positive gravity vector onto the forwards direction, which gives us the following expression,

$$a_{meas} = a_F + g \sin\theta \cos\phi = a_T, \tag{6.1}$$

where the term (a_T) is the forwards acceleration provided by the truck. That is, the forwards acceleration (a_F) of the vehicle without the forwards acceleration due to gravity.

$$a_F = a_T - g \sin\theta \cos\phi. \tag{6.2}$$

It is important to note that we do still *measure* the three dimensional acceleration of the vehicle. These measurements allow us, via the following nonlinear measurement equation to make measurements of the pitch (θ) and roll (ϕ) angles of the vehicle. This is very useful as it means that the system can directly measure these angles and so prevent the errors in the gyroscopic measurements from causing the angle estimates to diverge from the true value. As well measuring the pitch and roll angles, three dimensional acceleration measurements also allow us to measure the centripetal acceleration of the vehicle and so provide us with redundant measurements of the rate of rotation of the vehicle as it corners.

$$h(\mathbf{a}) = \begin{bmatrix} a_R \\ a_F \\ a_U \end{bmatrix} = \begin{bmatrix} -g\sin\phi - \omega_z v_F \\ a_T \\ g\cos\theta\cos\phi + \omega_x v_F \end{bmatrix} + \begin{bmatrix} v_R \\ v_F \\ v_U \end{bmatrix}. \tag{6.3}$$

Finally, as stated in Chapter 5 we chose to use the Euler angles method to represent the attitude of the vehicle. The Euler angles represent a computationally efficient three state representation of the vehicle attitude, compared to the four state quaternion representation or nine state rotation matrix representation. Also the pitch, roll, and yaw angles allow for an intuitive and understandable description of the vehicle. The Euler angles representation is not commonly deployed in navigational equations due to the singularity at a pitch angle of 90^o. However given the constraints for a terrestrial vehicle operating primarily on roads the singularity problem is unlikely to provide much trouble, even Baldwin St only gets to 19^o.

6.1 Euler Angles

As mentioned above we chose the Euler angles rotation formalism to represent the attitude of the vehicle. Before we derive the state differential equations we should familiarise ourselves with the Euler angles.

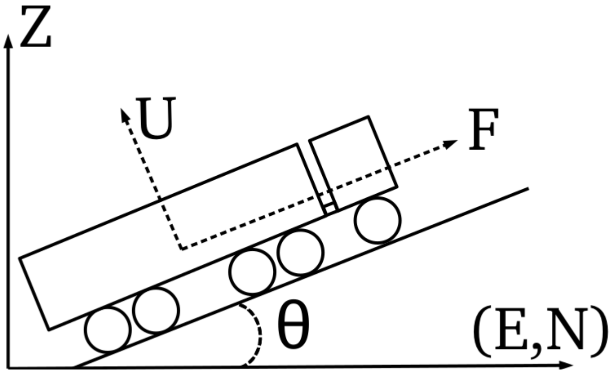


Figure 6.1: Side-view of the vehicle with the directions shown. Note that the bottom axis is the projection onto the horizontal X,Y plane. The F and U vectors show the forwards and upwards directions used for the acceleration variables a_T and a_U, respectively.

The last terms in the state vector θ, ϕ, γ are, respectively, the pitch, roll and heading angles as shown in Figures 6.1, 6.2 and 6.3 where the forwards, upwards and rightwards direction for the positioning of the accelerometers are also defined.

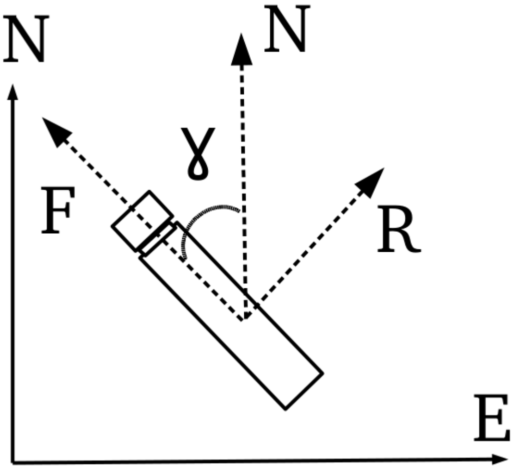


Figure 6.2: Top-view of the vehicle with the directions shown. Note that the Y axis points North and the X axis points East. We have the same F direction as seen in Figure 6.1 as well as the R direction for the rightwards facing acceleration term a_R.

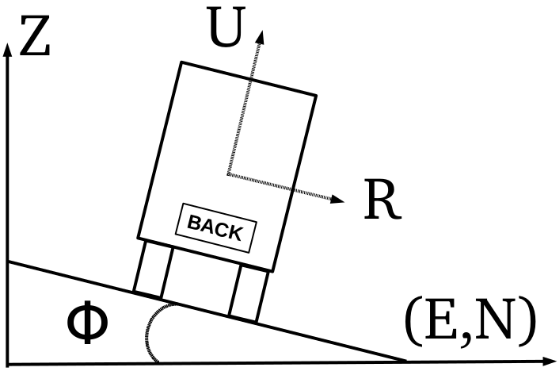


Figure 6.3: Rear-view of the vehicle with the directions shown. Note that as in Figure 6.1 the bottom axis is the projection onto the horizontal X,Y plane.

6.2 State Space and Equations

We can now write out the state vector and system and measurement equations of our system as they would be set out for use in a nonlinear state estimator.

Recall from Chapter 1 that our nonlinear state and measurement equations are given as.

$$\begin{aligned} \mathbf{x}(k+1) &= f(\mathbf{x}(k)) + \mathbf{w}(k), \\ \mathbf{z}(k) &= h(\mathbf{x}(k)) + \mathbf{v}(k). \end{aligned} \tag{6.4}$$

Where $f(\dots)$ is the system propagation equation and $h(\dots)$ is the state measurement equation. The state vector and differential equation of our system is given below,

$$\mathbf{x}(k) = \begin{bmatrix} x \\ y \\ z \\ v_F \\ a_T \\ \theta \\ \phi \\ \gamma \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}, \dot{\mathbf{x}}(k) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{v_F} \\ \dot{a_T} \\ \dot{\theta} \\ \dot{\phi} \\ \dot{\gamma} \\ \dot{\omega_x} \\ \dot{\omega_y} \\ \dot{\omega_z} \end{bmatrix} = \begin{bmatrix} (-\cos\theta\sin\gamma + \sin\theta\sin\phi\cos\gamma)\, v_F \\ (\cos\theta\cos\gamma + \sin\theta\sin\phi\sin\gamma)\, v_F \\ \sin\theta\cos\phi v_F \\ a_T - g\sin\theta\cos\phi \\ w_{aT} \\ (\omega_y\sin\theta + \omega_z\cos\theta)\tan\phi + \omega_x \\ \omega_y\cos\theta - \omega_z\sin\theta \\ (\omega_y\sin\theta + \omega_z\cos\theta)\sec\phi \\ w_{\omega R} \\ w_{\omega F} \\ w_{\omega U} \end{bmatrix}, \tag{6.5}$$

where x, y, z, v_F are the ENU positions and the forwards velocity. The θ, ϕ, γ terms are the Euler angles, and the ω terms are the angular velocities in the rightwards, forwards, and upwards directions. Any terms with a w simply denotes the process noise present for that variable.

There is only one acceleration term included in our state vector (a_T) rather than a three-dimensional term which one might expect. We were able to use a one dimensional acceleration model because of the constraints on our system. Recall that our system is only intended for terrestrial navigation and we have made the assumption of negligible lateral tyre slip. The assumptions that the vehicle cannot accelerate upwards (in the vehicle's body frame) or rightwards (without turning) imply that the only remaining direction for linear acceleration of the vehicle is forwards.

Rather than use a centripetal acceleration term that used the standard $\omega \times (\omega \times \mathbf{r})$ we made use of the following simple expression.

$$\omega \times \mathbf{r} = \mathbf{v}. \tag{6.6}$$

This allows us to relate the measured centripetal acceleration in terms of velocity, which we already estimate, rather than including the radius of the curvature of the road, a factor that would be difficult to estimate *a priori*.

To simplify the measurement equations we assume that the GPS measurements have been converted into the ENU frame and so are in units of metres, metres per second, and radians rather than degrees of latitude/longitude, knots, and heading angle. Therefore the measurements from the GPS unit, in terms of the state measurement equation are given as.

$$h(GPS) = \begin{bmatrix} x \\ y \\ z \\ v_F \\ -\gamma \cos\theta \cos\phi \end{bmatrix}. \tag{6.7}$$

These conversions from the ECEF to the ENU frame were made using the formulas presented in Subsection 5.3.3. The last measurement type, gyroscopic measurements, has the simplest measurement equation.

$$h(\omega) = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}. \tag{6.8}$$

Because the angular velocities are measured and estimated directly and so have a simple linear measurement equation.

We have now defined the nonlinear propagation ($f(\mathbf{x}(k))$) and measurement equations ($h(\mathbf{x}(k))$). This is all that is required for the unscented Kalman filter, however recall from Chapter 1 that the extended Kalman filter we must also define the state and measurement Jacobians that relate to their respective nonlinear equations.

$$J_f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}, J_h = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial h_n}{\partial x_1} & \frac{\partial h_n}{\partial x_2} & \cdots & \frac{\partial h_n}{\partial x_n} \end{bmatrix}. \tag{6.9}$$

Due to the number of terms the full write-outs of the state and measurement Jacobians have been left out of the main text and are available in Appendix A.4.

6.2.1 Vehicle Navigation Algorithm

Now that we have defined our state space, measurement and state equations, etc., we can write out a full description of the algorithm used for vehicle navigation in this thesis. The algorithm itself follows a modular setup where the same underlying model can be used by different filtering classes. We defined classes for the model, measurement and filter being used so that we could easily compare the effects of changing the model, adding measurement types, and using different filters.

The filters that we implemented for this algorithm were the unscented Kalman filter (UKF) and the extended Kalman filter (EKF), which were introduced in Chapter 1. However when we began testing with real-world data we had difficulty maintaining positive definiteness in the state covariance when using the UKF, which caused the UKF to fail in some instances of the algorithm. Therefore the algorithm given below, and the algorithm used for the combined navigation algorithm shown in Chapter 7 uses the EKF formulation rather than the UKF. This was obviously a disappointment

Initialisation

We start with some initial state estimate ($\mathbf{x}_0$) and covariance (P_0), sometimes known as the *prior*. These are derived from our prior knowledge of the system.

$$\hat{\mathbf{x}}(0|0) = \mathbf{x}_0, \; P(0|0) = P_0. \tag{6.10}$$

Prediction

Now we propagate the state estimate and uncertainty ahead in time.

$$\begin{aligned}\mathbf{x}(1) &= f\left(\mathbf{x}(0)\right) + \mathbf{w}(0),\\ \hat{\mathbf{x}}(1|0) &= f\left(\hat{\mathbf{x}}(0|0)\right),\\ P(1|0) &= P(0|0)J_f(1)P(0|0)^T + Q(1),\\ \hat{\mathbf{z}}(1|0) &= h\left(\hat{\mathbf{x}}(1|0)\right).\end{aligned} \tag{6.11}$$

Here $\mathbf{x}(0)$ denotes the true state of the system at time $t = 0$, and $\mathbf{w}(0)$ denotes the system noise, assumed to be zero mean Gaussian with covariance of $Q(0)$. The $\hat{\mathbf{x}}(1|0)$ term denotes the predicted state estimate at time $t = 0$ given 0 measurements. The $J_f(1)$ term denotes the state Jacobian, which is calculated by passing the predicted state $\hat{\mathbf{x}}(1|0)$ to the **get_state_jacobian** method of the model class. The $\hat{\mathbf{z}}(1|0)$ term denotes the predicted measurement obtained via the nonlinear measurement equation $h(\dots)$ which is accessed in the **measurement_prediction** method in the measurement class.

The nonlinear state propagation equation $f(\dots)$ is stored in the **state_prediction** method of the model class given to the filter. If the model has a process covariance that changes over time then this can be called at each iteration through the **get_process_covariance** method in the model class, otherwise the process covariance is obtained before the iteration begins and is simply multiplied through by the dt for that iteration.

Measurement and Update Now we obtain the new measurement and update the state estimate and uncertainty accordingly.

$$\begin{aligned}K(1) &= P(1|0)J_h^T(1)\left(J_h(1)P(1|0)J_h^T(1) + R(1)\right)^{-1},\\ \mathbf{r}(1) &= \mathbf{z}(1) - \hat{\mathbf{z}}(1|0),\\ \hat{\mathbf{x}}(1|1) &= \hat{\mathbf{x}}(1|0) + K(1)\mathbf{r}(1),\\ P(1|1) &= \left(I - K(1)J_h(1)\right)P(1|0).\end{aligned} \tag{6.12}$$

Here $J_h^T(1)$ is the measurement Jacobian, which is calculated by passing the predicted state to the **get_measure_jacobian** method in the measurement class. The only other foreign term above is the measurement covariance, denoted $R(1)$. The algorithm simply iterates through the data set until it is out of measurements. We usually ran the algorithm on stored data, however the EKF algorithm can easily be used as a real-time estimator, the only concern would be for processing power available on whatever device is running the algorithm.

The computer used for design and testing of the algorithm has two AMD A4-4020 CPUs, each with 1.61GB of RAM, and a clock speed of 2.6GHz, and is able to run the algorithm on experimental data at around 12.5 times real-time, i.e. 100s of data from GPS, accelerometer, and gyroscope data takes around 8s to filter. It is possible that speed-ups could be made such that the algorithm runs faster, however for the purposes of testing this was not necessary. However making a meaningful effort to streamline the algorithm would be a useful contribution to the project in future.

6.3 Two Dimensional Test Model

During design we wanted to be able to test the theoretical benefits of combining GPS and accelerometer data using a nonlinear Kalman filter. We built a model where a simulated vehicle was driving along a hilly, straight road while the 'driver' applied positive or negative accelerations. In our model whenever the vehicle was going up the hill the 'driver' would apply a positive acceleration, likewise whenever the vehicle was going down a hill the 'driver' would apply a negative acceleration.

We felt that this was a realistic situation for road-going vehicles and it allowed us to test our model in a situation where estimating the acceleration applied by the vehicle was nontrivial to determine. As an example, let's consider the case of a vehicle coming to a hill, the driver applies more accelerator but not enough to maintain speed and so begins to slow down.

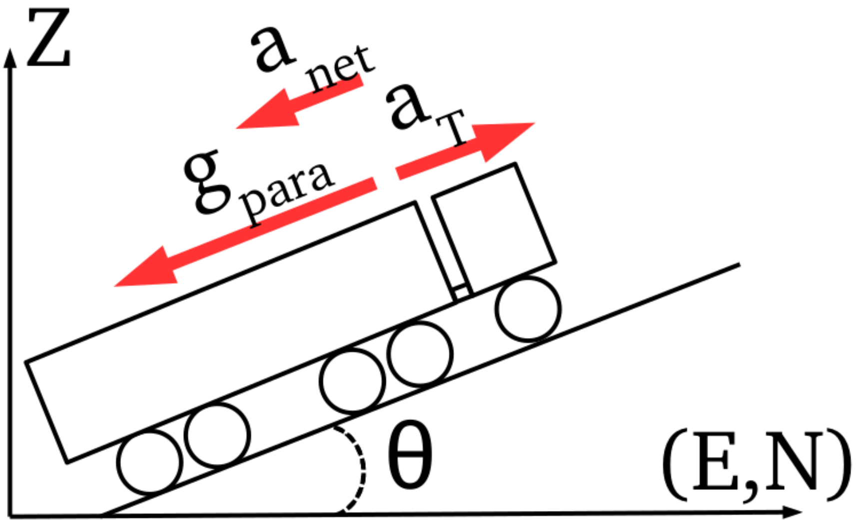


Figure 6.4: A vehicle is travelling up a hill and is experiencing a negative acceleration due to gravity (g_{para}), applying a positive acceleration (a_T) and experiencing a net negative acceleration (a_{net}).

It is easy to see from Figure 6.4 that we have the following acceleration relationship.

$$a_{net} = a_T - g_{para}. \tag{6.13}$$

Where g_{para} is the component of the gravitational acceleration that is parallel to the surface of the road. From Equation 6.3 we know that the measured acceleration will not be the net acceleration, instead it will be the following.

$$\mathbf{a}_{meas} = \mathbf{a}_T. \tag{6.14}$$

Hence the need for a more comprehensive model rather than simply measuring the acceleration and immediately using those measurements directly in the kinematic equations. This demonstrates one of the benefits of the model-based inference approach.

