0.1 Data Processing

NEALE & HAWKES

This section covers the process of converting the raw sensor data into directional wave spectra. There are two parts to this process: firstly, the sensor data must be converted to a dataset of heave, pitch and roll motions. Secondly, this data is used to create a directional wave spectra by performing cross-spectral Fourier analysis.

Throughout this process, there is a vast amount of data to be handled involving huge 3D datasets, and it quickly became apparent that a computer program should be developed to perform the analysis. Also, in the first half of the data processing, which involved complex 3D filters, it was useful to assess the performance visually, using rotating 3D graphics.

The development of the software will be tackled at the end of this section.

0.1.1 IMU Processing

HAWKES

The IMU gives values for acceleration and magnetic field strength along the x-, y- and z-axis, as well as the angular velocity about these axis.

The data was sampled at a rate of 50 Hz (20 ms period) to ensure everything ran smoothly on the buoy and on the shore computer. The magnetometer and gyroscope could be considered almost noise-free, since the real-time plotting of the sensor values produced smooth curves. However, the accelerometer was very noisy, so a moving-average low-pass filter was added to the sensor data to remove this noise. A 5-point moving average was used, which produced the best results by visual observation.

This reduced the effective sample rate to 10 Hz. As such, the fastest motions that could be measured would be at a frequency of 5 Hz (by the Nyquist principle). This was acceptable for the wave buoy application, but the sample rate could be increased depending on the power of the shore computer.

Before these sensors can be used, a calibration process also has to be applied. The calibration for the gyroscope and accelerometer is simple. The sensor was laid flat and still and the zero-offset measured, and then subtracted from all future readings.

For the magnetometer, the process is less simple. For an ideal magnetometer which is rotated in three-dimensions (such that it covers as many 3D headings as possible) the resultant x-, y and z-values should scribe a perfect sphere centred on the origin [Renaudin et al., 2010].

However, there are two forms of deformation that often occur. The first is hard iron deformation, caused by magnetic flux distortions which retain a constant position (distance and orientation) relative to the magnetometer. In this case, hard iron deformation could arise due to electrical noise or ferrous materials on the buoy - such as steel ballast weights. Hard iron calibration should be performed for each buoy or mounting system.

The other form of deformation is soft iron deformation. This causes the ideal sphere to be squashed into a spheroid. Soft iron deformation arises due to external magnetic flux, whose orientation relative to the sensor is not constant - although the strength of (distance from) the distortion is assumed constant. Soft iron deformation is not usually corrected for, since most systems containing a magnetometer are mobile and sources of soft iron deformation move around too much. However, for devices such as the wave buoy, which rotates in space but mostly does not translate - soft iron calibration should be performed in every location they are used.

In most cases, the soft iron deformation should be small since the wave buoy will be floating far from any interference. However, it was also intended to test the wave buoy in the University of Southampton Lamont Towing Tank which was surrounded by large amounts of iron. The towing tank is notorious for

rendering magnetometers useless; however, since the buoy would be stationary inside the tank (albeit rotating), a hard iron calibration should be enough to maintain use of the magnetometer.

In order to find the equation of the ellipsoid, and hence correct for hard and soft iron deformation, 12 variables must be found, such that [Renaudin et al., 2010]:

$$\begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & a_9 \end{bmatrix} \begin{bmatrix} m_{x,raw} - b_x \\ m_{y,raw} - b_y \\ m_{z,raw} - b_z \end{bmatrix} \tag{0.1.1}$$

where the hard-iron calibration deformation was represented by b_x, b_y, b_z and the soft-iron calibration by the constants a_{1-9}. Note that the ellipsoid would not necessarily be aligned to the principal axis. Also note, that the radius of the ideal sphere represents the Earth's magnetic field strength, but since all magnetometer values were normalized there was no need to quantify this.

No library could be found that performed ellipsoid fitting in the chosen programming language (C++) so it was decided to transpose a similar code from MATLAB, using an open-source library of matrix transformation functions. The code was based on an eigenvalue method developed by the Qingde and Griffiths [2004].

Another option would have been to use a least-squares fitting technique as employed by other codes [Renaudin et al., 2010]. Unofficial sources have reported that the code's create almost identical data, so the method by Qingde and Griffiths [2004] was used, since it was easier to transpose.

