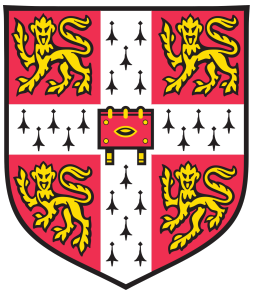
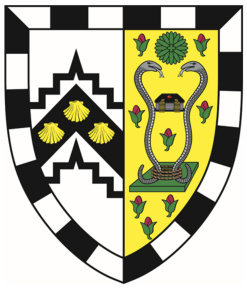
Designing an end-to-end inertial navigation system using earables

Ashwin Ahuja
Gonville & Caius College





A dissertation submitted to the University of Cambridge in partial fulfilment of the requirements for the Computer Science Tripos, Part III

University of Cambridge
Computer Laboratory
William Gates Building
15 JJ Thomson Avenue
Cambridge CB3 0FD
UNITED KINGDOM

Email: aa2001@cam.ac.uk

May 26, 2021

Acknowledgements

I first acknowledge that this work builds significantly on the work of the co-supervisor of this project, Andrea Ferlini. In particular, we use some code from their 2021 earable magnetometer calibration paper [1], including some Python code to complete the optimisation process for their calibration method. My iPhone application also significantly uses the basic framework of their application, which could connect to a single Bluetooth earable and save 3-axis magnetometer data. I would also like to thank Andrea for all his help and guidance throughout the project.

I would also like to thank my supervisor, Cecilia, for her support during the project. Finally, I am also grateful to Benjamin Yass and the Cambridge University 3D Printing Society, on whose 3D printer all the earable prototypes were manufactured.

Declaration

I, Ashwin Ahuja of Gonville & Caius College, being a candidate for the Part III of the Computer Science Tripos, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

Total word count: 11,974

Signed: *Ashwin Ahuja*

Date: 25/05/2021

Word count

Total word count: 11,974

Abstract

Earables (smart headphones) are the cutting edge of wearable technology, allowing users to combine leisure with sensing. They offer a number of advantages for inertial navigation, with reduced noise due to muscoloskeletal damping by being in the upper body and two independent measurements from both ears.

No commercially available earable contains a magnetometer, which is traditionally an important part of heading estimation. Therefore, I designed and manufactured a custom 3D printed earable platform. This includes an Arduino MCU board, 9-axis IMU (with a magnetometer), temperature sensor, pressure sensor, microphone and 550 mAh LiPo battery (with a power management system). The prototype was used to collect a unique in-the-wild dataset from six individuals walking around in a variety of noise environments.

I propose a novel earable IMU calibration framework, building on work by Sipos et al. [2], Shen et al. [3] and Ferlini et al. [1]. I also devise the best heading estimation and displacement estimation methods for earables.

I also develop a framework for position tracking for earable. Using the heading calculated with the magnetometer and pedestrian dead reckoning, I achieve a minimum average tracking drift for one earable of $0.15ms^{-1}$ and heading error of 9°.

Finally, I research how to fuse the measurements of two earables. I demonstrate that using a particle filter to combine headings leads to a 27% reduction in tracking drift. I also propose a novel low-power accurate (<15m error) smartphone tracking method using earables with a 65% power usage reduction compared to continuous GPS tracking.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Inertial Navigation Systems (INS) use inertial measurement units (IMUs) to maintain the location of a device without needing GPS. Earables (in-ear wearables) is an exploding area for wearable research. They allow users to combine leisure with sensing. Earables offer a number of potential advantages over other wearables for tracking, including reduced noise and two independent measurements (one from each ear). In this dissertation, I produce the first inertial navigation system for earables with an average drift of $0.15ms^{-1}$ for one earable and $0.11ms^{-1}$ when fusing both earables.

To do this, I had to overcome a number of significant challenges. The first challenge is the lack of an existing earable sensing platform with a magnetometer. Hence, in Chapter 3, I design and manufacture a new prototype portable earable sensing platform with a powerful Arduino microcontroller with a 9-axis IMU (including a magnetometer), Bluetooth transceiver, audio output, temperature, pressure sensor and microphone.