Our main aim for this test case was to ensure that the accelerometer and GPS measurements could be successfully combined in a way that made physical sense. The extra step of adding gyroscopic measurements is considerably simpler as unlike acceleration, there is no net angular velocity (excluding the small term due to the rotation of the Earth) and there are no extra corrections required for the measured angular velocity vector due to the presence of non-inertial reference frames. Therefore for this test case we only simulated accelerometer and GPS measurements.

6.3.1 Results using simulated data

We generated Gaussian noise to be added to both position and accelerometer measurements of the vehicle as it ‘drives’ along the road. In Figure 6.5 we see the measured, estimated and true height of a vehicle after one such simulation.

We then compared the performance of the system when relying on either position data, accelerometer data or a combination of both. The result of that test is in figure 6.7.

As you can see in Figure 6.7 the result of combining both position and accelerometer measurements, predictably, gives better results than using either system separately. The Figure is instructive, however, in illustrating the benefits of accelerometer data. Note that the changes in position-only data are jagged and the position estimate quickly leaps from one position to another. However by directly measuring the second derivatives of position the change in position for an accelerometer based inertial navigation system is much smoother and more realistic. Using a solely accelerometer-based inertial navigation system of course means that any errors may accumulate which is why the accelerometer-only position estimate tracks further and further from the true position.

As another way of comparing the benefit of adding accelerometer data to GPS measurements we compared the effect of increasing the standard deviation of the position noise on three systems. One system using only position measurements, one system using only accelerometer measurements, and one system using both position and accelerometer measurements. In order to identify the overall trend we ran each of the three systems one hundred times for each data point and took the average of

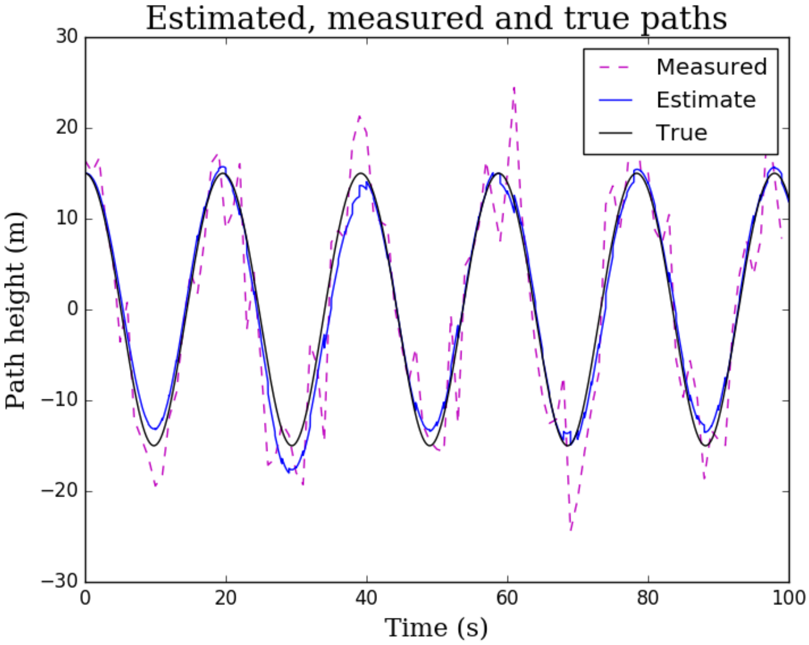


Figure 6.5: The height of a simulated 3D vehicle with measured, estimated and true values shown. The variance for the position measurements was taken from empirical data collected via a commercial GPS positioned at the University of Otago and so is consistent with the errors expected in real-world use.

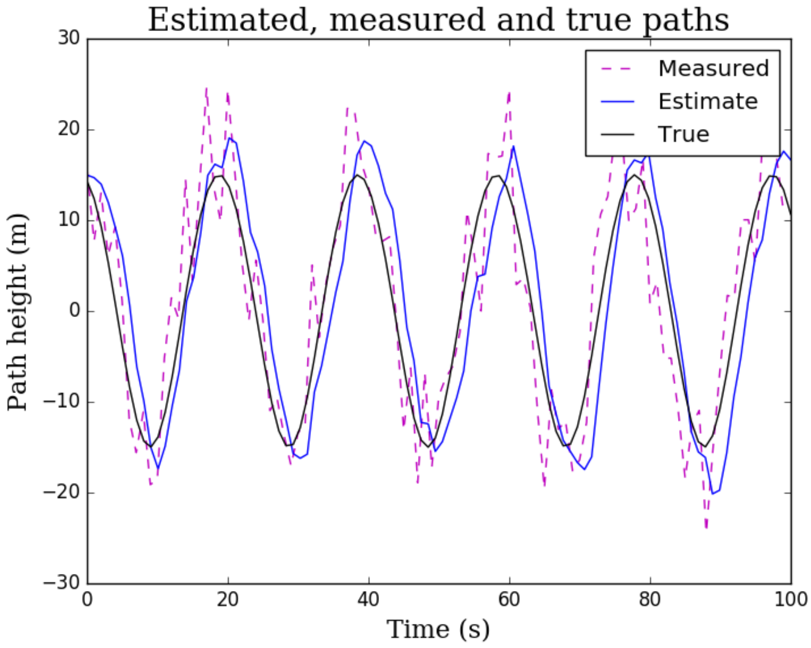


Figure 6.6: Using the same simulated position data as shown in Figure 6.5 but no acceleration data.

the root mean squared (RMS) error. Each simulation had the same underlying true data and standard deviation but uniquely generated additive noise.

As you can see in Figure 6.8, increasing the noise in position measurements, predictably increased the errors in systems using position and combined measurements and did not change purely accelerometer-based predictions. What is interesting is how the gradient of the combined system lessens compared to the purely position-based system as the EKF places more emphasis on the accelerometer data rather than the position data. This effect could be very useful for vehicles briefly losing sight of satellites, for example in urban canyons.

It should be stated that incorporating a change in covariance matrix from simulated data is a trivial matter, quite unlike estimating and incorporating a change in covariance matrix in real time from empirical data gathered from a vehicle that is constantly moving through different local geographies.

Such a system could utilise the number of satellites in view of the device and the horizontal and vertical dilution of precisions (HDOP and PDOP, respectively) reported. For instance,

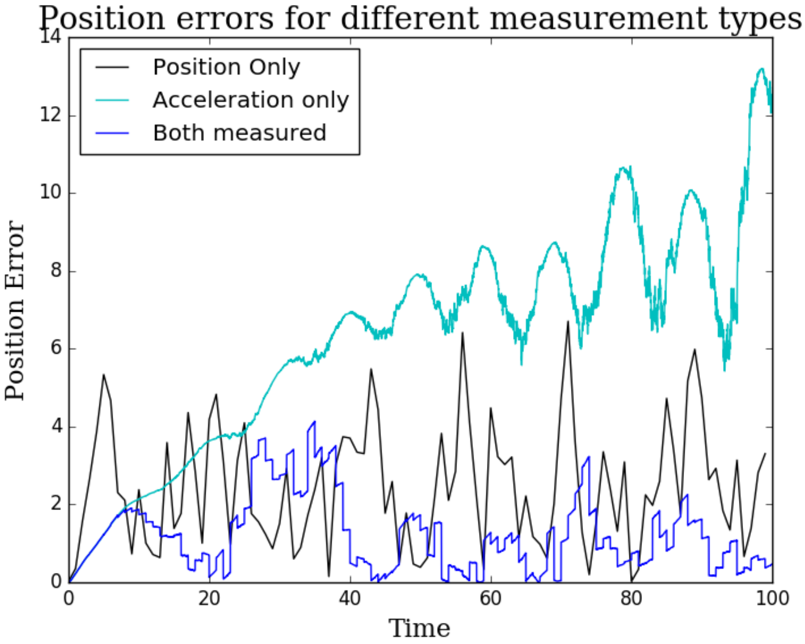


Figure 6.7: The absolute error in position is shown for systems using just position (GPS) measurements, just accelerometer measurements or combining both. The same simulated measurements were used for the combined and separate systems.

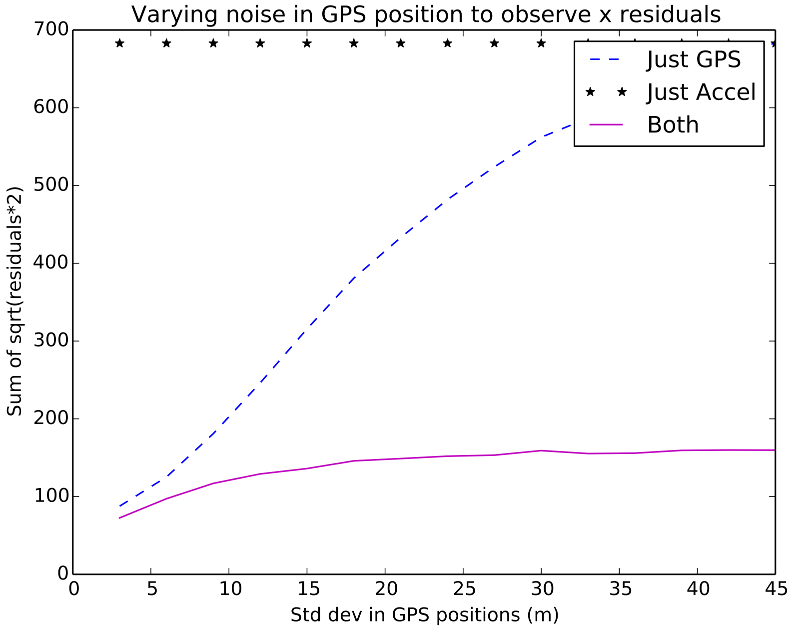


Figure 6.8: Standard deviation of position noise versus errors in position for systems using position-only measurements, accelerometer-only measurements or both.

$$R(k) = \frac{1}{N}\begin{bmatrix} c_1 HDOP & 0 & 0 \\ 0 & c_2 HDOP & 0 \\ 0 & 0 & c_3 VDOP \end{bmatrix}, \tag{6.15}$$

where N is the number of satellites, and the constants c_1, c_2, c_3 are determined by the particular device. As the number of satellites decreases the covariance of the GPS measurements would then increase and the EKF would place more emphasis on the accelerometer data until the line of sight to the missing satellites is reestablished. Therefore for short outages of GPS signals the EKF would allow for accelerometer measurements to more accurately determine the position and heading of a vehicle with obstructed GPS signal.

To develop such a system would be an immensely useful tool, however a comprehensive study would be required to prove its general applicability. Such a study is beyond the scope of this project but would be a valuable contribution to the field.

Chapter 7

Vehicle Navigation System Implementation

The previous chapters defined the system model and tested it under simulated conditions. In this chapter we apply this system to real world data. We begin by describing the hardware setup used in the testing. This entails the hardware used, calibrations performed, and the vibration isolation system used. Once data has been gathered we analyse the performance of the system when the vehicle is stationary. Then the combined GPS and inertial navigation system is compared against the GPS-only system. Of particular interest to this project is the performance of the combined system when subjected to simulated GPS outages.

7.1 Hardware

The purpose of this experiment was to combine GPS, accelerometer, and gyroscope measurements to improve vehicle navigation. For our experiment we wanted hardware that would be easy to modify and easy to use. A system diagram of the communication between hardware components is shown in Figure 7.1.

We chose a Raspberry Pi B+ as the processor, combined with a single channel GlobalSat BU-353S4 GPS unit operating on the L1 band (1575.42 MHz) which connects to the Raspberry Pi via USB. The accelerometer was a Freescale FXOS8700CQ 3-axis linear accelerometer and the gyroscope was a Freescale FXAS21002C 3-axis angular rate gyroscope, connected to the Raspberry Pi through an Arduino Alamode, as shown in Figure 7.1.

7.1.1 Host Processor

The Raspberry Pi B+ model was used in this thesis to receive and store the data from the various sensors. The Raspberry Pi used for this project is shown in Figure 7.2 with the 3D printed frame and vibration damping used for testing. The Raspberry Pi B+ has a Broadcom BCM2835 CPU with 512MB of SDRAM running at 700 MHz. It is equipped with an Ethernet socket, HDMI port, microSD memory card slot, four USB 2.0 connectors, and a 40 pin GPIO capable of handling SPI, I^2C and UART communication protocols [55]. During testing we added a USB wifi

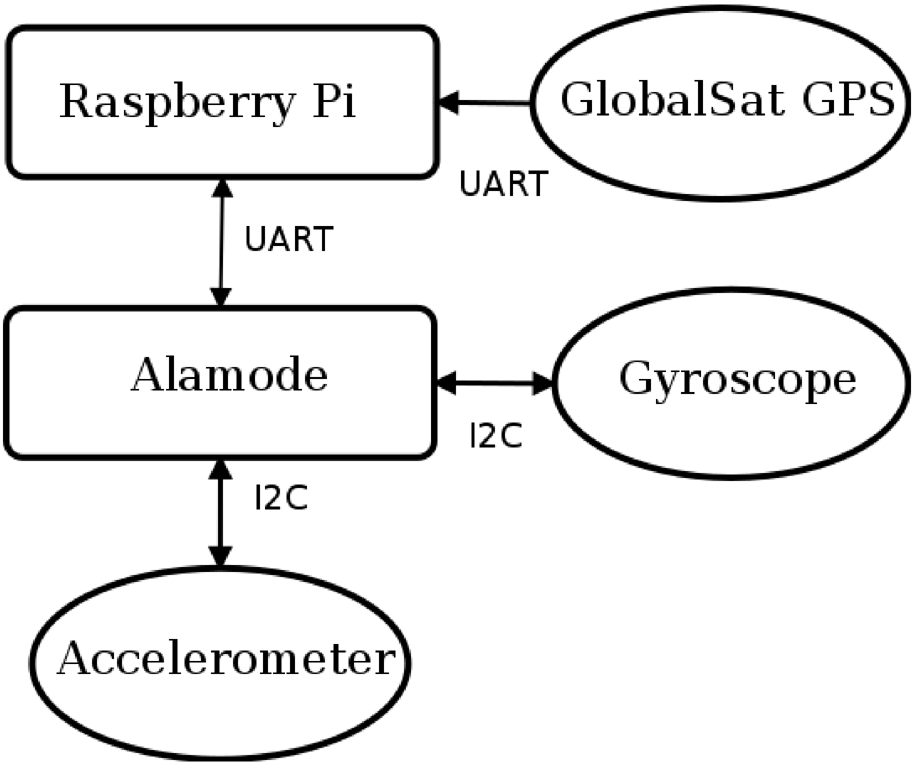


Figure 7.1: The Raspberry Pi communicates with the Alamode through the Pi's GPIO pins, using UART. The Raspberry Pi also communicates with the GPS device using UART, however, via the USB connector instead of the GPIO pins. The Alamode communicates with the accelerometer and gyroscope via I^{2}C.

dongle to the Raspberry Pi, which allowed the user to access the Pi via Secure Shell (SSH) through the user's smart phone and then manage software and data transfer as necessary.

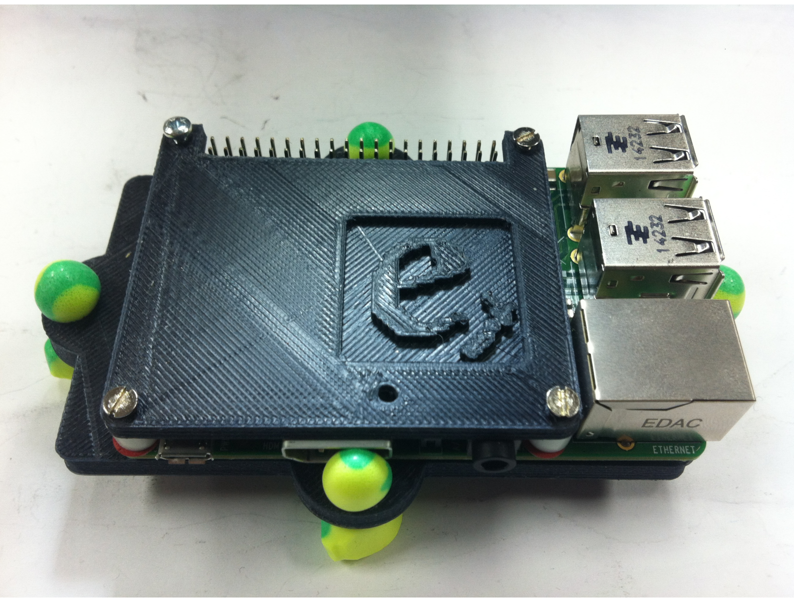


Figure 7.2: The Raspberry Pi B+ mounted on our ear plug vibration mounting plate. Note that the hole on top of the plate is for mounting the Arduino Alamode, which will be attached to the top.