Figure 0.1.1 shows an example of quite extreme hard- and soft-iron deformation. For reference, the calibration constants for the buoy in the towing tank are presented here:

$$\begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = \begin{bmatrix} 4.72296 & 0.204366 & -0.205252 \\ 0.204366 & 5.00365 & 0.114621 \\ -0.205252 & 0.114621 & 5.47836 \end{bmatrix} \begin{bmatrix} m_{x,raw} + 0.0213084 \\ m_{y,raw} - 0.0547494 \\ m_{z,raw} - 0.027601 \end{bmatrix} \tag{0.1.2}$$

The scale is significantly different to the images in Figure 0.1.1, which were created from raw sensor values; but the principal is the same. There are significant components of all three raw sensor values ($m_{x,raw}$, $m_{y,raw}$, $m_{z,raw}$) appearing in the final values indicating that the calibration was very valuable, although the hard-iron (offset) calibration was negligible - probably because the buoy carried little in the way of ferrous material.

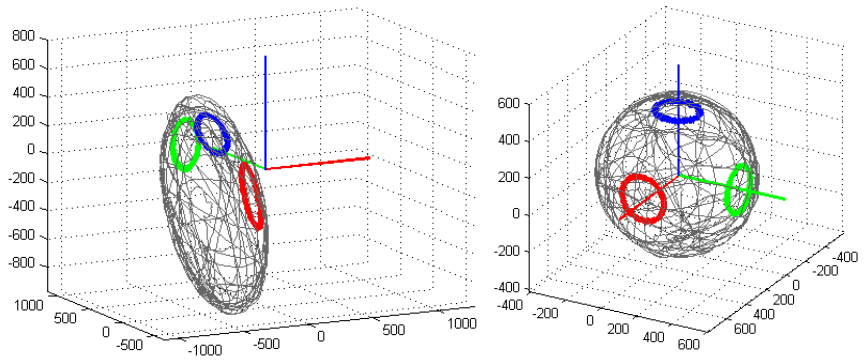

Figure 0.1.1: An example of quite extreme hard- and soft-iron deformation and the successful calibration by fitting the spheroid to a sphere centred on the origin. The scale is in terms of raw, digital sensor values. Images courtesy of Sebastian O.H. Madgwick.

After calibration, the processing of the sensor motions could begin. This closely followed a detailed report by Madgwick et al. [2011]; and the open-source sub-routines which perform the IMU calculations were also taken from Madgwick et al..

The normalized magnetometer values ${}^{S}\hat{m}_{t=0}$ (where t is the timestep and ^{S} refers to the sensor frame of reference) could be used to align the the buoy to a starting orientation, ${}^{S}_{E}\mathbf{q}_{t=0}$. Where ${}^{S}_{E}$ implies that the quaternion $\mathbf{q}$ refers to the rotation of the sensor frame relative to the earth frame of reference. The entire process is performed in quaternion notation - quaternions are 4-dimensional complex numbers which describe the 3-dimensional orientation of an object.

There are two methods for updating the orientation as the timestep advances. The first is through integrating the angular velocity, where the angular velocity measurements ω_x, ω_y and ω_z are expressed in rad/s and form a vector ${}^{S}\bar{\omega} = [0, \omega_x, \omega_y, \omega_z]$. A quaternion derivative equal to the rate of change of the orientation quaternion can be calculated as (using the tensor product, $\otimes$):

$$ {}^{S}_{E}\dot{\mathbf{q}}_{\omega,t} = \frac{1}{2}{}^{S}_{E}\hat{\mathbf{q}} \otimes {}^{S}\bar{\omega}_{t} \tag{0.1.3} $$

which can be numerically integrated to find the new orientation, from the previous timestep, ${}^{S}_{E}\hat{\mathbf{q}}_{est,t-1}$.

$$ {}^{S}_{E}\mathbf{q}_{\omega,t} = {}^{S}_{E}\hat{\mathbf{q}}_{est,t-1} + {}^{S}_{E}\dot{\mathbf{q}}_{\omega,t}\Delta t \tag{0.1.4} $$

where the subscript ω indicates that this value was found from angular rates.

The second method, called the vector method, uses a gradient descent optimization to find the compound orientation from the magnetometer and accelerometer. The magnetometer can only provide orientation relative to the magnetic field lines at the sensor location, and cannot not discern rotation about the field line - which in the UK, has an inclination of about $\approx 70°$. Thus, the accelerometer must be used to provide another direction vector (average gravitational force) which combines to provide the complete orientation. In this method, the orientation does not drift due to cumulative integration error, which the previous method suffered from.