Additionally, there is a lack of existing earable IMU data, so in Chapter 4, I collect the first of its kind in-the-wild IMU earable dataset.

A third challenge is earable IMU calibration. Thus, after benchmarking various techniques, I define an effective calibration framework, using accelerometer calibration by Sipos et al. [2], gyroscope calibration by Shen et al. [3] and magnetometer calibration by Ferlini et al. [1]. As part of the latter calibration, I devise a novel calibration point filtering and discarding strategy.

I also develop a framework for position tracking for earables. I define the best heading estimation approach, combining a gyroscopic approach and magnetometer approach with a complementary filter. My approach outperforms existing fusion algorithms. I find the best displacement estimation method to be pedestrian dead reckoning, which outperformed a kinematics approach by 93%.

Finally, I devise a novel multi-device sensor fusion approach to combine both earables. The proposed approach reduces power usage for accurate tracking by 65% by incorporating occasional GPS updates, whilst reducing tracking drift by 27% when maximising performance.

1.1 Earables

Earables involve adding new sensors to headphones. These sensors range from motion sensors (the focus of this thesis) to medical sensors and microphones. In this thesis, I design an around-the-ear 3D printed custom earable platform that could be used for a wide array of earable research. The choice to design my own earable arises from no commercial earables containing a magnetometer.

Earables offer several potential benefits for inertial navigation:

- The location of earables on a user's head means that motion sensors encounter less noise [4], due to muscoloskeletal damping. As a result of this, motion sensors in earables present less noise (better defined peaks) compared to those in phones and smartwatches (Figure 1.1), offering advantages for tracking. In this thesis, I find that an INS performed 37% better on earables than on a smartphone.
- Earables can offer auditory feedback. This is a particularly effective method of informing users of outputs of a system.

- Tracking the head's movements with earables opens the door to new opportunities compared to tracking the movement of a user's leg or hand. For example, the heading of an earable can be taken as a proxy for visual attention [5].

  In this thesis, in addition to real-time tracking, I devise a new solution to guide a user to the correct exit at a complicated intersection, outputting differing tones to tell them when they are looking in the correct direction.

- Earables offer two independent measurements (one from each earable). The data from these can be combined. In Chapter 7, I propose methods to combine data from earables to reduce the earable tracking drift by 27% and power usage of smartphone tracking by 65%.

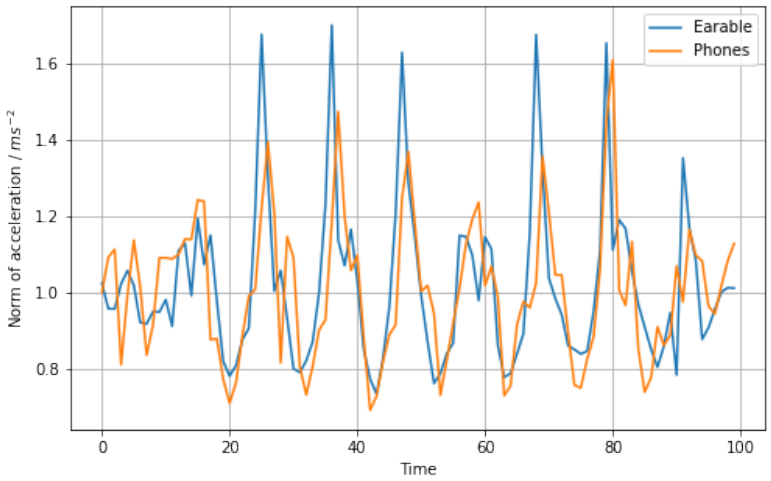


Fig. 1.1: Norm of acceleration of phone and earable

Compared to other mobile devices, earables also present a number of specific challenges. Earables need to be small to comfortably sit in or on a user's head for long periods of time. When creating my prototype, I am mindful of components and features to include to keep the device wearable.