Although, there exist boards that combine integrated processors and accelerometers, the Raspberry Pi provided a complete framework for development and testing. With an Ethernet connection we were able to pull the latest software directly from GitHub and run the Python driver for data collection, either directly with, a screen and keyboard, or via SSH. This proved very convenient for field testing, with a USB WiFi dongle attached to the Raspberry Pi, we were able to SSH into the Pi from a smartphone over the smartphone's WiFi hotspot. Furthermore by connecting the sensors via USB and GPIO the setup is modular, which would be easy to adapt to include other sensors or compare the performance of different sensors.

Other versions of embedded Linux hosts, e.g. Raspberry Pi 3, exist which could also be used to fulfill the same purpose. While it would be reasonably simple matter to upgrade the Raspberry Pi B+ to another host processor in the future, for our purposes the Raspberry Pi B+ was more than adequate.

7.1.2 Alamode

The breakout board used for the accelerometer and gyroscope came with an Arduino R3 connector [56]. The Alamode is an Arduino compatible microcontroller, seen in Figure 7.3. The Alamode plugs directly into the GPIO header on the Raspberry Pi, and comes with an Arduino R3 connector. The Alamode was programmed directly from the Raspberry Pi via the GPIO connection using the standard Arduino IDE. The Alamode comes with a Real-Time-Clock which the Raspberry Pi can use for reference in the absence of an internet connection. In operation the Alamode communicates with the accelerometer and gyroscope via I^2C, if there are any new measurements stored in the internal first in, first out (FIFO) buffers of either device the Alamode reads all stored measurements and then sends the measurements to the Raspberry Pi via UART. The UART was used instead of I^2C because the code developed to drive the GPS unit proved very effective at serial communication and was easily adapted to the Alamode-Raspberry Pi communication.

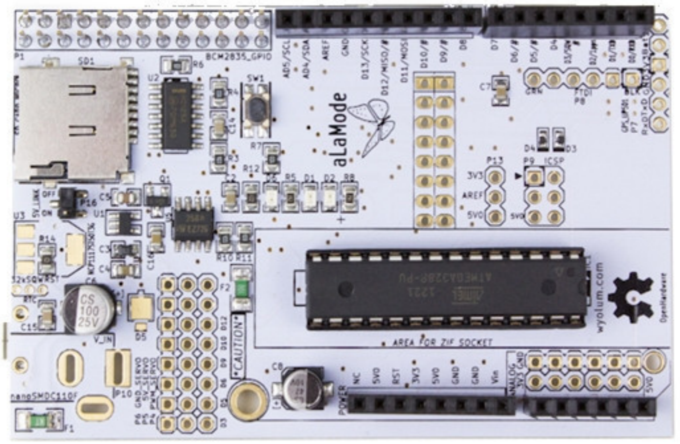


Figure 7.3: The Alamode communicates directly with the accelerometer and gyroscope via I^2C and then relays that information to the Raspberry Pi via UART. Photo from http://www.makershed.com/products/alamode-for-raspberry-pi .

7.1.3 Accelerometer

The accelerometer used was an Freescale FXOS8700CQ sensor with 3-axis linear accelerometer and 3-axis magnetometer. The FXOS8700CQ has 14-bit accelerometer resolution, a full scale range (FSR) of $\pm 2g \pm 4g, \pm 8g$, and an output data rate

(ODR) from 1.563 Hz to 800 Hz. The FXOS8700CQ is capable of communicating via I^2C or SPI. The FXOS8700CQ has a built-in FIFO buffer capable of storing 96 accelerometer measurements (32 measurements for each of the three axes) [10].

7.1.4 Gyroscope

The gyroscope used was a Freescale FXAS21002C 3-axis digital angular rate gyroscope. The FXAS21002C has an adjustable FSR from $\pm 250^o$/s to $\pm 2000^o$/s and an ODR from 12.5 Hz to 800 Hz. Like the FXOS8700CQ the FXAS21002C is able to communicate via either I^2C or SPI and has a built-in FIFO buffer that can hold 32 measurements in each axis [57].

7.1.5 GPS Sensor

The GPS device used in this project was the GlobalSat BU-353S4 device, shown in Figure 7.4. The GlobalSat BU-353S4 device is a single frequency GPS device, operating on the L1 (1575.42 MHz) frequency and uses the SiRF STAR IV GSD4e GPS chipset [48]. The device is capable of outputting the following NMEA sentences GGA, GSA, GSV, RMC, VTG, and GLL v2.2. This device was connected to the Raspberry Pi via USB 2.0. A Python driver for Linux was written by the author and Tim Molteno (Electronics Research Group, University of Otago) for interfacing with the GlobalSat BU-353S4 which has a simple protocol for collecting and storing the desired NMEA sentences.

7.2 Calibration

Both the FXOS8700CQ accelerometer and FXAS21002C gyroscope sensors come with standard factory calibration. The factory calibration is accurate for most applications, however, for the most accurate results when using the sensors they should be recalibrated [10, 57]. Recalibration is required for accurate results as the factory calibration will change slightly due to the thermal stresses experienced during the soldering of the chip to the board [10, 57]. Here we detail the calibration techniques used for both sensors and demonstrate the improvements found.

7.2.1 Allan Variance and Gyroscope Calibration

In order to calibrate the gyroscope we require the offset biases for each axis. It is, of course, a trivial matter to determine the offset biases for the gyroscope axes. Simply set the device running for an extended period of time while stationary and average the resulting data to determine the offsets.

Figure 7.4: A GlobalSat BU-353S4 GPS unit, which was used to make GPS measurements for both the vehicle tracking system and the 1Hz stationary GPS data referred to in chapter 2. Image taken from [48].

However, to understand the noise observed in the gyroscope we took the Allan variance of the noise signal to determine the amount of different noise sources present in the device.

The Allan variance was first developed in the 1960's by David W. Allan to characterise the noise present in precision crystal oscillators and atomic clocks. It has subsequently been adapted to model the random-drift errors present in inertial sensors [58].

Assume that you have a time series $\Omega(t)$ with N data points, each with sampling period dt. Now suppose we choose a set of n consecutive data points within the time series. The cluster will have an associated time period T which will be ndt. The cluster average is defined as,

$$\bar{\Omega}_i(T) = \frac{1}{T}\int_{t_i}^{t_i+T} \Omega(t)dt, \tag{7.1}$$

where $\bar{\Omega}_i(T)$ denotes the cluster average of $\Omega(t)$, starting at the ith data point and containing n data points. Given Equation 7.1 the Allan variance is simply defined as,

$$\sigma^2(T) = \frac{1}{2(N-2n)}\sum_{i=1}^{N-2n}\left[\bar{\Omega}_{i+n}(T) - \bar{\Omega}_i(T)\right]^2. \tag{7.2}$$

A convenient way to express the Allan variance for rate measurements, such as gyroscopes, is by using the integrated variable, in the case of gyroscopes that would be the angle θ.

$$\theta(t) = \int_0^t \Omega(t')dt'. \tag{7.3}$$

Discretising the equation above then gives as,

$$\bar{\Omega}_i(T) = \frac{\theta_{i+n} - \theta_i}{T}. \tag{7.4}$$

Therefore with no loss of generality we can rewrite the Allan variance as follows,

$$\sigma^2(T) = \frac{1}{2(N-2n)} \sum_{i=1}^{N-2n} \left[\theta_{i+2n} - 2\theta_{i+n} + \theta_i\right]^2. \tag{7.5}$$

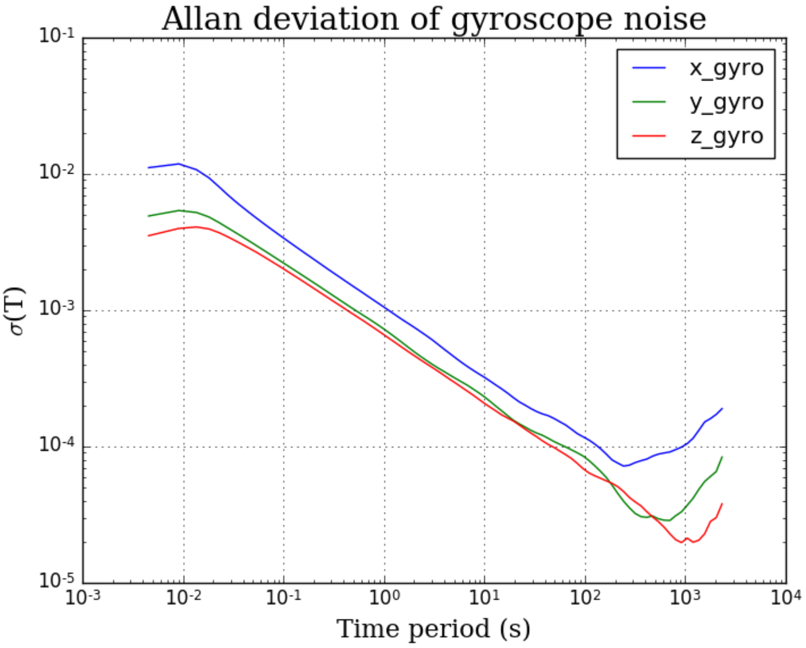


Figure 7.5: The Allan deviation ($\sqrt{\sigma^2(T)}$) of the three axes of the FXAS21002c gyroscope is shown above. Observe that on the loglog graph the Allan deviation of all gyroscope axes are approximately linear with a slope of approximately -0.5, implying that angle random walk is the dominant term. [58]

The most convincing reason to use the Allan variance to describe noise in inertial sensors is the quantitative description of various noise types. The Allan variance of a time series can be shown to provide quantitative descriptions of five common types of noise terms, namely quantization noise, angle random walk, bias instability, rate random walk, and rate ramp. Obtaining these is a trivial matter once the Allan variance has been calculated.

To obtain descriptions of the noise terms it is possible to simply read them off of a log-log graph like Figure 7.5. However a more objective approach is to apply a least-squares fit for the following coefficients [59],

$$\sigma^2(T) = \frac{3Q^2}{T^2} + \frac{N^2}{T} + (0.664B)^2 + \frac{K^2 T}{3} + \frac{R^2 T^2}{2}, \tag{7.6}$$

where Q is the quantization noise coefficient (arcsec), N is the angle random walk coefficient (rad/$\sqrt{\text{hr}}$), B is the bias instability coefficient (rad/hr), K is the rate random walk coefficient (rad$\sqrt{\text{hr}}$), and R is the rate ramp coefficient (radhr).

As the size of the cluster (n) increases there is necessarily fewer clusters in total and so the error for Allan variances of longer time periods is larger than the error for shorter time periods. The percentage error in the estimate of the Allan deviation is simply given by the following [58],

$$\delta\left(\sigma(T)\right) = \frac{1}{\sqrt{2\left(\frac{N}{n} - 1\right)}}. \tag{7.7}$$

Therefore the percentage error for the larger clusters is clearly much higher than for the smaller clusters.

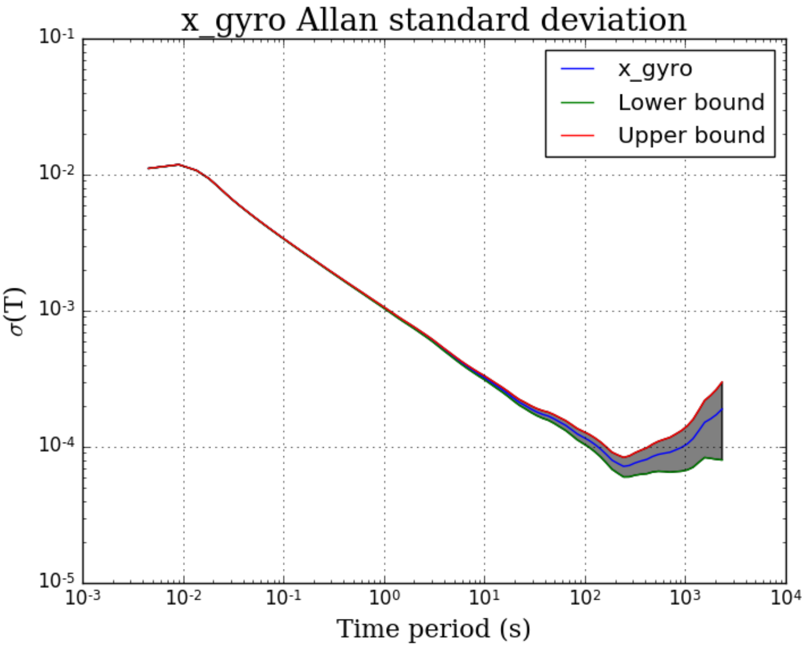


Figure 7.6: The Allan deviation $\left(\sqrt{\sigma^2(T)}\right)$ of the x axis gyroscope with upper and lower bounds shown. Note how the error increases for longer time periods.

A data set was gathered by running the gyroscope constantly for around 80 minutes, sampling at 250Hz. The results are given in the table below.

Coefficient	Q	N	B	K	R
x_gyro	3.101e-06	8.773e-04	1.504e-04	1.653e-10	9.544e-06
y_gyro	3.183e-06	4.012e-04	5.842e-05	1.221e-09	4.838e-06
z_gyro	9.666e-07	2.952e-04	1.421e-04	3.198e-06	2.196e-06

7.2.2 Accelerometer Calibration

We propose a quick, simple method for calibrating the accelerometer that does not require any special experimental setup or extra equipment and could easily be performed in the field if recalibration were required. We begin by assuming that each accelerometer direction has a linear offset and some scaling constant, i.e.

$$\begin{aligned} x_{measured} &= a_{x0} + k_x a_x, \\ y_{measured} &= a_{y0} + k_y a_y, \\ z_{measured} &= a_{z0} + k_z a_z, \end{aligned} \tag{7.8}$$

where $x_{measured}$ is the measured acceleration in the x direction, a_{x0} is the offset in that direction, and k_x is the linear scaling constant of the true acceleration in that direction (a_x), likewise for the other two directions. Then we slowly and smoothly rotate the device through angles in all three directions. After doing this for a reasonable period of time (starting at 30 seconds) we stop the device taking data. If the rotations are sufficiently smooth and slow the vector magnitude of the three accelerometers should be approximately g for the entire period, irrespective of the angle of orientation.

$$|g| = \sqrt{x^2_{measured} + y^2_{measured} + z^2_{measured}} \tag{7.9}$$

Using this relation and our assumption about the nature of the biases we will use a minimization function to estimate the offsets and the scaling constants.

One of the potential problems in this method of estimating biases is that it relies on the assumption of slow, smooth rotation. This is difficult without specialised equipment and so our first test was to look at the vector magnitude of the three accelerometer measurements while the device was being rotated. Figure 7.7 shows the uncalibrated accelerations reported as the device is rotated through a large range of angles. The total acceleration appears to be on average slightly larger than 10.0ms^{-2}. However, it appears to slowly change as the device is rotated, which is precisely the behaviour we would expect if the current calibration was imperfect.

To ensure that the calibration terms that we gained from our experiment were the product of the most testing possible we performed six tests where we slowly rotated the device. We then investigated the reported accelerations while the accelerometer was sitting stationary on a desk and periodically rotated it to ensure that at any orientation, the device reported a value as close as possible to the local gravity ($|g|$).

We found that the best calibration results were obtained by concatenating the results of each individual test together and then using the calibration terms from that total test.

Figure 7.8 shows the same data as seen in Figure 7.7 after the device has been calibrated with the calculated terms. It is worth noting that the data shown in Figures 7.7 and 7.8 was not used to calculate the calibration coefficients.

We see in Figure 7.8 that the small dip observed at around 50 seconds in Figure 7.7 has disappeared. Also we can see that the total measured acceleration is now slightly less than 10.0m^{-2}, as we would expect. In Figure 7.9 we plotted the reported accelerations on a sphere of radius 9.80667 ms^{-2}, the approximate theoretical acceleration due to gravity at the latitude of Dunedin, according to the 1984 Ellipsoidal Gravity Formula [51],

$$g(\phi) = 9.7803253359 \left[\frac{1 + 0.00193185265241 \sin^2(\phi)}{\sqrt{1 - 0.00669437999013 \sin^2(\phi)}} \right], \tag{7.10}$$

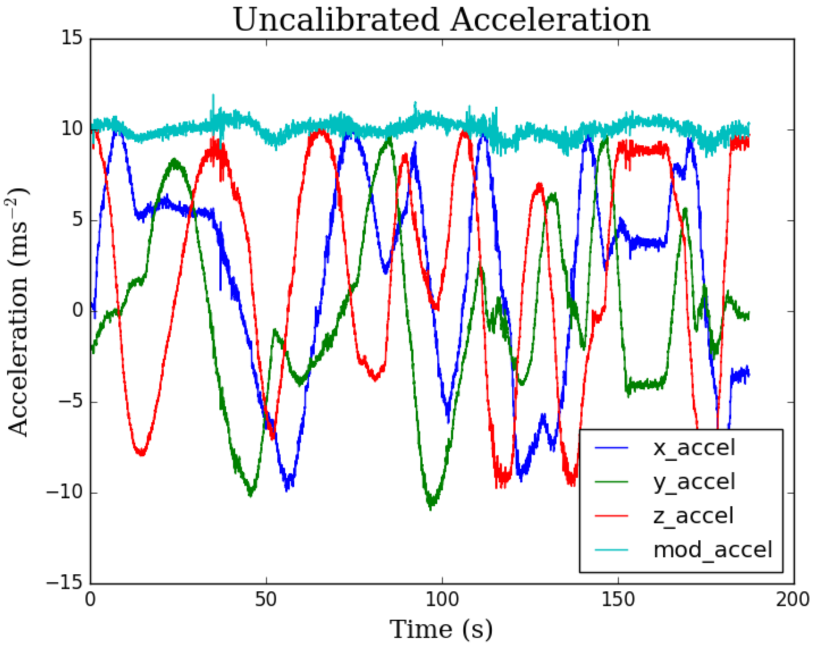


Figure 7.7: The accelerations reported prior to the calibration. Note how the total acceleration appears to smoothly increase and decrease as the device is rotated.

where $g(\phi)$ is the predicted acceleration due to gravity at latitude ϕ.