The gradient descent optimization uses the following algorithm, which increments k until convergence is reached; μ is the optimization step size which would normally be calculated on a per-optimization-step basis.

$$ {}^{S}_{E}\mathbf{q}_{k+1} = {}^{S}_{E}\hat{\mathbf{q}}_{k} - \mu\frac{\nabla f}{||\nabla f||} \tag{0.1.5} $$

In this application it was possible to let each iteration become one timestep so long as $\mu = \mu_t$ was equal or greater to the physical rate of change of orientation. Subscript ∇ indicates the orientation was calculated using the vector method.

$$ {}^{S}_{E}\mathbf{q}_{\nabla,t} = {}^{S}_{E}\hat{\mathbf{q}}_{est,t-1} - \mu_t\frac{\nabla f}{||\nabla f||} \tag{0.1.6} $$

In both cases ∇f is the gradient function, defined as

$$ \nabla f = \mathbf{J}^{T}_{g,b}({}^{S}_{E}\hat{\mathbf{q}}_{est,t-1}, {}^{E}\hat{\mathbf{b}})\boldsymbol{f}_{g,b}({}^{S}_{E}\hat{\mathbf{q}}_{est,t-1}, {}^{S}\hat{\mathbf{a}}, {}^{E}\hat{\mathbf{b}}, {}^{S}\hat{\mathbf{m}}). \tag{0.1.7} $$

The objective function $\boldsymbol{f}_{g,b}$ is a function of the previous time-step orientation quaternion ${}^{S}_{E}\hat{\mathbf{q}}_{est,t-1}$ $= [q1, q2, q3, q4]$, the normalized accelerometer data ${}^{S}\hat{\mathbf{a}} = [0, a_x, a_y, a_z]$, the Earth's magnetic field vector

at the sensor location ${}^{E}\hat{\mathbf{b}} = [0, b_x, 0, b_z]$ and the magnetometer data ${}^{S}\hat{\mathbf{m}} = [0, m_x, m_y, m_z]$. Thus

$$\boldsymbol{f}_{g,b}({}^{S}_{E}\mathbf{q}_{\hat{est},t-1}, {}^{S}\hat{\mathbf{a}}, {}^{E}\hat{\mathbf{b}}, {}^{S}\hat{\mathbf{m}}) = \begin{bmatrix} 2b_x(0.5 - q_3^2 - q_4^2) + 2b_z(q_2q_4 - q_1q_3) - m_x \\ 2b_x(q_2q_3 - q_1q_4) + 2b_z(q_1q_2 + q_3q_4) - m_y \\ 2b_x(q_1q_3 + q_2q_4) + 2b_z(0.5 - q_2^2 - q_3^2) - m_z \end{bmatrix}. \tag{0.1.8}$$

The transposed Jacobian function $\mathbf{J}^{T}_{g,b}$ is a function of the previous timestep orientation and the Earth's magnetic field vector only.

$$\mathbf{J}^{T}_{g,b}({}^{S}_{E}\hat{\mathbf{q}}_{est,t-1}, {}^{E}\hat{\mathbf{b}}) = \begin{bmatrix} -2b_zq_3 & 2b_zq_4 & -4b_xq_3 - 2b_zq_1 & -4b_xq_4 + 2b_zq_2 \\ -2b_xq_4 + 2b_zq_2 & 2b_xq_3 + 2b_zq_1 & 2b_xq_2 + 2b_zq_4 & -2b_xq_1 + 2b_zq_3 \\ 2b - xq_3 & 2b_xq_4 - 4b_zq_2 & 2b_xq_1 - 4b_zq_3 & 2b_xq_2 \end{bmatrix} \tag{0.1.9}$$

The gradient function has been broken down into programmable sections, which allowed the orientation from the vector method to be computed. The orientation from the angular velocity could also be calculated, giving two different methods for calculation.

The first method gives instant and accurate step-changes between iterations, but is subject to cumulative error as the orientation is calculated using 'dead-reckoning' techniques. The second method produces an orientation that could be maintained, since it uses the magnetometer's absolute orientation - but this method is slow to respond, since the time-stepping is also the optimization routine.

In order to make the most of both methods a filter fusion algorithm can be applied, which attempts to combine the signals to create an accurate and reliable orientation.

$${}^{S}_{E}\mathbf{q}_{est,t} = \frac{\beta\Delta t}{\mu_t}\left(-\mu_t\frac{\nabla f}{||\nabla f||}\right) + {}^{S}_{E}\hat{\mathbf{q}}_{est,t-1} + {}^{S}_{E}\dot{\mathbf{q}}_{\omega,t}\Delta t \tag{0.1.10}$$

The value of β was chosen as 0.05 by observation of the 3D graphics. A higher value of β represents greater reliance on the magnetometer to correct the orientation from the 'dead-reckoning' gyroscope calculation. If β was too low, the sensor would appear to over-shoot during rotation, and if it was too high the 3D animation would rotate too slowly due to the slow descent algorithm.