Additionally, magnetometers, which are an important part of tracking, experience significant interference both from the magnet in integrated speakers and the electromagnetic field generated by Bluetooth audio transmission. In this project, I use Ferlini et al.'s semi-automatic calibration [1] to eliminate this noise, improving the performance of heading tracking by 37%.

1.2 Inertial Navigation

Inertial Navigation consists of a few stages. In the first stage, you take and pre-process raw data from the IMU. The raw data that comes from these sensors is noisy and often contains a temporal drift. The first challenge is to effectively calibrate these sensors. There are a number of possible calibration models, which have not been tested on earables. In this dissertation, I test calibration methods to produce an effective calibration framework.

Secondly, the pre-processed data is used to find the orientation of the device. The yaw can be found by using the magnetometer or gyroscope. This report shows that combining the two performs better. There are several algorithms to do this, I compare them in Chapter 6.

Finally, I use the heading and linear acceleration to find the change in (x, y) position of the user at each timestamp. In this dissertation, I adapt an existing pedestrian dead reckoning model to complete step length estimation effectively on earables.

1.3 Contributions

With the challenges of earables and inertial navigation in mind, it is possible to summarise the contributions of this dissertation as follows:

1. I design and manufacture two novel earable prototypes (Figure 1.2) for collecting and transmitting motion data (9-axis IMU) over Bluetooth Low Energy (BLE) and providing audio feedback. The device is designed to use minimal power and can run for over a day of continuous use. It also contains other sensors (temperature, pressure, microphone) which we don't experiment with, so the device can be used as a platform for earable experimentation in the future.

Fig. 1.2: Final earable prototype

2. I collect a new in-the-wild dataset collecting IMU data from earables and an iPhone. The user study in agreement with the university ethics committee captures data of six users walking both inside and outside, in low noise and high noise situations. There is no similar publicly available data, and this data will be shared with the research community.
3. I devise a comprehensive framework for IMU sensor calibration on earables, evaluating and adapting existing calibration methods for accelerometers, gyroscopes and magnetometers.
4. I evaluate heading estimation algorithms for earables, defining a novel effective method which leverage a complementary filter to combine gyroscopic and magnetometer heading.
5. I devise an effective pedestrian dead reckoning method adapting Lu et al.'s [6] method to estimate displacement.
6. I implement near real-time tracking on an iPhone wirelessly connected to an earable.
7. I create a novel method to guide users to a particular heading combining tracking and audio feedback on earables.
8. I propose a method to fuse measurements from two earables to improve the accuracy of tracking by 26.8%.
9. I create a novel smartphone tracking method using earables to reduce the power usage of accurate (<15m error) tracking by 69.0%

Chapter 2

Background

This chapter starts by putting my work into the context of existing systems, before moving onto discussing important related work.

2.1 Existing Complete Systems

Tracking users without GPS is not new. Generally, due to tracking drift, systems are used to supplement other navigation systems. For example, the Honeywell LaseREf [7] uses GPS and inertial navigation to maintain an accurate aircraft position.

With the invention of MEMS, INSs have been implemented on smartphones [8], smartwatches [9] and even smartglasses [10]. Loh et al.'s smartglasses work [10] provides an insight into possible gains that the muscoskeletal dampening effect could have for an INS system and motivates this dissertation for earables.

The audio feedback system I produce has been made before. However, tracking generally come from ultrasonic sensors [11] or computer vision [12]. Lee et al. [13] produced a tool that uses a gyroscope but this supplements computer vision.

2.2 Earable Prototype Design

In Chapter 3, I design a prototype earable. As part of my research, I looked at existing earables. One such earable is the the eSense platform [14], using an in-ear design. I also looked at more realistic 'lab prototypes' such as Ota et al. [15] and Pham et al.'s designs [16]. These used an around-ear design to allow for space for hand-soldered electronics. These prototypes were all 3D printed, out of ABS or PLA.

2.3 Inertial Navigation

As discussed in Chapter 1, inertial navigation consists of three stages.

1. **Pre-processing**: Calibration of IMU data.
2. **Heading Estimation**: Using IMU data to find the heading of the device.
3. **Displacement Estimation**: Using IMU data to find the change in position of the user.