7.2.3 Vibration Damping and Analysis

A large part of the difficulty involved with using inertial navigation is dealing with system vibrations. That is not to say that there are no processes for dealing with vibration noise, quite the contrary there are many useful and efficient approaches to dealing with vibration noise.

When dealing with some amount of vibration noise there are some simple filters which may easily be integrated, e.g. the moving-average or the low-pass filter. Indeed the Kalman filter itself has a rigorous framework for dealing with noise such as this. However some degree of physical isolation of the system from such vibrations, when used in conjunction with signal processing will yield superior results than either used separately.

For our vibration damping mount the Raspberry Pi was bolted to a 3D printed base. The base then had 4 holes, through which commercially available ear plugs were squeezed. The whole stack was then supported by the ear plugs, as seen in Figure 7.2. Ear plugs are ideal for our purposes as they are readily available, extremely cheap, and designed to damp vibrations. While it's true that their main

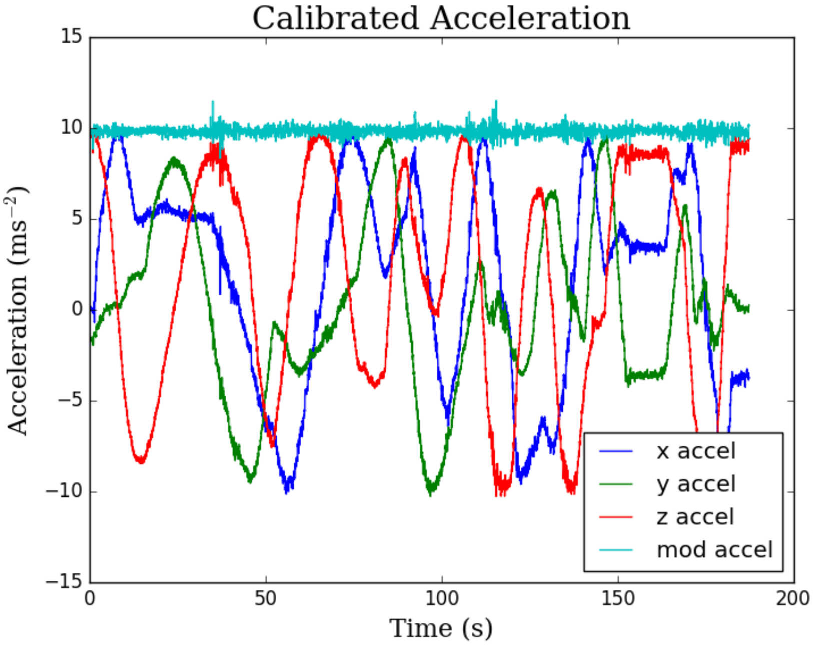


Figure 7.8: The measured accelerations after the calibration. Observe that the total acceleration is now reasonably constant. It is important to remember that the data shown was not used to calculate the calibration coefficients.

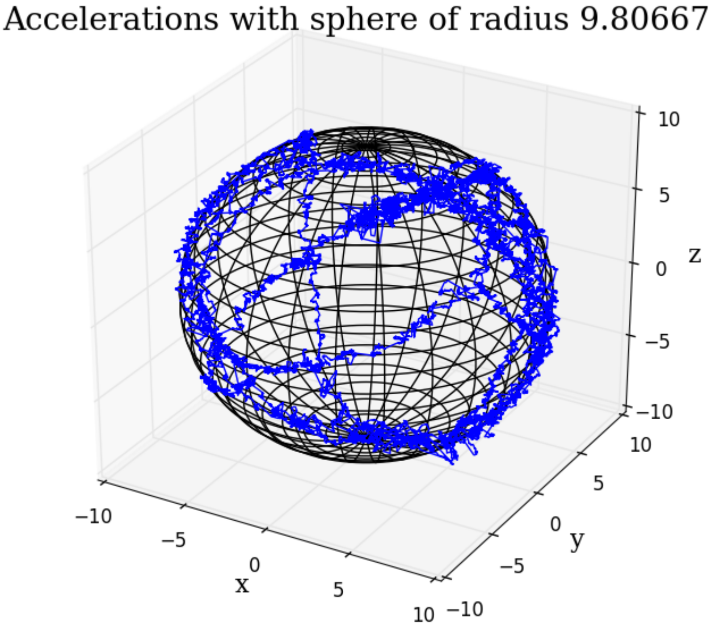
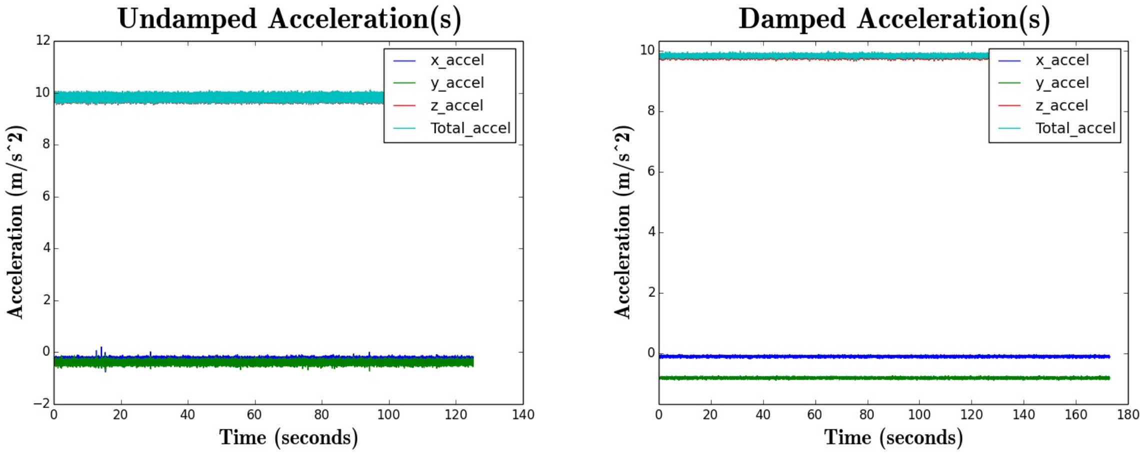


Figure 7.9: Here the calibrated accelerations are plotted in three-dimensional space along with a sphere of radius 9.80667 ms^{-2}.

purpose is the damping of vibrations in air, the same principles still apply.

For an initial test of the efficacy of the vibration damping the device was placed on top of a computer stack while the computer was working at around 100% capacity. The undamped vibrations are shown in Figure 7.10 (a).



(a) (b)

Figure 7.10: Undamped (a) and damped (b) accelerations measured on top of computer running at $\sim$ 100%.

While the graphs shown in Figure 7.10 seem to indicate a significant damping we wanted to ensure this effect was statistically significant. Before each test began, code was run on the computer that would occupy as much of the CPU as possible (specifically it was 3D rendering in OpenSCAD). The code was run for several minutes before each test to ensure that the computer was at a high enough temperature so that the fan was running, thus ensuring similar vibrational conditions for each test.

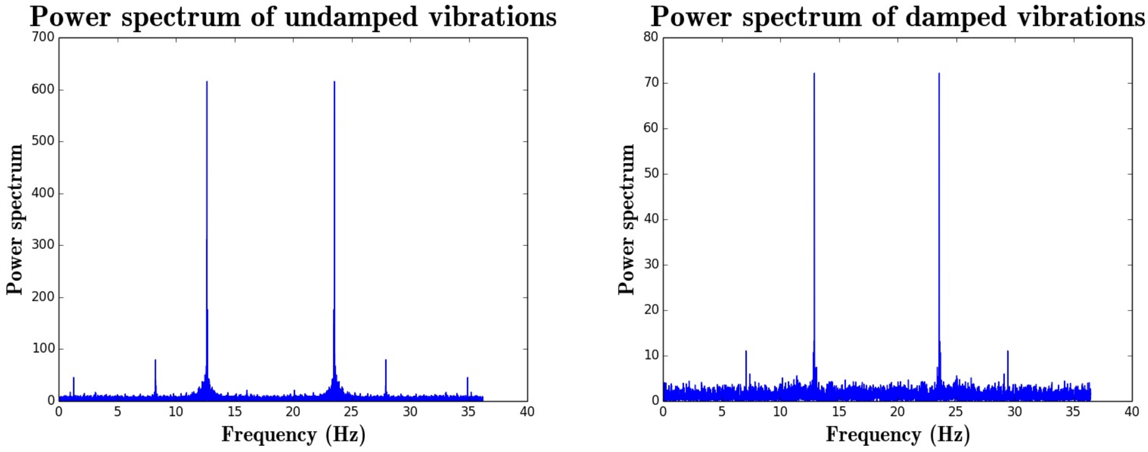
Five data sets for both the damped and undamped case were taken. Each of these data sets consisted of approximately 9000 acceleration measurements, taken on the same day, on the same computer. The device was placed at the same position for each test.

Firstly we wanted to make sure, using the undamped data, that the vibration environments for each test were reasonably similar and the vibrations did not differ by a significant amount from test to test. To do this we looked at the sample variances of the data. Below is a table for the sample variances of both the damped and undamped tests.

	Variance	Sample Length
Undamped1	0.01845214	10111
Undamped2	0.01777448	9456
Undamped3	0.01818243	9346
Undamped4	0.01841536	18643
Undamped5	0.01656852	9133
Damped1	0.001833	9180
Damped2	0.00157797	7316
Damped3	0.00182781	6227
Damped4	0.00234142	12145
Damped5	0.00216810	8104

As you can see the variances of the total acceleration for the undamped case were reasonably consistent across samples with the mean variance being 0.01788 ± 0.00131. There was less consistency present in the damped results, however there is clearly considerably damping. The mean variance in the damped data was 0.001907 ± 0.000434. In terms of variance reduction, then, using the mean of the values above, this constitutes an 89% reduction in variance due to vibrations.

The power spectrum of some of the data, shown in Figure 7.11, indicates the effect of the damping on our results.



(a) (b)

Figure 7.11: The power spectral density of undamped (a) and damped (b) accelerations measured on top of computer running at ∼ 100%. Note the reduction in the height of the main peaks at 12.6 and 23.5 Hz by almost a factor of ten.

While this experimental setup was largely improvised with the materials at hand it does strongly suggest that the ear plug vibration dampeners offered a significant reduction in acceleration vibration noise. A more rigorous test would involve a vibration source capable of vibrating through a wide range of frequencies and comparing the performance of the damped and undamped systems. However the priority for this project was the larger task of implementing the sensor fusion algorithms in the system. A useful contribution in future would be to more rigorously analyse the possible benefits from an ear plug vibration damping mount.

7.3 Stationary Results

Angle estimation is a critical part of the inertial navigation system. Without accurate angle estimation the acceleration vector can have an incorrect magnitude and or direction. This could lead to a combined GPS/INS system that produces worse navigation results than GPS alone.

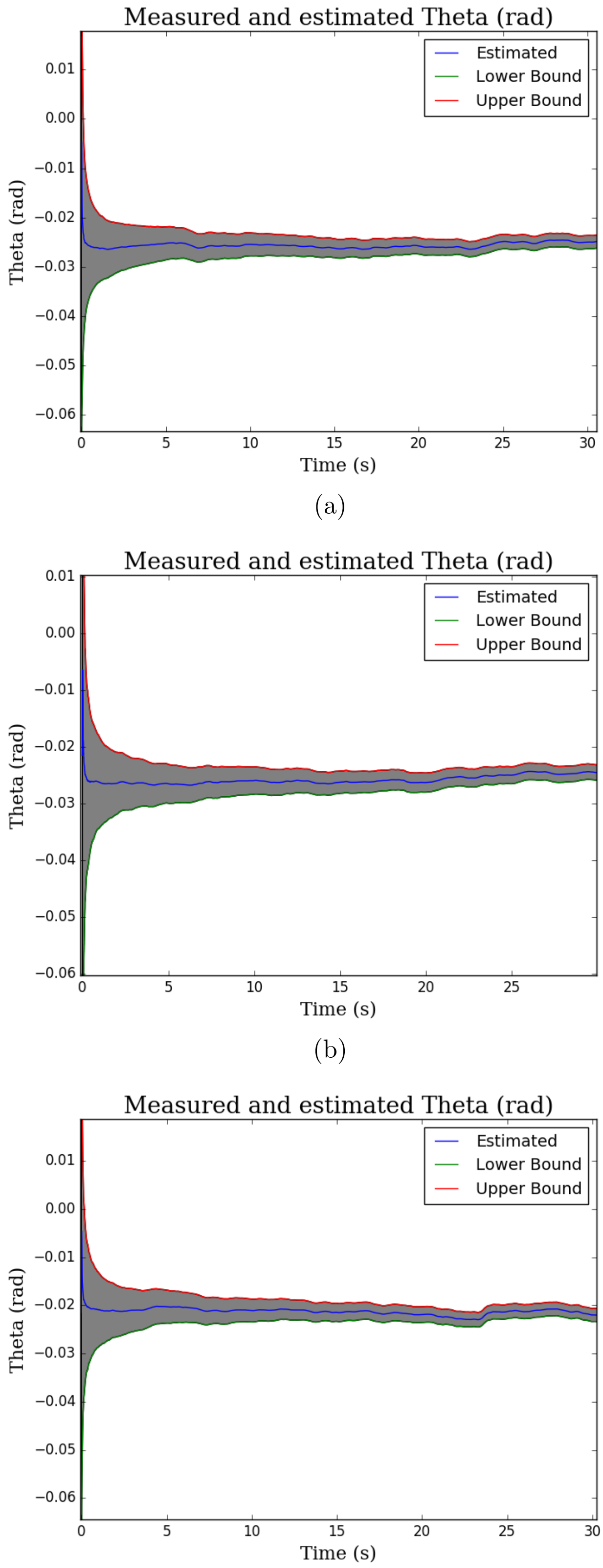
A useful way of parameterising the minimum error in an inertial navigational system is to look at a system where the position, velocity and acceleration are well known and use these to tune the filter accordingly. The simplest situation with a known acceleration, and velocity is, naturally, to have the net acceleration and velocity equal to zero.

Stationary data were taken from the vehicle (2001 Subaru Legacy station wagon) in a parking space. Data was taken for more than an hour to ensure that the position was accurately known, and that plenty of inertial data was available for tuning. During this time the vehicle was running, to provide a minimum process noise for the inertial navigation. By comparing the variances in the inertial measurements with the measurement noise of the devices, calculated during the calibration process in Section 7.2, we were able to obtain a minimum estimate for the inertial process noise due to the vibration of the vehicle.

In the next section we describe an experiment where the same circuit is driven around Dunedin multiple times. The data gathered during these circuits is then filtered using the algorithm outlined in Chapter 6. Before each circuit was driven the vehicle was stationary for at least 30 seconds. Using the process noise estimated from the hour long test, we filtered the data from this stationary period to ensure that the output angles were sensible and reproducible.

In Figure 7.12 we see the initial pitch estimation from three separate data sets, where each data set is taken from the same parking space. Each data set is plotted within a 68% confidence interval. The effect of more measurements is easily seen as the large prior uncertainty shrinks as more measurements are made.

The estimated pitch angles of the three data sets agree to within 0.003 radians, or 0.17 degrees, of each other. The 68% uncertainty in pitch estimation after 30



(a)

(b)

(c)

Figure 7.12: All three figures show the pitch (θ) estimation and 68% confidence interval. The data was taken from the same parking space, on three different occasions. Note how the confidence interval begins at the larger prior assumption and then reduces as more measurements are made.

seconds for these tests was approximately ±0.0015 radians. This means that the confidence intervals overlap with each other. Given the slight variations in wheel placement over a given park we believe these values to be in fair agreement with one another.