The initial value for ${}^{S}_{E}\mathbf{q}_0$ did not matter, since the buoy would settle to the absolute orientation within a few seconds. The initial value was, therefore, taken as the null quaternion ${}^{S}_{E}\mathbf{q}_0 = [1, 0, 0, 0]$ which represents zero rotation, rather than implementing a separate algorithm to use just the magnetometer and accelerometer data at initialization.

Using the quaternion for each timestep, the absolute values of pitch and roll of the sensor in the Earth's reference system can be found, by converting to Euler angles of pitch and roll. This effectively gives the North-South and East-West tilt of the buoy - *regardless of the buoys rotation.*

This is required for the calculation of the directional wave spectra in second half of the wave buoy data processing.

The other value required for the second half of the processing is the heave of the buoy - which can be found by performing double integration on the accelerometer data. The accelerometer values were resolved to vertical using quaternion vector rotation from ${}^{S}_{E}\mathbf{q}_{est,t}$, since components of a_x and a_y would contribute to the vertical acceleration, as well as a_z, when the buoy was pitching and rolling respectively.

Visually, when β was equal to zero, the animation of the buoy would rotate indefinitely due to the gyroscope drift - which was caused by the cumulative error of integration. For orientation purposes, this could be compensated by compounding the orientation with the absolute values from the magnetometer - however, the accelerometer suffers from even worse cumulative error and a creative method had to be

employed to remove it, since the GPS was not fast or accurate enough to compensate for it.

Accelerometers are notorious for having huge cumulative error, partly because they have more physical error than other sensors, such as gyroscopes. Also, the component of gravity (g) limits the resolution of the results, since small oscillations in a_z are carried on a large $1g$ signal. Finally, the accelerations must be integrated twice creating numerical error, and far more importantly, integrating all the physical errors. Even the most expensive accelerometers (thousands of pounds) only guarantee error-free position tracking for 100ms or so; whereas the wave buoy data samples could be up to several minutes.

Firstly, the extent of the cumulative error was found. Figure 0.1.2 shows a typical wave dataset lasting 24 seconds. In this 24 seconds the cumulative error reached 2.7 kilometres, though the signal still carried the wave oscillations (around 8cm height) on this increasing value.

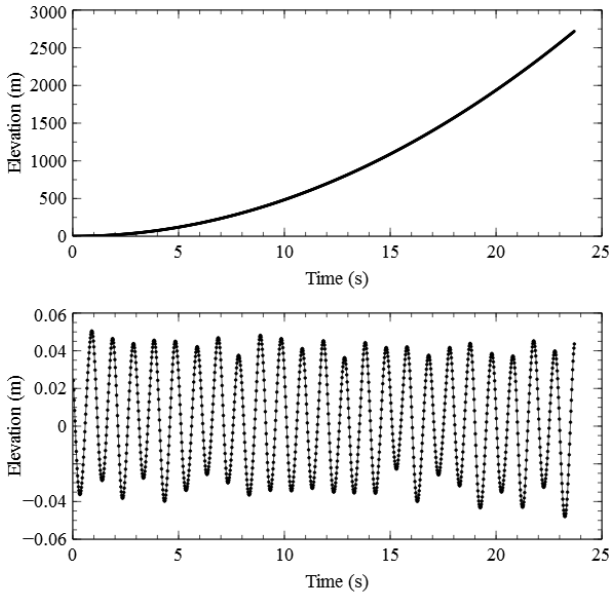

Figure 0.1.2: The top graph shows the typical cumulative error acting on the accelerometer, the small wave oscillations are still present in this signal, but not visible due to the large-scale drift. The lower graph shows this signal after the high-pass filter was implemented.

It was recognized that this cumulative error represented a low-frequency drift; though its frequency varied over time, and over different data sets. This low frequency could be removed by performing a high-pass filter on the data. This was done by taking a moving average and deducting it from the signal. The time-scale of the moving average dictated the time-constant of the high-pass filter, which was typically a 25-point (hence half-second or 2 Hz) high-pass filter. Figure 0.1.2 also shows the same signal after this process was performed.

This process relies on the fact that the average sea height should be constant, and the average of all the data. This provides a reference location which is not available for most accelerometer applications.

Using the methods discussed in this section, the heave, pitch and roll can be used to create the directional and non-directional wave spectra.

0.1.2 Wave Spectra Processing

NEALE

Unlike the controlled environment of a towing tank or wave basin, the wave systems occurring on a sea or lake are always irregular, and therefore it is not possible to obtain wave characteristics without data manipulation. Therefore, a statistical analysis was performed in order to obtain the characteristics of the

wave system. These characteristics included the significant wave height, mean frequency and mean wave direction.