2.3.1 Sensor Calibration

Sensor calibration is particularly important for MEMS IMUs. I first look at the effects that lead to noise before discussing correction methods.

Sources of Noise

The first source of noise is electrical noise from the circuitry that converts motion into a voltage. This is the most significant noise and manifests as a constant bias.

$$f(\mathbf{x},\mathbf{b}) \sim \mathbf{x} - \mathbf{b}$$

Thermo-mechanical noise arises from the changing temperatures of nearby components. Leland et al. [17] demonstrate that this noise is Gaussian. Woodman et al. [18] extend this to accelerometers.

$$f(\mathbf{x},\mathbf{b},\sigma_1) \sim \mathcal{N}(\mathbf{x}-\mathbf{b},\sigma_1)$$

Finally, temperature effects introduces biases which grow over time. These tend to be linear [19], but can also be more complex [18].

Hence, one can estimate the reading of a sensor as:

$$f(\mathbf{x},\mathbf{b}_1,\mathbf{b}_2,\mathbf{b}_3,...,\mathbf{b}_n,\sigma_1,t) \sim \mathcal{N}(\mathbf{x}-\mathbf{b}_1-\mathbf{b}_2 t-\mathbf{b}_3 t^2-...-\mathbf{b}_n^{n-1},\sigma_1) \tag{2.1}$$

The temporal drift does not exist for magnetometers, since they use a constant point of reference, the Earth's magnetic field. For a magnetometer, the main source of noise is magnetic interference from nearby devices.

Offset Finding

The simplest accelerometer and gyroscope calibration method involves finding the IMU static bias [20] combined with a moving window (empirically set to $\frac{1}{4}$s). Hence, the calibration is as follows:

For a stationary calibration clip:

$$\mathbf{b} = \frac{1}{n}\sum_{n=1}^{n}\mathbf{x} \tag{2.2}$$

For live sensor usage:

$$\mathbf{x}'_t = \frac{1}{z}\sum_{t'=t-z}^{t}\mathbf{x}_{t'} - \mathbf{b} \tag{2.3}$$

In practise this method doesn't always work well, so I consider specific accelerometer and gyroscope calibration methods.

Accelerometer Calibration

Frosio et al. Error Model Frosio et al. [21] leverage a model as follows, where A is the raw sensor output and A′ is the calibrated output:

$$\mathbf{A}' = \mathbf{S}(\mathbf{A}-\mathbf{O}) \tag{2.4}$$

$$\mathbf{S} = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}, \quad \mathbf{O} = \begin{bmatrix} O_x \\ O_y \\ O_z \end{bmatrix} \tag{2.5}$$

The authors describe a calibration process where the device is placed in several orientations, capturing accelerometer data for each. The error for a particular orientation k is equal to:

$$e_k = a_x^2+a_y^2+a_z^2-g^2 \quad = \sum_{i=x,y,z}\left\{\sum_{j=x,y,z}\left[S_{ij}\cdot\left(A_{j,k}-O_j\right)\right]^2\right\}-g^2 \tag{2.6}$$

The authors minimise the cumulative sum of squares of the errors e_k for all ks. They describe using Newton's method for quadratic convergence (described by Moré et al. [22]) to find the optimal values of **S** and **O**.

Sipos et al. Error Model Sipos et al. [2] describe a slightly different error model consisting of nine unknown parameters - three scale factor corrections, three angles of non-orthogonality and three offsets.

$$a'_p = \begin{pmatrix} 1 & 0 & 0 \\ \alpha_{yx} & 1 & 0 \\ \alpha_{zx} & \alpha_{zy} & 1 \end{pmatrix} \begin{pmatrix} \mathrm{SF}_{ax} & 0 & 0 \\ 0 & \mathrm{SF}_{ay} & 0 \\ 0 & 0 & \mathrm{SF}_{az} \end{pmatrix} \times \left(\begin{pmatrix} a_{mx} \\ a_{my} \\ a_{mz} \end{pmatrix} - \begin{pmatrix} b_{ax} \\ b_{ay} \\ b_{az} \end{pmatrix} \right) \tag{2.7}$$

Compared to Frosio et al.'s model, **S** is approximated into a separate transformation and scale factor matrix. This estimation reduces the likelihood of over-fitting to the calibration clips. Sipos et al. also describe a different algorithm for optimisation of the parameters, using the Levenberg-Marquandt Algorithm (described by Yamashita et al. [23]).