We also looked at the roll estimation while stationary. In Figure 7.13 we see the estimated roll angles, again, within 68% confidence intervals. After 30 seconds of measurements the three roll angle estimations are within 0.006 radians, or 0.34 degrees, of each other. This is approximately twice the difference observed in the pitch estimation. Given that the parking space is on relatively flat ground, with a gutter, there is likely to be more variation in the roll angle for a given park within the space, than there is pitch variation.

The 68% uncertainty in roll estimation was approximately ±0.0004 radians. This is significantly smaller that the uncertainty of ±0.0015 radians for the pitch estimation. The source of this larger uncertainty can be found in the measurement equations used for the three axis accelerometer,

$$h(\mathbf{a}) = \begin{bmatrix} a_R \\ a_F \\ a_U \end{bmatrix} = \begin{bmatrix} -g \sin\phi - \omega_z v_F \\ a_T \\ g \cos\theta \cos\phi + \omega_x v_F \end{bmatrix} + \begin{bmatrix} v_R \\ v_F \\ v_U \end{bmatrix} .$$

Note that the rightwards facing accelerometer measures $a_R = -g \sin\phi - \omega_z v_F$. Both the forwards speed (v_F), and the upwards facing angular velocity (ω_z) are measured directly. This allows us to use the information from the rightwards facing accelerometer to accurately estimate the roll (ϕ) term. However the only term in our accelerometer measurement equation that is dependent on the pitch (θ) is the measurements from the upwards facing accelerometer, which also relies on the roll term. Therefore there is more information available about the roll term, and so the uncertainty decreases more quickly than for the pitch estimation.

As mentioned in Equation 6.1 we could have expressed the measurements from the forwards facing accelerometer as the following,

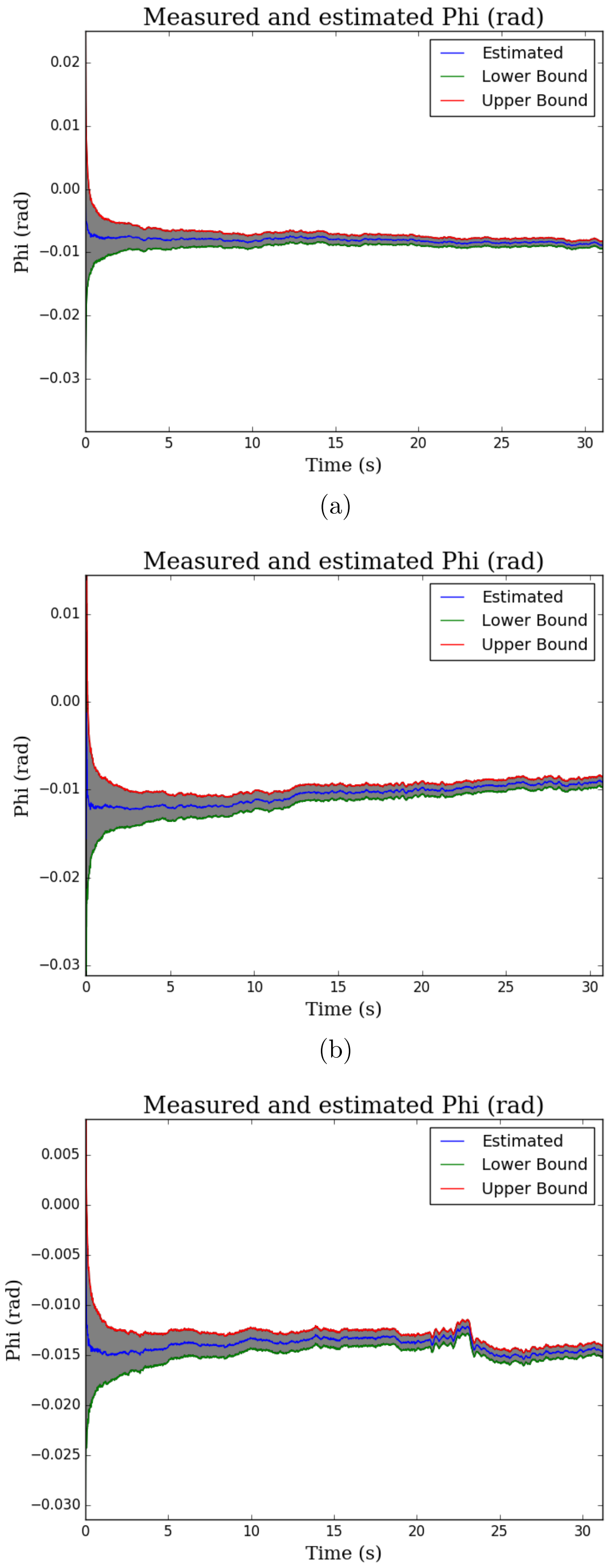
$$a_{meas} = a_F + g \sin\theta \cos\phi.$$

We then could have defined a_F to be zero for this stationary test to obtain a superior estimate of the pitch angle. However we chose to prioritise the estimation of driver input while driving, over the estimation of pitch angle while parked.

The roll estimation is also aided by the small angle approximation, which says the following for small angles,

$$\sin x \approx x, \quad \cos x \approx 1 - x^2. \tag{7.11}$$

Therefore, any acceleration due to a small roll angle will result in an accelerometer measurement that is proportional to the roll angle. However, any acceleration due to



(a)

(b)

(c)

Figure 7.13: Just as in Figure 7.12 all three figures show the angle estimation, in this case roll (ϕ), within a 68% confidence interval. The data was taken from the same parking space, on three different occasions.

a small pitch angle will result in an accelerometer measurement that is proportional to the square of the pitch angle, and so a much smaller signal to identify within the noise. However we believe that angle estimation to within 0.4 degrees should be sufficient for our purposes.

With the inertial noise floor estimated, and the angular estimation software operational we are able to begin the combined GPS and inertial navigation discussed in the following section.

7.4 Experiment

One of the main aims of this project was to improve vehicle navigation by combining GPS, accelerometer, and gyroscope measurements. In order to accomplish this we required real-world data to analyse.

To test our filtering algorithms against each other and against the raw data we drove a circuit around Dunedin while collecting data. To ensure that the data was repeatable the circuit was driven ten times. From this data, six circuits were randomly chosen to be used to estimate the noise present in driving and tune the filter. The filter was then tested with data from the remaining four data sets.

Figure 7.14 plots the circuit that was used for our experiment, with the raw GPS data from our test rig, which was sampling at 1Hz.

From Figure 7.14 it is clear that just GPS data does a reasonable job of position estimation. Indeed for the straight sections, particularly the long section on the left hand side of the circuit, one can tell which lane the vehicle was in (for concerned readers that street is a two lane, one-way street).

However there is room for improvement. In Figure 7.15 we see a zoomed-in view of the circuit at the point of a hairpin bend. Plotted are three separate GPS tracks around the corner, we can clearly see that the GPS data has trouble accurately measuring position when the vehicle is undergoing large changes in direction.

Furthermore, as we have mentioned previously if a GPS unit does not have line of sight to a minimum of four satellites, it cannot get any new data. In the following section we demonstrate the effects of a simulated loss of signal by removing the GPS data for short intervals and compare the performance of an integrated system versus a system that solely relies on GPS data.

7.5 Simulated GPS outages

A large motivation for this project was to create a navigation system that could bridge short GPS outages by using accelerometer and gyroscope based inertial navigation. Therefore using the data from the experiment we simulated a total loss of

Figure 7.14: The circuit began on the lower right-hand side of the map and was driven anti-clockwise. After the 90° right hand turn near the center of the image the map ascends a hill, before reaching the peak and beginning the descent at the sharp left hand turn at the top right of the circuit. Note that this is a plot of the raw GPS data, not the filtered data.

Figure 7.15: The raw GPS data from three different circuits is shown here. In all three cases one can clearly see that the unaugmented GPS data has trouble when a vehicle changes directions quickly around sharp corners. Note that from this perspective the vehicle enters from the left hand side, driving on the left hand side of the road up to the top of the image.

GPS signal for various intervals and compared the filters using just GPS and GPS and IMU data to see the effect of the outage on position estimation.

In Figure 7.16 we see the results of a GPS based navigation system losing satellite signal for fifteen seconds. As it has no redundancy system, in the event of total GPS signal loss the only courses of action are either assuming no position change (i.e. stay at the last estimated latitude and longitude until a new fix is available), or assuming no state change (i.e. keep the same velocity and heading and update latitude and longitude accordingly until a new fix is available).

Figure 7.16: A fifteen second loss of GPS signal was simulated. The blue track shows the results from a solely GPS-based navigation system, operating on the assumption of a constant speed and heading between samples. The red track is the GPS signal with the fifteen seconds of data removed.

In Figure 7.17 we see the results of the same loss of GPS signal as used in Figure 7.16 when using a combined GPS and inertial navigation system.

As you can see in Figure 7.17, the combined GPS and inertial navigation system compensates for the short term loss of GPS signal reasonably well. The system shows the vehicle turning a corner, although running slightly wider than the true corner. The points shown in Figure 7.17 for the combined system is the estimated position shown every second. The GPS device used is able to take measurements

Figure 7.17: The same fifteen second simulated GPS signal loss but with an integrated GPS and IMU system. The combined system with the simulated signal loss is the blue path while the red path is the original GPS signal.

every second and these are plotted with the fifteen second gap removed from the measured data.

To best compare the combined and solo GPS navigation systems we plotted the performance of both in Figure 7.18. We see in Figure 7.18 that the combined system clearly deals with the loss of GPS signal better. We naturally analysed the performance of the filter for different test cases and different data sets to ensure that the performance as seen in Figures 7.16 - 7.18 was not an isolated case of superior performance. Another example of this is shown in Figure 7.19.

Figure 7.18: Both sets of data here had the same fifteen second period of GPS measurements removed to simulate a loss of GPS signal. The track in blue shows the results of such a loss of signal on a GPS-only navigational system while the track in red shows the results of such a loss of signal on a combined GPS and IMU navigational system.

The combined inertial navigation and GPS filter also performed well in the presence of linear accelerations in the absence of GPS signals. In Figure 7.20 we see the results of a fifteen second simulated loss of GPS signal just before the vehicle began braking for a traffic light. While the combined system (shown in blue) does overshoot slightly in terms of the stopping position it far out-performs the filter

Figure 7.19: The effects of losing signal at another corner. Again the GPS measurements used by the filter are plotted in blue, while the filter output is plotted in red.

based solely on GPS measurements, shown in green in Figure 7.20.

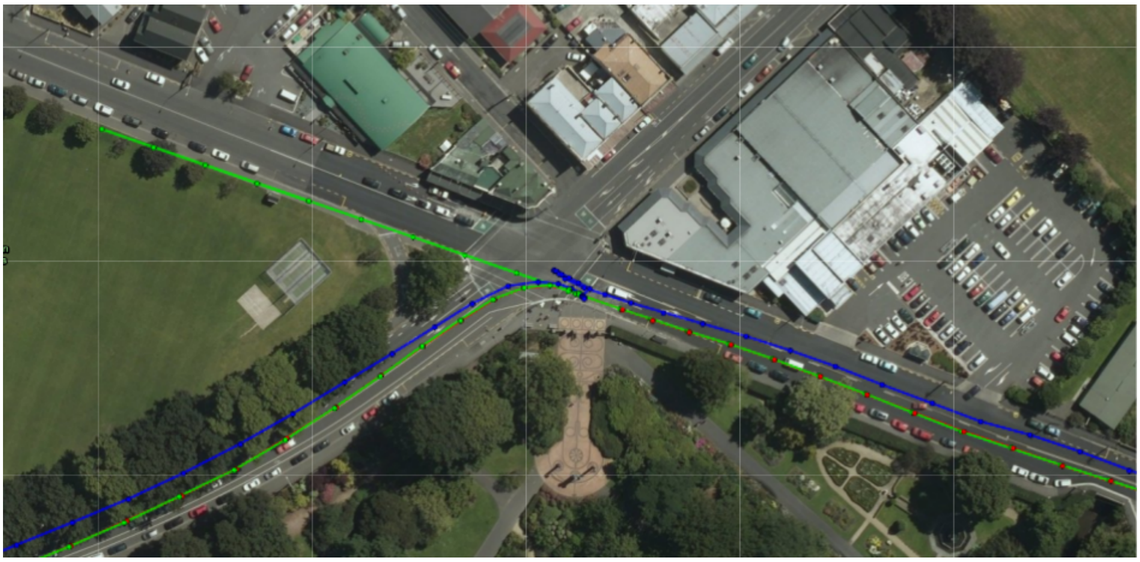


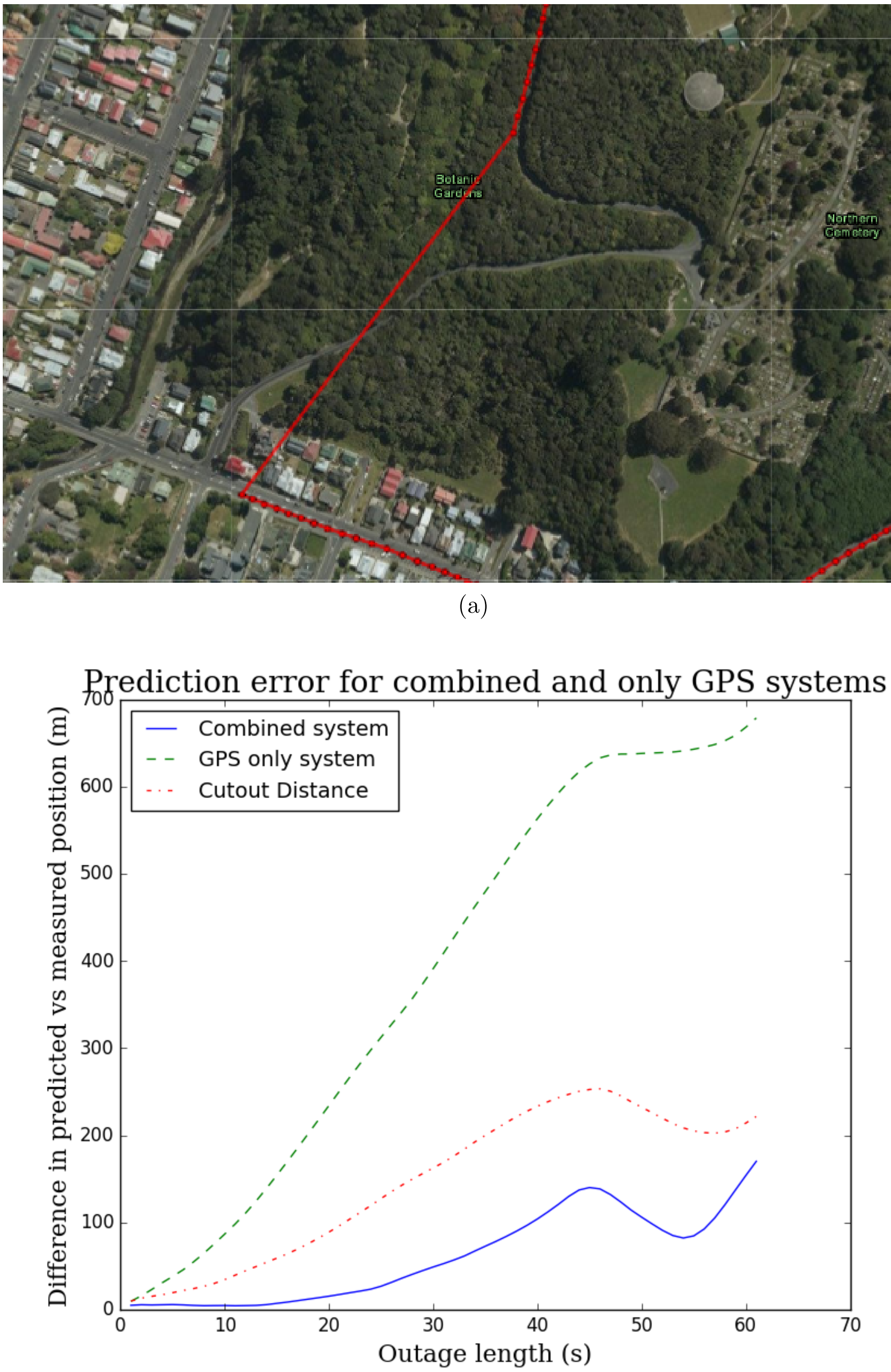
Figure 7.20: The filter loses GPS signal for fifteen seconds just as the vehicle is braking for a set of traffic lights. The path in blue is the combined GPS and inertial navigation filter, the path in green is the results of a solely GPS-based system.