It is possible to define the surface of a sea or lake as an infinite number of sinusoidal waves, each with a different amplitude, frequency, phase and direction. The data obtained from the wave buoys was broken down into these component waves through the use of a Fourier Transform. This also transformed the data from the time domain into the frequency domain [Lloyd, 1998].

The frequency interval was defined as $\frac{1}{T}$ where T is equal to the total length of the data record in seconds. This value was also the minimum frequency that could be measured as its period was equal to the length of the data record.

The maximum frequency able to be analysed was defined by the Nyquist principle. This principle states that the maximum frequency measured should be no greater than half the sampling frequency. This is because at least two points per wave are required to sufficiently define the peaks and troughs.

In order to calculate the significant wave height and mean frequency a Fourier analysis was performed on the vertical (heave) displacements. As a result a set of complex wave elevations and corresponding frequencies were obtained. The magnitude of the complex wave elevations were taken. These values were then converted into a wave amplitude energy density spectrum which allowed the relative importance of each component wave to be identified. This was achieved using the following energy density equation:

$$S_\zeta(\omega_n) = \frac{\zeta_n^2}{2\delta\omega} \tag{0.1.11}$$

where ζ = wave amplitude (m) and ω = frequency (Hz).

It can be shown that the variance of the wave amplitude (m_0) is equal to the area under its energy spectrum [Lloyd, 1998]. Furthermore the variances of the velocity (m_2) and acceleration (m_2), also known as spectral moments, are equal to the areas under the velocity and acceleration spectra respectively. Overall it can be shown that:

$$m_n = \int_0^\infty \omega^n S_\zeta(\omega) d\omega \tag{0.1.12}$$

The mean frequency and period were obtained by calculating the centre of area of the spectrum as follows:

$$\overline{\omega} = \frac{m_1}{m_0} \tag{0.1.13}$$

$$\overline{T} = \frac{m_0}{m_1} \tag{0.1.14}$$

The bandwidth describes to what extent the energy is spread out over different frequencies. An energy spectrum with a narrow bandwidth will have the majority of its energy concentrated over a small band of frequencies. Conversely an energy spectrum with a wide bandwidth will have its energy spread over a wide band of frequencies; this indicates the occurrence of a large number of troughs above the datum level and vice versa. It may be calculated as follows:

$$\in = \sqrt{1 - \frac{m_2^2}{m_0 m_4}} \tag{0.1.15}$$

The significant wave height could then be calculated:

$$\overline{H}_{\frac{1}{3}} = 4.00\sqrt{m_0}\sqrt{1 - \frac{\in^2}{2}} \tag{0.1.16}$$

This value was divided by 2 to obtain the significant wave amplitude.

To verify the result a zero crossing analysis was performed on a small section of data. This analysis defines a wave as the portion of data between two upwards crossings of the datum line [Manly Hydraulics Laboratory, 2000]. The significant wave height is then defined as the average of the greatest 33% of the peaks and troughs. Using the same data set, it was found that the significant wave heights calculated using the spectrum method and the zero crossing method were 0.0085 m and 0.0106 m respectively. An increase of 25% was observed using the zero crossing method, however this was expected due to the data set containing peaks below the datum line and vice versa.

The wave amplitude energy density spectrum could also be described as a non-directional spectrum. To obtain the mean wave direction a directional spectrum was calculated by multiplying the non-directional energy spectrum with a directional spreading function [National Data Buoy Center, 1996]:

$$S(\omega, \theta) = S_\zeta(\omega) D(\omega, \theta) \tag{0.1.17}$$

where D, the directional spreading function is calculated as follows:

$$D(\omega, \theta) = \frac{1}{\pi}(\frac{1}{2} + r_1 \cos(\theta - \theta_1 + r_2 \cos(2(\theta - \theta_2))) \tag{0.1.18}$$

The variables in the directional spreading function equation are:

$$r_1 = \frac{1}{a_0}(a_1^2 + b_1^2)^{\frac{1}{2}} \tag{0.1.19}$$

$$r_2 = \frac{1}{a_0}(a_2^2 + b_2^2)^{\frac{1}{2}} \tag{0.1.20}$$

$$\theta_1 = Mean\ wave\ direction = \tan^{-1}(\frac{b_1}{a_1}) \tag{0.1.21}$$

$$\theta_1 = Principle\ wave\ direction = \tan^{-1}(\frac{b_2}{a_2}) \tag{0.1.22}$$

The following equations will adhere to particular notation:

1 = wave elevation (heave displacement), 2 = east-west wave slope, 3 = north-south wave slope