Magnetometer Calibration

Magnetometers experience minimal temporal drift, but often require frequent re-calibration to deal with differing background magnetic noise. Ferlini et al. demonstrate (Figure 2.1) [5] that for earables, the magnetometer experiences significant noise during audio playback.

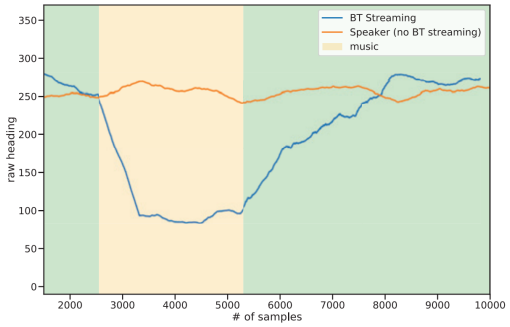


Fig. 2.1: Demonstration of additional magnetometer noise caused by music

Ferlini et al. propose a novel semi-automated calibration method [5] motivated by the average smartphone owner unlocking their phone 150 times a day [24]. When unlocking the device, a user's phone and earables are aligned. Hence, one can utilise a few seconds of earable readings at these times and the calibrated phone heading (which fuses GPS to maximise accuracy) to calibrate the earable magnetometer, using a simple offset model.

$$\mathbf{m}' = (\mathbf{m} - \mathbf{b}) \tag{2.8}$$

The error per reading is defined as:

$$e_i = \arctan \frac{y_{\text{raw}} - y_{\text{offset}}}{x_{\text{raw}} - x_{\text{offset}}} - \text{phone\_calibrated\_heading} \tag{2.9}$$

As with Section 2.3.1, the authors minimise the sum of squares of the heading errors of calibration points. However, this method assumes that the earth's magnetic field runs perpendicularly to the earth. This is not true other than at the equator, requiring tilt compensation.

Tilt Compensation For, tilt compensation, one first finds the pitch (θ) and roll (ϕ) angles from the accelerometer [25]. This can only be found when the accelerometer is at rest ($\|a\| \approx g$).

$$\phi = \arctan \frac{a_y}{a_z} \tag{2.10}$$

$$\theta = \arctan \frac{-a_x}{a_y \sin\phi + a_z \cos\phi} \tag{2.11}$$

With the angles θ and ϕ, the magnetometer readings are rotated to the flat plane where $\theta = \phi = 0$

$$\begin{pmatrix} m'_x \\ m'_y \\ m'_z \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta\sin\phi & \sin\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{pmatrix} \begin{pmatrix} m_x \\ m_y \\ m_z \end{pmatrix} \tag{2.12}$$

Gyroscope Calibration

Initially, for gyroscopes, it is possible to use the same models as for accelerometers since Woodman et al. [18] show that they encounter similar noise. The expected rotation rate of a gyroscope at rest is zero. However, this is not particularly effective and it would be better to calibrate the gyroscope when it is moving. Therefore, I consider two calibration methods.

Yang et al. Calibration Yang et al.'s model [26] includes a bias (**B**), scale factor (K), random error (v), non-orthogonality error (**M**) and a package misalignment error (**C**).

$$\begin{bmatrix} w'_x \\ w'_y \\ w'_z \end{bmatrix} = KC_b^n M \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix} + \begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \tag{2.13}$$

To find the correct values of the package misalignment error and scale factor, the authors require the gyroscope to be moving at a known rotational rate. In Yang et al. they utilise a single-axis turntable.