While we were pleased with how the system performed without GPS, it is clearly not perfect. The filter appears to perform worse when accelerating quickly or cornering sharply as shown in Figure 7.21 where the vehicle was travelling downhill and navigating sharp corners.

While the results thus far have seemed reassuring a quantitative measure of filter performance during GPS outages has been missing. We will endeavour to show the quantitative benefit of using a combined GPS and IMU model by varying the outage duration at various points and observing how far the filter was from the first GPS measurement after the outage. We will compare the combined GPS and IMU filter against a GPS only filter that assumes constant velocity and heading. We will compare the filters across areas with sharp corners, shallow corners, and no corners to determine the what benefits a combined system has across a range of terrains. For brevity we simply look at the horizontal error rather than heading, velocity or altitude errors, as the horizontal error is the most important for our vehicle navigation problem.

In Figure 7.22 we see the horizontal error present in a combined GPS and inertial navigation system and a solely GPS-based system that assumes constant heading and speed between GPS measurements. The x axis in Figure 7.22(b) is the outage time in seconds, and the total distance covered over that outage is shown in Figure 7.22 in red.

Figure 7.21: When undergoing hard cornering or large accelerations the filter has some difficulty correcting quickly enough. As seen during this 30 second interval.



(a)

(b)

Figure 7.22: (a) The route cutout for the maximum output. Note the initial 90° corner as well as the later corner. (b) The error in latitude and longitude (m) in blue, the horizontal distance the outage covers in red and the horizontal error in a system that only uses GPS and assumes constant heading and speed between GPS measurements in green.

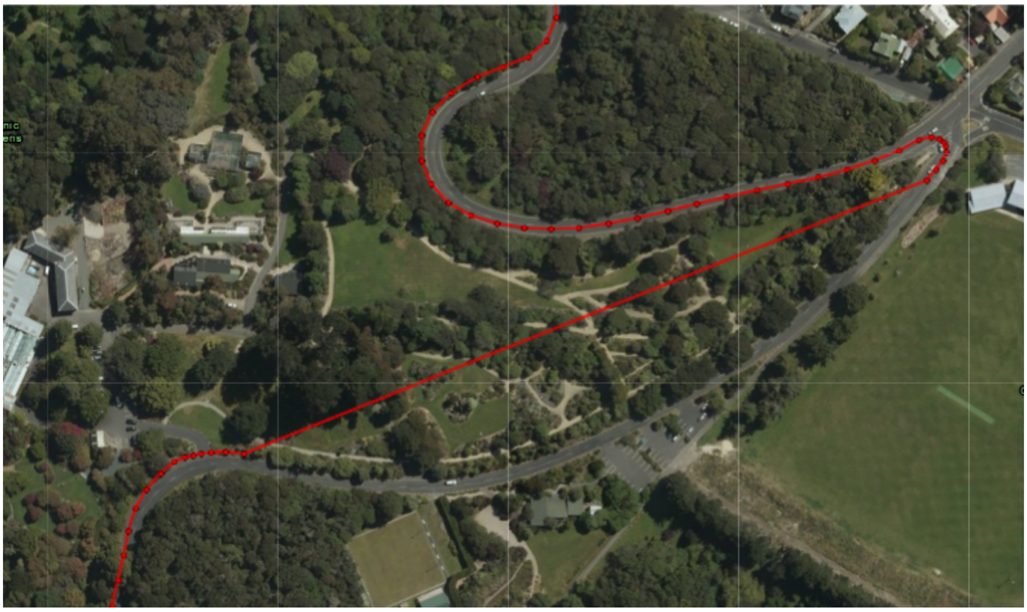
The route shown in Figure 7.22 involves two sharp corners and one shallow corner. As we can see from Figure 7.22(b) the error in the system based solely on GPS measurements is, predictably, larger than that of the combined system. In fact, it is also significantly larger than the distance covered over the blackout, i.e. in this instance it would have been better to only rely on the GPS measurements and not assume that the vehicle maintains speed and heading between measurements.

The route covered in the cutout in Figure 7.23 is dominated by one long, shallow corner. Thus, so we would expect the system that only uses GPS measurements to perform better in this cutout than in Figure 7.22, and indeed that is what we see in Figure 7.23 (b). In this case there is approximately equal error in assuming constant speed and heading during the outage versus assuming that the vehicle is simply at the last measured data point. However, as in Figure 7.22, the combined GPS and inertial navigation system has superior performance to that of a system that only uses GPS measurements.

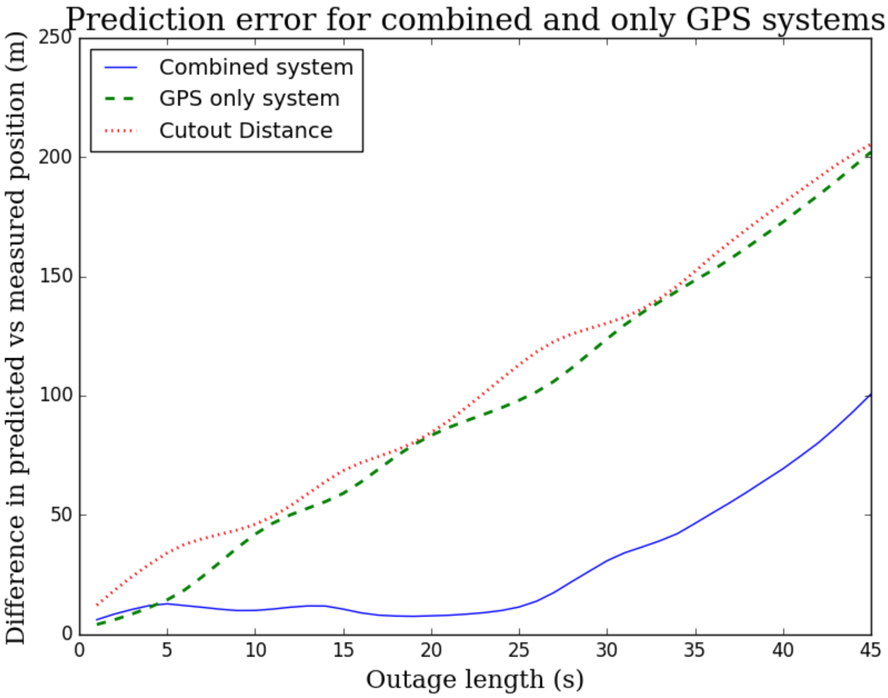
During the GPS outage shown in Figure 7.23 there were a series of three speed bumps, which can be seen in the cutout distance as gradient changes as the vehicle slowed down for the speed bump and then accelerated afterwards. These variations in speed caused the GPS-only system some difficulties, however, as the changes in speed were not very large and quite symmetric the performance of the GPS-only system was still similar when compared to outage distance.

Finally Figure 7.24 shows the effect of a GPS cutout where there are no corners in the outage period. Again we would expect that the GPS-only system would perform better than previously, and again that is what we see. Indeed for outages of up to 25 seconds the assumption that the vehicle is maintaining speed and heading has less error than assuming that the vehicle is stationary at the last measured position. However, at this point in the route the vehicle stopped for a traffic light, and thus the assumption of constant speed and heading still produced considerable errors. We can see that the combined system again had superior performance against the GPS-only system, even in a situation where the GPS-only system should perform well.

So we can see that in the event of GPS outages the combined GPS and inertial navigation system out-performs a solely GPS-based navigation system. Whether that system assumes constant speed and heading between measurements or only uses measured data and makes no assumptions. The bridging of short-term GPS outages is especially useful for vehicles operating in urban areas that can be affected by urban canyons and lose GPS signal for short periods of time [60, 61].



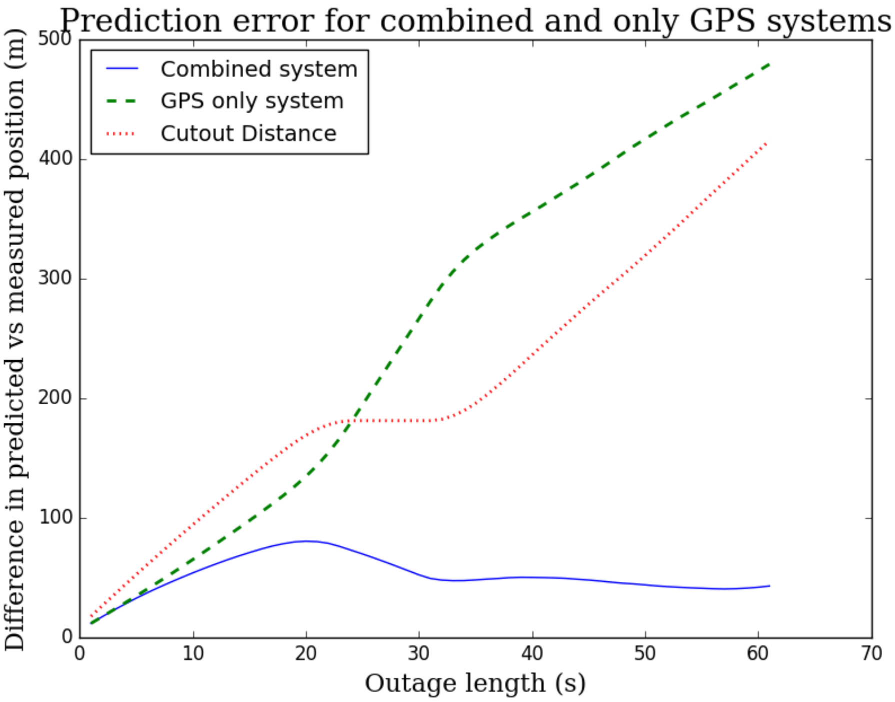
(a)



(b)

Figure 7.23: (a) The route cutout for the maximum output. Note the smooth, shallow corner covered during the maximum outage. (b) The error in latitude and longitude (m) in blue, the horizontal distance the outage covers in red and the horizontal error in a system that only uses GPS and assumes constant heading and speed between GPS measurements in green.

(a)



(b)

Figure 7.24: (a) The route cutout for the maximum output. Note that for the entire cutout there are no corners and during driving the vehicle did not change lane. (b) The error in latitude and longitude (m) in blue, the horizontal distance the outage covers in red and the horizontal error in a system that only uses GPS and assumes constant heading and speed between GPS measurements in green.

Chapter 8

Conclusion

This project investigated the use of different models to extract better information from single frequency GPS devices. The first approach to this was to investigate the fit of different noise models to the noise in single-frequency GPS. The second approach was to use sensor fusion techniques to combine GPS and inertial sensors to provide better vehicle navigation solutions.

In Chapter 3 we analysed the noise observed in GPS signals from two different types of devices, using data taken from four locations, at various latitudes. Using the Akaike Information Criterion (AIC) we compared the goodness of fit of each of our proposed noise models. We found that for both devices, and across every location, that the independent, identically distributed Gaussian (i.i.d. Gaussian) noise model was the worst fit to the data out of all of the candidate models. We found that the higher order autoregressive or mixed autoregressive, moving average noise models provided the best fit to the data. Following this, in Chapter 4 we demonstrated how one might filter GPS data using the Ornstein-Uhlenbeck noise model, a model that fit the empirical data better than the i.i.d. Gaussian noise model.

In Chapter 6 we defined a realistic vehicle model that was implement in Chapter 7. By placing constraints on the possible movements of the vehicle (no tyre slip, and the vehicle was always in contact with, and perpendicular to, the road) we were able to introduce redundant measurements in our system, and so decrease the uncertainty in our state estimation. When implemented, our vehicle navigation algorithm allowed a real-time estimation of driver input, and proved able to bridge short-term losses of GPS signal by inertial navigation.

8.1 Noise models for GPS data

Previous work has mentioned the problem of time dependent noise in single frequency GPS units [3–5]. We investigated the noise present in GPS measurements and compared the fit of several time dependent noise models to the observed noise. We compared the performance of noise models proposed by others, as well as the Ornstein-Uhlenbeck model, which we believe is a novel addition to the literature. Measurements were made at four locations using two different types of GPS device. The results of these were used as data to compare the relative performance of the models within our candidate set.

The confirmation of the work of other authors, and of anecdotal evidence from our own previous work with GPS data would have justified the experiment performed in Chapter 2, without any further goals. However, during the process of the experiment we realised that the autoregressive models provided a convenient noise model that performed significantly better than the i.i.d. Gaussian model and allowed for simple parameter estimation and filtering.

In Chapter 4 we showed how to incorporate an autoregressive process of arbitrary order into a standard Kalman filter for further state estimation, however in order to implement such a system a constant sampling period would be needed. Such constant sampling GPS units are very common and cheaply available (the GlobalSat BU-353 used in this thesis was less than $US30). However we also showed the results of filtering using the Ornstein-Uhlenbeck noise model, which can easily filter GPS data with non-constant sampling periods, for example, GPS units that sample from a warm start rather than a hot start.

8.2 Added benefit of using inertial navigation

We saw in Chapter 7 that a modern GPS unit sampling at 1Hz can accurately plot the path of a vehicle. However by combing a GPS and inertial measurement unit (IMU) we demonstrated that we can bridge short-term outages in the GPS signal by relying on inertial navigation. This is a very relevant problem to solve as such outages are common when navigating through areas with imperfect skyview, either because of local geography, foliage, or urban canyons. The addition of the IMU requires a nonlinear filter and therefore more computational time, however we believe that these costs are more than outweighed by the navigational benefits. As inertial sensors and processors become cheaper and more readily available such filtering will be easily implemented on various hardware platforms. The reader's smartphone almost certainly has the required sensors and processing power to run the navigation algorithms used in Chapter 7 to improve their own navigation.

A substantial part of Chapter 7 was dedicated to using inertial navigation to bridge short term outages in the GPS signal. This work is particularly applicable for use in inner-city navigation where urban canyons can temporarily disrupt the line-of-sight to the GPS satellites. As shown in Chapter 7 our system can cope with sharp cornering, and linear accelerations during periods with no GPS signal. Therefore if a vehicle loses GPS signal while driving in the city our system can tell if it turns a corner, or accelerates. Given the prevalence of GPS navigation in large cities we believe that such an application could be extremely useful for both commercial and private applications.

8.3 Future Work

There were several areas that we looked into that we feel merit further investigation. In the context of fitting noise models to GPS noise we only had access to limited amounts of data. While every effort was made to ensure the general applicability of our results a valuable area of future work would be to gather significantly larger data sets from across the globe and by applying the same selection criteria and time series modeling as detailed in Chapter 2 one could demonstrate with strong certainty which noise models fit the errors seen in GPS signals across the globe.

In Chapter 4 we proposed a way of filtering moving GPS data using an Ornstein-Uhlenbeck noise model. Unfortunately we did not have time to investigate this idea, however given the two main aims of this thesis, this would be a natural extension to the work herein described.

Later in Chapter 5 we introduce the idea of inertial navigation with the example of position estimation using accelerometer measurements in an elevator. We believe that this example could easily be expanded to provide a useful lab experiment for an undergraduate course. It combines data collection, filtering, and model creation in a very intuitive and familiar way. Also the required materials are readily available in most (multi-story) physics or engineering buildings.

We mentioned in Chapter 6 that GPS devices report the number of satellites being used by the device and also the Dilution of Precision (DOP) values associated with their current positions in the sky. Using this information to create a dynamic covariance matrix, designed for moving vehicles that is capable of increasing and decreasing the covariance depending on the satellites positions would be a useful contribution to the literature. However such an experiment would require a moving platform with a precisely known position, an experimental setup that we did not have the time to implement.

Another valuable contribution to this project would be to integrate the vehicle navigation algorithm with accurate street maps. By assuming that while the vehicle was in motion it was on a road, a minimization algorithm could be used to place it at the most likely road. The inclusion of accurate street maps with the algorithm could, theoretically, further improve the accuracy of the algorithm, especially when operating without GPS signal, where the uncertainty in the position can grow quickly.

Our initial intention with the nonlinear filtering described in Chapter 6 was to write the nonlinear model in such a way that it could be filtered by either an Extended Kalman filter (EKF) or an Unscented Kalman filter (UKF). The results of these two could then be compared against each other in terms of filter performance and computational expense. However when filtering larger data sets we ran into problems maintaining positive definite matrices when filtering with the UKF and so the results presented in Chapter 7 are solely those obtained by the EKF. There is reason to suspect that the UKF would provide superior filter performance for more nonlinear aspects of the filtering process [23, 26]. Given this implementing a version of the UKF which maintained positive semi-definiteness would be a valuable contribution to the work of this project.

Bibliography

Bibliography

[1] Francois Caron, Emmanuel Duflos, Denis Pomorski, and Philippe Vanheeghe. GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects. *Information Fusion*, 7(2):221–230, 2006.

[2] Andrew A. Thompson. A Ballistic Filter for GPS and Accelerometer Measurements. *Army Research Laboratory*, April 2010.