Variables a_0,a_1,a_2,b_1 and b_2 occurring in Equations (0.1.19) to (0.1.22) are known as the Longuet-Higgins directional parameters [National Data Buoy Center, 1996]. They are calculated as follows:

$$a_0 = \frac{C_{11}}{\pi} \tag{0.1.23}$$

$$a_1 = \frac{Q_{12}}{k\pi} \tag{0.1.24}$$

$$a_2 = \frac{C_{22} - C_{33}}{k^2\pi} \tag{0.1.25}$$

$$b_1 = \frac{Q_{13}}{k\pi} \tag{0.1.26}$$

$$b_2 = \frac{2C_{23}}{k^2\pi} \tag{0.1.27}$$

C_{xy} and Q_{xy} are co-spectral and quadrature spectral density functions respectively. A Fourier analysis is performed on the data for the wave elevations [1], east-west wave slopes [2] and north-south wave slopes [3]. The values of C_{xy} and Q_{xy} can then be calculated as follows:

$$C_{xy} = \frac{Re[X]Re[Y] + Im[X]Im[Y]}{2\delta\omega} \tag{0.1.28}$$

$$Q_{xy} = \frac{Im[X]Re[Y] - Re[X]Im[Y]}{2\delta\omega} \tag{0.1.29}$$

The wave number, k, may be calculated using the following equation, derived from linear wave theory [National Data Buoy Center, 1996].

$$k^2C_{11} = (C_{22} + C_{33}) \tag{0.1.30}$$

Finally the directional energy spectrum may be plotted as shown in Figure **??**. The mean wave direction may be found by taking the frequency of the maximum value of the wave amplitude energy density spectrum and calculating the corresponding value of θ_1.

0.1.3 Software

HAWKES

A computer program was developed to deal with the hardware-software interface, to plot a 3D visualization of the wave buoy and to process all the data - including calibration, IMU and spectral processing. The program was called WABDAP - Wave Buoy Directional Analysis Program.

The program was created in C++ using the Qt libraries for open-source cross-platform GUI distribution, meaning that the program will compile on Microsoft Windows, Apple OSX, all Linux platforms and smartphone platforms such as Android or iOS. Though some modifications should be made before attempting to distribute this program on to a mobile device [Nokia, 2013].

The program makes uses of asynchronous multi-threading. One thread handles the **graphical user interface** (GUI), which contains the 3D graphics, live plotting of the sensor data and other user interface functions. Amongst other things, this thread is responsible for spawning other threads.

The **serial communications** thread uses an open-source library called qserialport, which controls the connection to the serial port on which the XBee module is connected. It is responsible for reading the sensor values emitted by the sensor to all the other threads that require it. For example, the serial thread sends sensor values (with thread-safety) to the GUI thread, so that it can update the live-sensor plots (Figure 0.1.3).

This thread could be improved to include Bluetooth communications, particularly if it were to be used on platforms with built-in Bluetooth - such as mobile devices.

A third thread, called the **IMU** thread, handles the IMU processing using the sensor measurements emitted from the serial thread. This processing is quite intensive, so benefits from running in a separate thread. This thread emits the quaternion to other threads, including the GUI thread which uses it to update the 3D graphics.

The 3D graphics consists of an openGL 2.0 animation which takes the quaternion from the IMU calculations to display the orientation of the buoy. The buoy was represented by an animation of a die, rather than modelling the actual buoy shape in openGL. The die was chosen because its faces could easily