Shen et al. Calibration Shen et al. [3] propose a method to find the attitude of an IMU. As part of this, they describe a method to continuously recalibrate gyroscope readings. More precisely, the method recalibrates the heading angle measured purely by the gyroscope using the method described in Section 2.3.2.

To do this, they first filter accelerometer readings to find when $\|a\| \approx g$. Then, the authors find the difference between the heading measured by the gyroscope and the heading measured by the (calibrated) magnetometer. This is used to find a rotation offset.

2.3.2 Heading Estimation

Given a calibrated IMU, the next step is estimating the heading of the user.

Throughout the rest of this section, I am going to present the maths in terms of quaternions. Quaternions consist of a scalar part s $\in \mathcal{R}$ and a vector part $\mathbf{v} = (x,y,z)$, $\mathbf{v} \in \mathcal{R}^3$. Every unique rotation in 3D space can be represented by a unit quaternion.

Heading from Gyroscope

To get the heading from a gyroscope, I maintain a quaternion (q) per time step and integrate the measurements of the gyroscope [27]. This is done using a Taylor series:

$$\mathbf{q}_{t+1} = \mathbf{q}_t + \dot{\mathbf{q}}_t \Delta t + \frac{1}{2!}\ddot{\mathbf{q}}_t \Delta t^2 + \frac{1}{3!}\dddot{\mathbf{q}}_t \Delta t^3 + \cdots \tag{2.14}$$

This is generally truncated to the second term, as this expression has a rapidly reducing error. Hence, it's

necessary to find an expression for $\dot{\mathbf{q}}$. This is found through first principles and is described by Sola [28].

$$\dot{q} = \frac{1}{2}\begin{bmatrix} -\omega_x q_0 - \omega_y q_1 - \omega_z q_2 \\ \omega_x q_3 + \omega_z q_1 - \omega_y q_2 \\ \omega_y q_3 - \omega_z q_0 + \omega_x q_2 \\ \omega_z q_3 + \omega_y q_0 - \omega_x q_1 \end{bmatrix} \tag{2.15}$$

Given a quaternion for each time step, the attitude is converted to Euler angles:

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \arctan \frac{2(q_0 q_1 + q_2 q_3)}{1-2\left(q_1^2+q_2^2\right)} \\ \arcsin\left(2(q_0 q_2 - q_3 q_1)\right) \\ \arctan \frac{2(q_0 q_3 + q_1 q_2)}{1-2\left(q_2^2+q_3^2\right)} \end{bmatrix} \tag{2.16}$$

The heading is the yaw (ψ).

Heading from Magnetometer

Getting the heading from a magnetometer is much simpler:

$$\psi = \arctan \frac{m_y}{m_x} \tag{2.17}$$

In practice the results of the heading from the magnetometer and gyroscope should be combined. I survey a number of approaches that fuse these headings.

Complementary Filter

The first such method is a complementary filter (used by Shen et al. [3]). Here, one independently finds the heading using the magnetometer and gyroscope. These two headings can be any two headings.

The two headings are fused as follows:

$$\theta = (1-\alpha)\theta_\omega + \alpha\theta_m \tag{2.18}$$

The choice of α determines how much of each method is used. Shen et al. [3] suggest using a low value to fuse the gyroscope and magnetometer heading. Other papers suggest increasing the value of α over time, as the gyroscopic drift increases.

Madgwick Filter

This filter was originally designed to fuse gyroscope and accelerometer headings but was extended to also use magnetometer data [29]. Madgwick filters use gradient descent to enable better performance at lower sampling rates.

As with the complementary filter, the attitude (3D heading) is estimated from the gyroscope using numerical estimation. An attitude estimation is also found from the accelerometer using gradient descent to solve the following minimisation problem, which denotes finding the minimal rotation to take the acceleration to the gravitational force.

$$f = \min_{\mathbf{q} \in \mathcal{R}^{4\times 1}} \mathbf{q}^* \otimes \mathbf{g} \otimes \mathbf{q} - \mathbf{a} \tag{2.19}$$

In this approach, the heading from the accelerometer and gyroscope is fused with a complementary filter:

$$\mathbf{q}_t = \gamma_t \dot{\mathbf{q}}_{\omega,t} + (1-\gamma_t)\mathbf{q}_{\Delta,t+1} \tag{2.20}$$

Madgwick [29] also demonstrates how to add the magnetometer, minimising the error in accelerometer and magnetometer heading simultaneously by changing the optimisation function.