[3] Ailin Mao, Christopher G. A. Harrison, and Timothy H. Dixon. Noise in GPS coordinate time series. *Journal of Geophysical Research*, 104(B2):2797–2816, February 1999.

[4] Alfred Leick, Lev Rapoport, and Dmitry Tatanikov. *GPS Satellite Surveying.* John Wiley & Sons, 2015.

[5] L. Li and H. Kuhlmann. Real-time deformation measurements using time series of GPS coordinates processed by Kalman filter with shaping filter. *Survey Review*, 44(326):189–197, July 2012.

[6] Anon. Global positioning system standard positioning service performance standard. Technical report, USA Department of Defense and GPS NAVSTAR, September 2008. 4th Edition.

[7] Paul D. Groves. *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. Artech House, 2008.

[8] Weidong Ding and Jinling Wang. Precise Velocity Estimation with a Stand-Alone GPS Receiver. *Journal of Navigation*, 64(2):311–325, 2011.

[9] Youjing Cui and Shuzhi Sam Ge. Autonomous vehicle positioning with GPS in urban canyon enviroments. *Robotics and Automation, IEEE Transactions on*, 19(1):15–25, 2003.

[10] Freescale Semiconductor. *Xtrinsic FXOS8700CQ 6-Axis Sensor with Integrated Linear Accelerometer and Magnetometer Datasheet*, 4 edition, 03 2014.

[11] Anon. Official U.S. government information about the Global Positioning System (GPS) and related topics.

[12] Anon. Official GLONASS Website.

[13] Anon. BeiDou Navigation Satellite System Official Website. Online.

[14] Rui C. Barbosa. Long March 3C launches Beidou-21 satellite.

[15] Anon. Official Galileo Website.

[16] Aboelmagd Noureldin, T. Ben Karmat, and Jacques Georgy. *Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration.* Springer, 2013.

[17] William J. Hughes Technical Center. Global Positioning System (GPS) Standard Positioning Service (SPS) Performance Analysis Report. Technical Report 86, Federal Aviation Administration, July 2014.

[18] Pratap Misra and Per Enge. *Global Posiitioning System Signal, Measurements, and Performance.* Ganga-Jamuna Press, 2 edition, 2006.

[19] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME, Series D: Journal of Basic Engineering*, 82:35–45, 1960.

[20] R. E. Kalman and R. S. Bucy. New Results in Linear Filtering and Prediction Theory. *Journal of Fluids Engineering*, 83(1):95–108, March 1961.

[21] Maria Isabel Ribeiro. Kalman and Extended Kalman Filter: Concept, Derivation and Propagation. Via University of Lisbon, Institute for Systems and Robotics Instituto Superior Tecnico Av. Rovisco Pais, 1 1049-001 Lisboa PORTUGAL, February 2004.

[22] James Douglas Hamilton. *Time series analysis*, volume 2. Princeton university press, 1994.

[23] S.J. Julier and J.K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92:401–422, March 2004.

[24] Rudolph van der Merwe and Eric A. Wan. The square-root unscented Kalman filter for state and parameter-estimation. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 6. IEEE, 2001.

[25] Rudolph van der Merwe. *Sigma-point Kalman filters for probabilistic inference in dynamic state-space models.* PhD thesis, University of Stellenbosch, 2004.

[26] Mathieu St-Pierre and Denis Gingras. Comparison between the unscented Kalman filter and the extended Kalman filter for the position estimation module of an integrated navigation information system. *Intelligent Vehicles Symposium, 2004 IEEE*, 2004.

[27] Fredrik Orderud. Comparison of Kalman filter estimation approaches for state space models with nonlinear measurements. *Proc. of Scandinavian Conference on Simulation and Modeling.*, 2005.

[28] N. J. Gordon, D. J. Salmond, and A. F. M. Smithordon. Novel Approach to Nonlinear/non-Gaussian Bayesian state estimation. *IEEE Proceedings F (Radar and Signal Processing)*, 140(2):107–113, 1993.

[29] Manya Afonso. Particle filter and extended kalman filter for nonlinear estimation: a comparative study. *IEEE Transactions on Signal Processing*, 10, 2010.

[30] John L. Crassidis, Landis Markley, and Yang Cheng. Survey of nonlinear attitude estimation methods. *Journal of Guidance, Control, and Dynamics*, 30(1):12–28, 2007.

[31] Elliot D. Kaplan and Christopher J. Hegarty. *Understanding GPS Principles and Applications*. Artech House, 2 edition, 2006.

[32] Mamoun F Abdel-Hafez. The autocovariance least-squares technique for GPS measurement noise estimation. *Vehicular Technology, IEEE Transactions on*, 59(2):574–588, 2010.

[33] Kia Fallahi, Chi-Tsun Cheng, and Michel Fattouche. Robust positioning systems in the presence of outliers under weak GPS signal conditions. *Systems Journal, IEEE*, 6(3):401–413, 2012.

[34] Jonathan D. Cryer and Kung-Sik Chan. *Time series analysis: with applications in R*. Springer, 2008.

[35] Chris Chatfield. *Time-series forecasting*. CRC Press, 2000.

[36] William H. Jeffreys and James O. Berger. Ockham's razor and Bayesian analysis. *American Scientist*, 80(1):64–72, Jan 1992.

[37] Mr. Bayes and Mr. Price. An essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfrs. *Philosophical Transactions*, pages 370–418, Jan 1763.

[38] Hirotogu Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control,*, 19(6):716–723, 1974.

[39] Asger Roer Pedersen. A new approach to maximum likelihood estimation for stochastic differential equations based on discrete observations. *Scandinavian journal of statistics*, pages 55–71, 1995.

[40] Hamparsum Bozdogan. Model selection and Akaike's information criterion (AIC): The general theory and its analytical extensions. *Psychometrika*, 52(3):345–370, 1987.

[41] John Rice. *Mathematical statistics and data analysis*. Cengage Learning, 2006.

[42] George E. Uhlenbeck and Leonard S. Ornstein. On the theory of the Brownian motion. *Physical review*, 36(5):823, 1930.

[43] Peter E. Kloeden and Eckhard Platen. *Numerical solution of stochastic differential equations*, volume 23. Springer, 1992.

[44] Oldrich Vasicek. An equilibrium characterization of the term structure. *Journal of financial economics*, 5(2):177–188, 1977.

[45] Susanne Ditlevsen. Statistical methods for diffusion models. Example: the Ornstein-Uhlenbeck process. Lecture notes from Prof Ditlevsen's graduate course on Statistical methods for diffusion models.

[46] Bradley Panckhurst. Power Efficient, Telemetry-Enabled Position Sensor for Animal Tracking. Master's thesis, University of Otago, 2016.

[47] Keith Payne. *Development of Two New GPS Avian Tracking Systems*. PhD thesis, University of Otago, 2013.

[48] Anon. *Datasheet for USB GPS Receiver : BU-353S4(Datasheet*. GlobalSat Technology Corporation, September 2013.

[49] David Titterton and John L. Weston. *Strapdown Inertial Navigation Technology*, volume 17. IET, 2004.

[50] A. D. King. Inertial navigation-forty years of evolution. *GEC review*, 13(3):140–149, Jan 1998.

[51] Anon. Department of Defense World Geodetic System 1984-its definition and relationships with local geodetic datums. Technical report, National Imagery and Mapping Agency, 1997. (Amendment 1, 3 Jan. 2000; Amendment 2, 23 June 2004).

[52] Geographic coordinate system.

[53] Jack B Kuipers. *Quaternions and rotation sequences*. Princeton university press, 1999.

[54] Yaakov Bar-Shalom, X. Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation*. Wiley-Interscience, 2001.

[55] Anon. *Raspberry Pi Model B+ Datasheet*. Raspberry Pi Foundation.

[56] Anon. *Alamode V1.0 Datasheet*. Wyolum.

[57] Freescale Semiconductor. *3-Axis Digital Angular Rate Gyroscope, FXAS21002C, Datasheet*, 2.0 edition, 02 2015.

[58] Naser El-Sheimy, Haiying Hou, and Xiaoji Niu. Analysis and modeling of inertial sensors using Allan variance. *Instrumentation and Measurement, IEEE Transactions on,*, 57(1):140–149, 2008.

[59] Pin Lv, Jianye Liu, Jizhou Lai, and Kai Huang. Allan variance method for gyro noise analysis using weighted least square algorithm. *Optik-International Journal for Light and Electron Optics*, 126(20), 2015.

[60] Charles Vicek, Patricia McLain, and Midhael Murphy. GPS/dead reckoning for vehicle tracking in the" urban canyon" environment. *Vehicle Navigation and Information Systems Conference, 1993., Proceedings of the IEEE-IEE.*, pages 461–34, 1993.

[61] S. Godha and M. E. Cannon. GPS/MEMS INS integrated system for navigation in urban areas. *Gps Solutions*, 11(3):193–203, Jul 2007.

Appendices

A.1 Papers and Conferences

During this project a large effort was made to present work and interact with the larger academic and industrial communities. As such the author presented parts of the thesis at three separate conferences and workshops, the IEEE Instrumentation and Measurement Workshop (2014), ENZCon 2014, and The 6th International Conference on Automation, Robotics and Applications (ICARA 2015), the conference papers of which were peer reviewed and are available online at the databases for the separate conferences.

During the course of this project the author was also involved in other events for general collaboration with the various post graduate student communities. One such event was the JENESYS 2.0 program, a cultural exchange where a group of students from New Zealand received sponsorship from the Japanese government to go to Japan to visit industrial sites, university laboratories and stay with local host families.

Another such event, slightly closer to home, was the inaugural Electro-Hui between the Victoria University Engineering faculty and the University of Otago, Electronics Research Group. This event was conceived as a slightly more informal conference for postgraduate students from each of the universities to present their research to fellow students.

During the course of this project the author also helped write two papers, of which the first has been submitted to the peer-reviewed “GPS Solutions” journal. The second paper is still being worked on before submission.

A.2 Open source note

Throughout this project a concerted effort to only use open-source software was undertaken. This was done in an effort to support the numerous open-source projects that have sprung up in recent years.

While MatLab is an extremely popular tool in some areas of programming an modeling a license can be expensive for institutions or individuals. By far the majority of the programming in this project was done in Python, often using the NumPy, SciPy and MatPlotLib packages which provide much of the functionality of MatLab including ode and pde solvers, vector and matrix operations, various inbuilt filters and the ability to easily make customisable figures and plots.

The electronic design automation (EDA) program EAGLE will be familiar to any readers familiar with designing and building printed circuit boards (pcb’s). In recent years the EDA program KiCad has become widespread enough that the software now has a large, dedicated team which have provided a free program with functionality that is comparable to EAGLE. KiCad offers many similar tools to EAGLE, including a schematic editor, PCB layout program (with 3D view), footprint selector, etc.

Furthermore the support structure for the eventual unit was designed in OpenSCAD, an open-source application for creating 3D CAD objects. Unlike applications like SolidWorks which are interactive modelers OpenSCAD is a textual description language which is written in a very similar way to how one might write a piece of code in C or Java. This makes models very easy to modify and adapt to different requirements or parameters.

The hardware used in this project of an Arduino plate and Raspberry Pi base, combined with accelerometer, gyroscopic and GPS sensors. The sensors constituted the only major break with the open-source effort of this project as, given the low cost of most common sensors and the high costs involved in developing and producing them, an open-source alternative to the current commercially driven options is, at the very least, a long way away. Alternatively the Arudino Alamode plate which communicated between the Raspberry Pi and the accelerometer and gyroscope, like all Arduino products the Alamode is open-source hardware and freely available on the Arduino website. The Raspberry Pi was originally developed by a team based at the University of Cambridge who were disappointed in the decline in numbers and skill of the students applying to read Computer Science. While the current iteration of the board is not open-source hardware, the schematics for previous editions of the board are available on the Raspberry Pi website.

A.3 Rotation formalisms

A.3.1 Rotation Matrices

When performing a rotation on a vector the most familiar method to the majority of people is the rotation matrix. A rotation matrix is characterised as an orthogonal matrix with real entries and a determinant of 1. Sometimes improper rotation matrices with determinants of -1 are included, however these combine proper rotations and reflections which are not possible with a rigid physical object and so will be ignored for the rest of this thesis.

Being orthogonal matrices the transpose of a rotation matrix is the inverse of that matrix and so represent the same rotation in the opposite direction.

$$R^T = R^{-1}. \tag{A.1}$$

Combining subsequent rotations is a simple matter of multiplying the two rotation matrices.

$$R_3 = R_2 R_1. \tag{A.2}$$

However it is worth noting that rotations are not commutative, i.e. the order of rotation matters. Therefore, in general,

$$R_2 R_1 \neq R_1 R_2. \tag{A.3}$$

When using rotation matrices to convert some vector $\mathbf{x}$ from reference frame B to reference frame E we will adopt the following convention.

$$\mathbf{x}^E = R_B^E \mathbf{x}^B. \tag{A.4}$$

Where the reference frame that the vector $\mathbf{x}$ is in is denoted by its superscript. The subscript of the matrix denotes the reference frame $\mathbf{x}$ is being transferred *from* and the superscript of the matrix denotes the reference frame that $\mathbf{x}$ is being transferred *to*.

Rotation matrices are a familiar, easy-to-use method for transforming between reference frames. However the 3D rotation group SO(3) only has three degrees of freedom and so the rotation matrix has six redundant entries. Another way of rotating three-dimensional vectors is by quaternion multiplication.

A.3.2 Quaternion Rotations

The quaternions are a hypercomplex number set which were first described by William Rowan Hamilton in 1843. While standard complex numbers just have one imaginary index, usually denoted i, quaternions have three, denoted i, j, k. For example, a general expression for a quaternion ($\bar{\mathbf{q}}$) is given below.

$$\bar{\mathbf{q}} = a + bi + cj + dk. \tag{A.5}$$

Just as you multiply complex numbers using the identity $ii = -1$, you multiply quaternions with the following identities.

$$i^2 = j^2 = k^2 = ijk = -1$$

$$\begin{aligned} ij &= k, & ji &= -k, \\ jk &= i, & kj &= -i, \\ ki &= j, & ik &= -j. \end{aligned} \tag{A.6}$$

From the identities above it is clear that quaternion multiplication, like the multiplication of rotation matrices, is non-commutative.

Below is a worked example of quaternion multiplication, denoted by the $\otimes$ symbol.

$$\begin{aligned} &(a_1 + b_1 i + c_1 j + d_1 k) \otimes (a_2 + b_2 i + c_2 j + d_2 k) = \\ &(a_1 a_2 - b_1 b_2 - c_1 c_2 - d_1 d_2) + \\ &(a_1 b_2 + b_1 a_2 + c_1 d_2 - d_1 c_2) i + \\ &(a_1 c_2 - b_1 d_2 + c_1 a_2 + d_1 b_2) j + \\ &(a_1 d_2 + b_1 c_2 - c_1 b_2 + d_1 a_2) k. \end{aligned} \tag{A.7}$$

Importantly for our purposes the group of unit quaternions can span the hypersphere of the 3D rotation group SO(3). This can easily be seen when viewed in the context of Euler's formula and Euler's rotation theorem. According to Euler's rotation theorem any rotation or series of rotations of a rigid body or coordinate system about a fixed point is equivalent to a single rotation by an angle, known as the Euler angle (θ) around a fixed axis, known as the Euler axis and typically represented by a unit vector ($\hat{\mathbf{u}}$).

Euler's formula allows us to express a complex number as a complex exponential. Likewise we can also express a quaternion as a hypercomplex exponential as shown below,

$$\bar{\mathbf{q}} = e^{\frac{\theta}{2}(u_x i + u_y j + u_z k)} = \cos\left(\frac{\theta}{2}\right) + (u_x i + u_y j + u_z k)\sin\left(\frac{\theta}{2}\right), \tag{A.8}$$

$$= \cos\left(\frac{\theta}{2}\right) + \hat{\mathbf{u}}\sin\left(\frac{\theta}{2}\right), \tag{A.9}$$

Therefore by setting the Euler angle as θ and the Euler axis as the unit vector $\hat{\mathbf{u}}$ you can then represent the entire 3D rotation group.