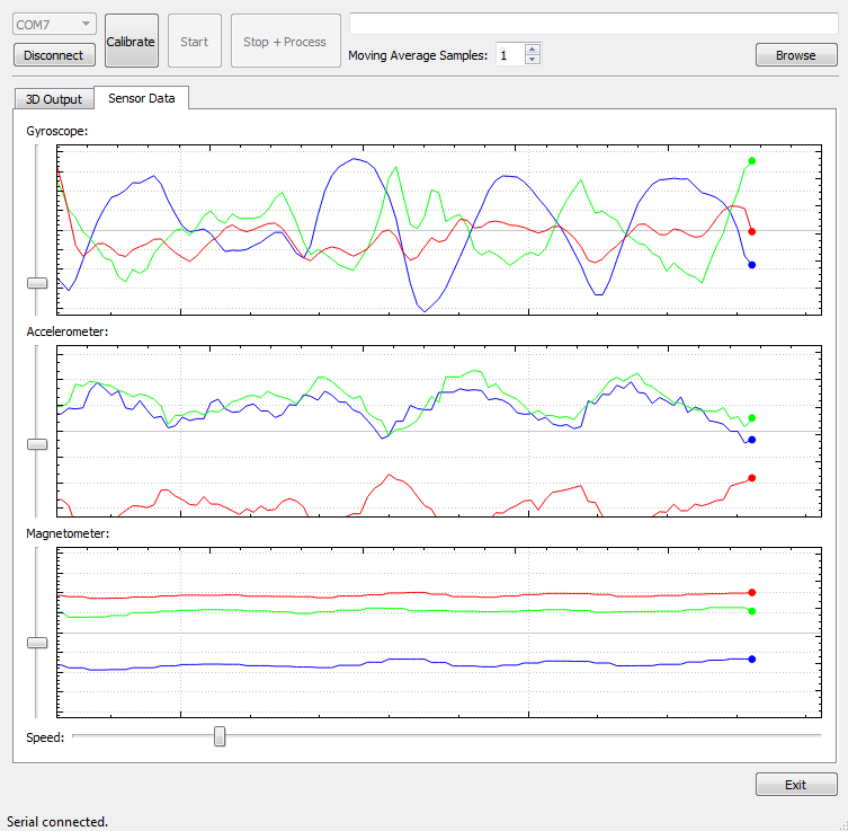

Figure 0.1.3: The serial thread sends sensor data to the main thread for plotting. The scale of the x and y axis can be changed using the sliders, to amplify the signals or change the time-history that is visible. Each dotted vertical line marks one second, and the vertical axes are the non-dimensional raw digital outputs from the sensor. These graphs were used to ensure the sensors were working and to tune the low-pass filter, which could be changed via the 'Moving Average Samples' option on the toolbar. Clearly, the accelerometer was very noisy when the filter was turned off.

be identified - as opposed to using a plain cube which would have been confusing. Figure 0.1.4 shows the 3D animation of the buoy within WABDAP.

Another thread, called the **calibration** thread, is spawned when the calibration process is started. Whilst the calibration process is running, this thread handles all the necessary processing for calibration, including the computationally expensive ellipsoid fitting algorithm. This thread emits the calibration constants to the other threads before destroying itself after the calibration is complete. The GUI for this thread is shown in Figure 0.1.5.

A final thread, called the logger, is created when the logging process begins. During the logging it collects data from the serial and IMU threads, but after logging it processes all the logged data to obtain the directional spectra. This thread takes a long time, particularly because it must write all the results to a text file, including a very large 3D dataset, and benefits from being in its own thread so that the program can continue running.

Note that, due to the object-orientated and threaded nature of the program, it can begin logging new data whilst still processing the previous dataset.

This program runs well on quad-core systems since four threads are usually running (the calibration and logging threads would not normally be run at the same time), though also performed well enough on a dual-core laptop. The laptop was a 1.2Ghz dual core laptop, and struggled to perform some of the

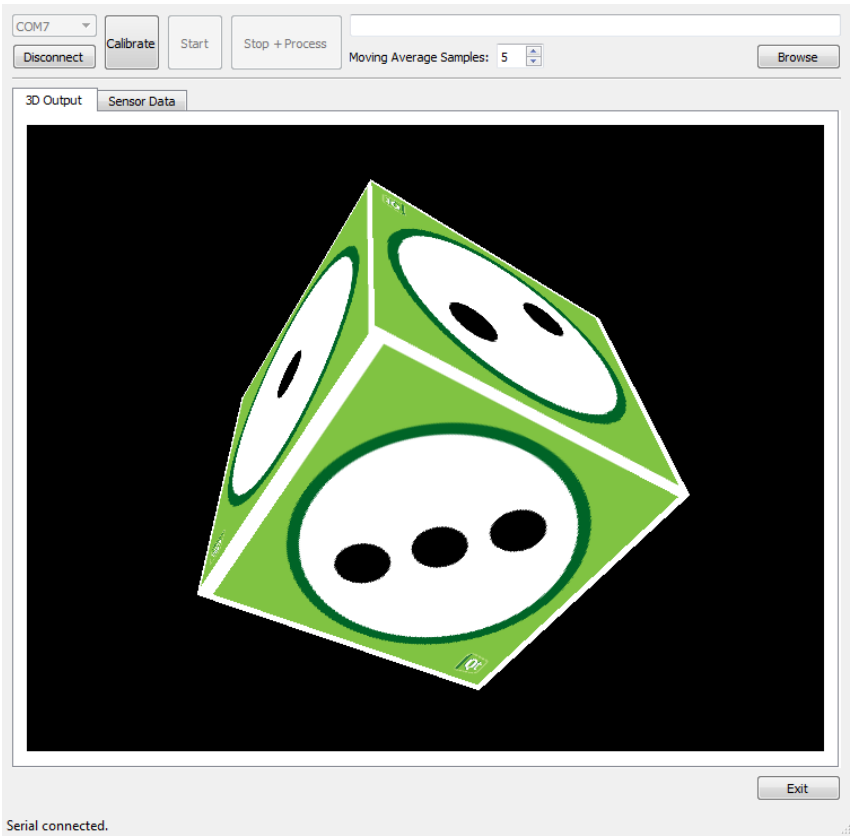

Figure 0.1.4: The 3D animation created in WABDAP, illustrating the real-time buoy orientation. This was necessary, since it was the best means to check the filter fusion algorithm.

logging thread calculations without slowing the user interface, since the processor was saturated, but was still usable.