Representing subsequent rotations is also simple in the quaternion framework, if some rotation $\bar{\mathbf{q}}_1$ is performed, followed by some rotation $\bar{\mathbf{q}}_2$ then the effective quaternion rotation is just $\bar{\mathbf{q}}_3 = \bar{\mathbf{q}}_2\bar{\mathbf{q}}_1$, where the two rotations are multiplied using quaternion multiplication. Then to take some 3D position vector ($\mathbf{p}^B$) and rotate it with some quaternion ($\bar{\mathbf{q}}_B^E$) all that needs to be done is the following quaternion multiplication,

$$\mathbf{p}^E = \bar{\mathbf{q}}_B^E \otimes \mathbf{p} \otimes \bar{\mathbf{q}}_B^{E-1}, \tag{A.10}$$

where $\mathbf{p}^E$ is the rotated vector, $\bar{\mathbf{q}}_B^E$ is the rotation quaternion and $\bar{\mathbf{q}}_B^{E-1}$ is the conjugate quaternion of $\bar{\mathbf{q}}_B^E$. The only issue that remains is multiplying a vector of length 3 and a quaternion of length 4. This is accomplished by setting the three components of the position vector as the imaginary components of a quaternion and setting the component of the quaternion to zero. With this done simply perform standard quaternion multiplication on the new quaternion.

As a four dimensional vector the quaternion representation of a rotation is computationally efficient. Also if renormalization is required due to numerical errors this is more easily performed with a quaternion than a rotation matrix.

The manipulation of quaternions is not intuitive and so some text advocate avoiding quaternions in order to make the navigation equations easier to follow, therefore easier to spot mistakes [7]. We decided to instead define a quaternion class in Python. This class included built-in methods for rotating a three-dimensional vector, as well as quaternion multiplication, conjugation, normalization, etc. This quaternion class then underwent thorough unit testing using hundreds of random unit quaternions.

A.3.3 Quaternion Propagation over time

Just like any other rotation formalism the quaternion formalism must allow an update to the transformation given some rotation. In the presence of some angular velocity $\boldsymbol{\omega}$ the quaternion $\bar{\mathbf{q}}$ is given below [49],

$$\dot{\bar{\mathbf{q}}} = 0.5\,\bar{\mathbf{q}} \otimes \bar{\boldsymbol{\omega}}, \tag{A.11}$$

where $\bar{\boldsymbol{\omega}}$ denotes a quaternion with imaginary components $\omega_x, \omega_y, \omega_z$, and a real component of zero.

For a slightly clearer description of Equation A.11 we can write out the quaternion multiplication as a matrix multiplication.

$$\dot{\mathbf{q}} = \begin{bmatrix} \dot{a} \\ \dot{b} \\ \dot{c} \\ \dot{d} \end{bmatrix} = \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix} \begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}. \tag{A.12}$$

$$\begin{aligned} \dot{a} &= -0.5\left(b\omega_x + c\omega_y + d\omega_z\right), \\ \dot{b} &= 0.5\left(a\omega_x - d\omega_y + c\omega_z\right), \\ \dot{c} &= 0.5\left(d\omega_x + a\omega_y - b\omega_z\right), \\ \dot{d} &= 0.5\left(-c\omega_x + b\omega_y + a\omega_z\right). \end{aligned} \tag{A.13}$$

A.3.4 Euler Angles

The Euler angles rotation formalism represents the rotation as three separate rotations, about the three axes of the coordinate system. A positive rotation follows the right-hand corkscrew rule. A rotation around the upward or z axis is called (γ), the forwards or y axis is called (ϕ) and around the rightwards or x axis is called (θ).

We can write these rotations in the familiar form of the rotation matrices given below.

$$\begin{aligned} R_x &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}, \\ R_y &= \begin{bmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{bmatrix}, \\ R_z &= \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}. \end{aligned} \tag{A.14}$$

Since rotations are not commutative we must specify the order or rotations. In our case we adopt the common z-y-x order, (that is, first a rotation about the z axis, then the y axis, then the x axis). This gives us the following composite rotation matrix.

$$R_E^B = R_x R_y R_z. \tag{A.15}$$

$$R_E^B = \begin{bmatrix} \cos\phi\cos\gamma & \cos\phi\sin\gamma & -\sin\phi \\ -\cos\theta\sin\gamma + \sin\theta\sin\phi\cos\gamma & \cos\theta\cos\gamma + \sin\theta\sin\phi\sin\gamma & \sin\theta\cos\phi \\ \sin\theta\sin\gamma + \cos\theta\sin\phi\cos\theta & -\sin\theta\cos\gamma + \cos\theta\sin\phi\sin\gamma & \cos\theta\cos\phi \end{bmatrix}. \tag{A.16}$$

In Equation A.16 the matrix rotates some vector from the E (East North Up) frame to the B (Body) frame. The inverse rotation of a rotation matrix is simply the transpose of the original rotation matrix, i.e.

$$R_B^E = (R_x R_y R_z)^T = R_z^T R_y^T R_x^T, \tag{A.17}$$

$$R_B^E = \begin{bmatrix} \cos\phi\cos\gamma & -\cos\theta\sin\gamma + \sin\theta\sin\phi\cos\gamma & \sin\theta\sin\gamma + \cos\theta\sin\phi\cos\theta \\ \cos\phi\sin\gamma & \cos\theta\cos\gamma + \sin\theta\sin\phi\sin\gamma & -\sin\theta\cos\gamma + \cos\theta\sin\phi\sin\gamma \\ -\sin\phi & \sin\theta\cos\phi & \cos\theta\cos\phi \end{bmatrix}. \tag{A.18}$$

For our purposes the system of Euler angle's is an intuitive and convenient description of the rotation of the body frame with respect to the the ENU frame. These angles may be familiar to the reader as pitch (θ), roll (ϕ), and yaw (γ). This does allow a simple separation of the uncertainty in each rotation. So, in our situation where we have a highly constrained system, the pitch and roll angles of our system have much less freedom than the yaw angle.

Because our system is intended for terrestrial, road-going vehicles only we need not concern ourselves with the problem of gimbal lock which can occur at high pitch or roll angles. Simply put, if a road-going vehicle is approaching a pitch or roll angle of 90^o navigation may not be the primary point of concern.

A.3.5 Euler Angle propagation over time

An important part of any transformation formalism is the process of updating the transformation given some frame rotation $\boldsymbol{\omega}$. We can calculate the rate of change of each of the Euler angles by using the appropriate rotation matrices and the angular velocity. The rates of change of the Euler angles given below will be denoted by dot notation.

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ 0 \\ 0 \end{bmatrix} + R_x \begin{bmatrix} 0 \\ \dot{\phi} \\ 0 \end{bmatrix} + R_x R_y \begin{bmatrix} 0 \\ 0 \\ \dot{\gamma} \end{bmatrix}. \tag{A.19}$$

The equation given above can then be rearranged to give the following [49].

$$\begin{aligned} \dot{\theta} &= (\omega_y \sin\theta + \omega_z \cos\theta)\tan\phi + \omega_x, \\ \dot{\phi} &= \omega_y \cos\theta - \omega_z \sin\theta, \\ \dot{\gamma} &= (\omega_y \sin\theta + \omega_z \cos\theta)\sec\phi. \end{aligned} \tag{A.20}$$

A.4 State Jacobian

T The following page shows the state Jacobian used in the filtering performed in Chapter 7. Due to the size of the Jacobian the

$$dt\begin{bmatrix}
\frac{1}{dt} & 0 & 0 & (-\cos\theta\sin\gamma+\sin\theta\sin\phi\cos\gamma) & 0 & (\sin\theta\sin\gamma+\cos\theta\sin\phi\cos\gamma)v_F & \sin\theta\cos\phi\cos\gamma v_F & (-\cos\theta\cos\gamma-\sin\theta\sin\phi\sin\gamma)v_F & 0 & 0 & \\
0 & \frac{1}{dt} & 0 & (\cos\theta\cos\gamma+\sin\theta\sin\phi\sin\gamma) & 0 & (-\sin\theta\cos\gamma+\cos\theta\sin\phi\sin\gamma)v_F & \sin\theta\cos\phi\sin\gamma v_F & (-\cos\theta\sin\gamma+\sin\theta\sin\phi\cos\gamma)v_F & 0 & 0 & 0 \\
0 & 0 & \frac{1}{dt} & \sin\theta\cos\phi & 0 & \cos\theta\cos\phi v_F & -\sin\theta\sin\phi v_F & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{1}{dt} & 1 & -g\cos\theta\cos\phi & g\sin\theta\sin\phi & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{1}{dt} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \frac{1}{dt}+(\omega_y\cos\theta-\omega_z\sin\theta)\tan\phi+\omega_x & (\omega_y\sin\theta+\omega_z\cos\theta)\left(\frac{1.0}{\cos\phi)^2}\right) & 0 & 1 & \sin\theta\tan\phi & \cos\theta\tan\phi \\
0 & 0 & 0 & 0 & 0 & (-\omega_y\sin\theta-\omega_z\cos\theta) & \frac{1}{dt} & 0 & 0 & \cos\theta & -\sin\theta \\
0 & 0 & 0 & 0 & 0 & (\omega_y\cos\theta-\omega_z\sin\theta)\left(\frac{1.0}{\cos\phi)}\right) & (\omega_y\sin\theta+\omega_z\cos\theta)\tan\phi\left(\frac{1.0}{\cos\phi)}\right) & \frac{1}{dt} & \sin\theta(1.0/\cos\phi) & \cos\theta(1.0/\cos\phi) & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{dt} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{dt} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{dt}
\end{bmatrix}. \tag{A.21}$$

A.5 Individual Tag Results

Below are the AIC values for the individual tags used for the experiment described in Chapter 3.

A.5.1 Dunedin (45.52°S) data set

Tag_3542326081 Data length: 1238

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	11092.71	10886.45	10779.6	10941.58	10855.54	10845.67
LatΔAIC	313.11	106.85	0.0	161.98	75.94	66.07

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	11060.67	10912.88	10826.56	10954.82	10900.99	10888.11
LonΔAIC	234.11	86.32	0.0	128.26	74.43	61.55

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	12341.17	12137.9	12057.95	12191.36	12147.34	12135.55
AltΔAIC	283.22	79.95	0.0	133.41	89.39	77.6

Tag_3544164930 Data length: 1375

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	11286.75	11120.02	11044.63	11163.83	11108.23	11092.57
LatΔAIC	242.12	75.39	0.0	119.2	63.6	47.94

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	10946.93	10741.41	10621.74	10809.81	10695.45	10693.64
LonΔAIC	325.19	119.67	0.0	188.07	73.71	71.9

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	12098.08	11858.99	11744.27	11941.69	11834.57	11818.95
AltΔAIC	353.81	114.72	0.0	197.42	90.3	74.68

Tag_3609997124 Data length: 1848

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(2,2)
Lat AIC	13623.2	13597.52	13542.65	13608.04	13599.53	13595.9
LatΔAIC	80.55	54.87	0.0	65.39	56.88	53.25

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(2,2)
Lon AIC	13716.65	13703.76	13641.29	13711.86	13706.09	13696.22
LonΔAIC	75.36	62.47	0.0	70.57	64.8	54.93

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(2,2)
Alt AIC	16255.52	16237.63	16117.98	16254.02	16247.41	16189.35
AltΔAIC	137.54	119.65	0.0	136.04	129.43	71.37

A.5.2 North Stradbroke Island (27.25^{o}S) data set

Tag_3609746981 Data length: 573

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	4679.77	4636.48	4545.11	4656.14	4619.19	4613.8
Lat ΔAIC	134.66	91.37	0.0	111.03	74.08	68.69

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	4544.51	4527.7	4475.35	4537.02	4533.26	4532.33
LonΔAIC	69.16	52.35	0.0	61.67	57.91	56.98

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	5176.92	5158.1	5100.17	5167.88	5166.2	5161.5
AltΔAIC	76.75	57.93	0.0	67.71	66.03	61.33

Tag_3610008640 Data length: 575

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	3965.18	3888.97	3828.13	3924.1	3897.52	3898.63
LatΔAIC	137.05	60.84	0.0	95.97	69.39	70.5

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	4030.65	4003.54	3940.48	4015.31	3999.1	3994.98
LonΔAIC	90.17	63.06	0.0	74.83	58.62	54.5

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	4757.7	4740.61	4684.76	4749.6	4746.36	4741.75
AltΔAIC	72.94	55.85	0.0	64.84	61.6	56.99

Tag_3610192452 Data length: 575

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	5292.46	5481.42	5003.52	5249.71	5107.08	5084.17
LatΔAIC	288.94	477.9	0.0	246.19	103.56	80.65

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	4500.34	4467.86	4379.18	4484.73	4440.82	4437.9
LonΔAIC	121.16	88.68	0.0	105.55	61.64	58.72

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	5401.76	5316.29	5193.25	5350.29	5314.35	5265.21
AltΔAIC	208.51	123.04	0.0	157.04	121.1	71.96

Tag_3610455108 Data length: 578

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	4148.89	4077.15	3984.51	4106.92	4068.04	4035.61
LatΔAIC	164.38	92.64	0.0	122.41	83.53	51.1

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	4183.5	4109.81	4023.61	4138.92	4090.97	4079.44
LonΔAIC	159.89	86.2	0.0	115.31	67.36	55.83

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	5006.14	4965.91	4866.34	4982.18	4956.12	4934.83
AltΔAIC	139.8	99.57	0.0	115.84	89.78	68.49

A.5.3 Port Douglas (16.30°S) data set

Tag_3609746981 Data length: 811

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	6050.49	6032.31	5970.13	6041.24	6033.58	6028.26
LatΔAIC	80.36	62.18	0.0	71.11	63.45	58.13

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(1,1)
Lon AIC	6150.6	6143.55	6079.36	6150.45	6143.35	DNC
LonΔAIC	71.24	64.19	0.0	71.09	63.9920175622	DNC

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	6970.38	6961.18	6895.89	6968.89	6970.82	6968.16
AltΔAIC	74.49	65.29	0.0	73.0	74.93	72.27

Tag_3610008640 Data length: 762

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	6612.04	6230.78	6112.03	6382.53	6243.2	6187.25
LatΔAIC	500.01	118.75	0.0	270.5	131.17	75.22

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	6879.98	6618.51	6518.04	6668.21	6637.21	6599.62
LonΔAIC	361.94	100.47	0.0	150.17	119.17	81.58

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	7018.38	6877.45	6792.35	6926.33	6879.25	6850.92
AltΔAIC	226.03	85.1	0.0	133.98	86.9	58.57

Tag_3610192452 Data length: 804

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	5919.09	5799.67	5734.28	5833.93	5798.09	5791.57
LatΔAIC	184.81	65.39	0.0	99.65	63.81	57.29

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	6189.34	6149.31	6059.82	6164.29	6141.26	6111.09
LonΔAIC	129.52	89.49	0.0	104.47	81.44	51.27

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	6969.95	6945.8	6862.09	6955.77	6956.42	6953.33
AltΔAIC	107.86	83.71	0.0	93.68	94.33	91.24

Tag_3610455108 Data length: 808

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	6682.24	6465.26	6222.8	6515.58	6364.8	6298.61
LatΔAIC	459.44	242.46	0.0	292.78	142.0	75.81

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	5973.4	5951.56	5876.67	5961.67	5936.49	5927.27
LonΔAIC	96.73	74.89	0.0	85.0	59.82	50.6

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	7037.01	7014.92	6906.62	7027.2	7007.93	6982.1
AltΔAIC	130.39	108.3	0.0	120.58	101.31	75.48

A.5.4 Thursday Island (10.35^{o}S) data set

Tag_3610008640 Data length: 574

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	3660.77	3485.05	3429.82	3540.87	3496.33	3477.35
LatΔAIC	230.95	55.23	0.0	111.05	66.51	47.53

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	3498.08	3350.28	3282.48	3410.46	3338.71	3319.6
LonΔAIC	215.6	67.8	0.0	127.98	56.23	37.12

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	4953.37	4755.87	4660.54	4834.73	4768.22	4728.39
AltΔAIC	292.83	95.33	0.0	174.19	107.68	67.85

Tag_3610192452 Data length: 576

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(2,2)
Lat AIC	3538.82	3419.07	3339.12	3457.87	3403.31	3394.29
LatΔAIC	199.7	79.95	0.0	118.75	64.19	55.17

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	3741.71	3649.24	3572.91	3684.28	3632.87	3616.19
LonΔAIC	168.8	76.33	0.0	111.37	59.96	43.28

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	4649.21	4518.55	4419.77	4571.95	4510.57	4474.73
AltΔAIC	229.44	98.78	0.0	152.18	90.8	54.96

Tag_3610455108 Data length: 570

Lat AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lat AIC	3976.65	3759.4	3685.04	3836.37	3771.12	3734.26
LatΔAIC	291.61	74.36	0.0	151.33	86.08	49.22

Lon AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Lon AIC	4101.71	3822.61	3729.16	3942.0	3840.83	3789.45
LonΔAIC	372.55	93.45	0.0	212.84	111.67	60.29

Alt AIC	Gauss	O-U	AR(9)	MA(1)	MA(3)	ARMA(3,3)
Alt AIC	4744.88	4616.79	4487.26	4666.16	4614.26	4556.42
AltΔAIC	257.62	129.53	0.0	178.9	127.0	69.16