In the future, it would be best to allow post-processing of the logged data, rather than instantly calculating the spectra - as this would reduce the demand on the shore computer. However, there was no opportunity to build a post-processing script due to time-constraints. This also meant that no attempt was made to try permutations of the data processing - such as changing the high-pass filter frequency to improve the results of the accelerometer.

Overall, the program performed well and the time-consuming process of plotting 3D graphics and developing a GUI was definitely worthwhile, as it allowed a huge amount of testing to be performed quickly and efficiently with real-time results.

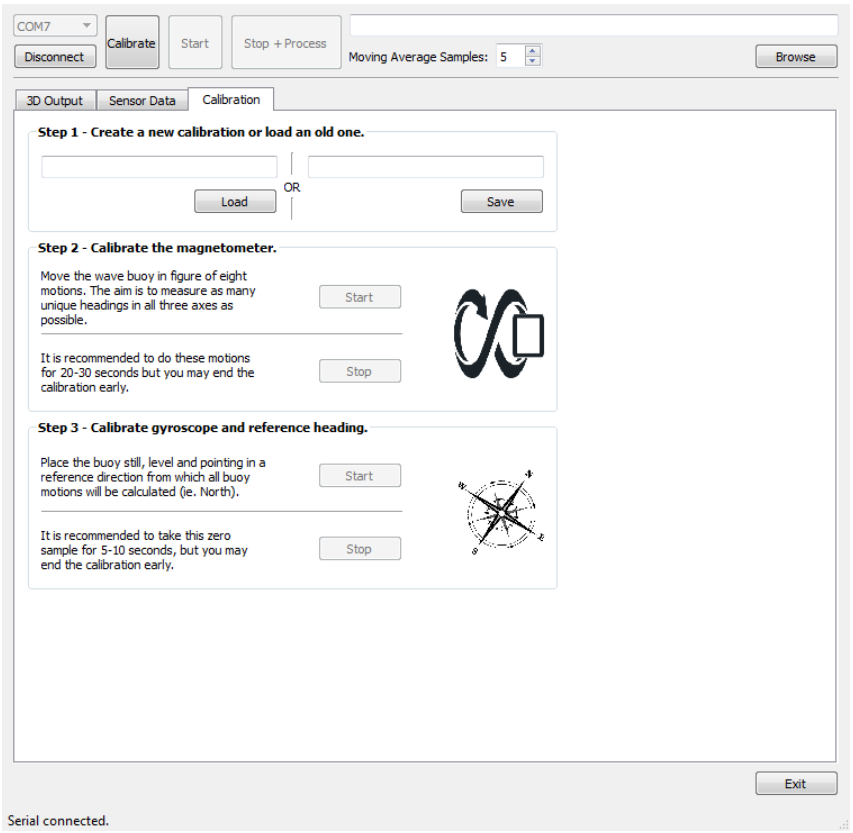

Figure 0.1.5: This tab instructs the user through the calibration process and communicates to the calibration thread via the GUI thread. The user can save calibration results, so that the process does not have to be re-run each time the program is used.

Bibliography

A. R. J. M. Lloyd. *Seakeeping: Ship Behaviour in Rough Weather*. Ellis Horwood Ltd, Chichester, UK, 1998.

S.O.H. Madgwick, A. J L Harrison, and R. Vaidyanathan. Estimation of imu and marg orientation using a gradient descent algorithm. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1–7, 2011. doi: 10.1109/ICORR.2011.5975346.

Manly Hydraulics Laboratory. Analysis of ocean wave data, 2000. URL `http://www.mhl.nsw.gov.au/www/wave_data_stats.htmlx`.

National Data Buoy Center. Nondirectional and directional wave data analysis procedures. Technical report, Stennis Space Center, January 1996. URL `http://www.ndbc.noaa.gov/wavemeas.pdf`.

Nokia. Qt project, 2013. URL `qt-project.org`. Version 5.0.2.

L Qingde and J.G. Griffiths. Least squares ellipsoid specific fitting. *Geometric Modeling and Processing*, 2004.

V. Renaudin, M.H. Afzal, and G. Lachapelle. Complete triaxis magnetometer calibration in the magnetic domain. *Journal of Sensors*, 2010.