Mahony Filter

The Mahony Filter [30] is an improved filter focused in reducing computational complexity.

It consists of a few stages. In the first, the authors compute the orientation error from previous estimates using the accelerometer and magnetometer measurements.

$$\mathbf{v}\big(\mathbf{q}_{est,t}\big) = \begin{bmatrix} 2(q_2 q_4 - q_1 q_3) \\ 2(q_1 q_2 + q_3 q_4) \\ \big(q_1^2 - q_2^2 - q_3^2 + q_4^2\big) \end{bmatrix} \tag{2.21}$$

$$\mathbf{e}_{t+1} = \gamma(\mathbf{a}_{t+1} \times \mathbf{v}\big(\mathbf{q}_{est,t}\big)) + (1-\gamma)(\mathbf{m}_{t+1} \times \mathbf{v}\big(\mathbf{q}_{est,t}\big)) \tag{2.22}$$

$$\mathbf{e}_{i,t+1} = \mathbf{e}_{i,t} + \mathbf{e}_{t+1}\Delta t \tag{2.23}$$

The authors then compute the updated gyroscope measurements applying proportional-integral compensation:

$$\omega_{t+1} = \omega_{t+1} + \mathbf{K}_p \mathbf{e}_{t+1} + \mathbf{K}_i \mathbf{e}_{i,t+1} \tag{2.24}$$

Following this, the authors use the method described previously to find the heading.

Fourati Filter

The Fourati filter [31] is effectively a complementary filter. It fuses the magnetometer heading at low frequency regions, where it is expected to perform best, with the gyroscopic heading at high frequency regions.

2.3.3 Displacement Estimation

To find the change in position (i.e., displacement) of a user for any given timestamp, I use:

$$\begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} d\cos\theta \\ d\sin\theta \end{pmatrix} \tag{2.25}$$

The heading (θ) can come from one of the methods described above, whilst there are a couple of methods to calculate the distance moved in a timestep (d).

Kinematics

Firstly, you calculate the norm of the acceleration:

$$acc_t = \sqrt{a_{t_x}^2 + a_{t_y}^2 + a_{t_z}^2} - g \tag{2.26}$$

It is possible to isolate the x and y components of acceleration (in the correct plane) given a heading θ:

$$\mathbf{a_t} = \begin{pmatrix} acc_t\cos\theta \\ acc_t\sin\theta \end{pmatrix} \tag{2.27}$$

Then, the velocity of the user is updated:

$$\mathbf{v_t} = \mathbf{v_{t\text{-}1}} + \mathbf{a_t}\Delta t \tag{2.28}$$

and the position is:

$$\begin{pmatrix} x_t \\ y_t \end{pmatrix} = \begin{pmatrix} x_{t-1} \\ y_{t-1} \end{pmatrix} + \mathbf{v_{t\text{-}1}} + \frac{1}{2}\mathbf{a_t}\Delta t^2 \tag{2.29}$$

Pedestrian Dead Reckoning (using Step Length Estimation)

In pedestrian dead reckoning (PDR), we identify every step that the user makes when walking. This is used to identify the distance moved in each time step given the time taken to take a known length stride. This has two parts to it.

The first part involves finding the timestamps for steps. Xu et al. [32] suggest the following simple method for step detection for a smartphone IMU:

- Calculate the signal vector magnitude of the acceleration.

$$\text{SVM}_t = \sqrt{a_{x_t}^2 + a_{y_t}^2 + a_{z_t}^2} \tag{2.30}$$

- Run this signal through a low-pass filter with a 3Hz cut-off frequency.
- Identify (and pair) all the peaks and valleys in the smoothed signal, using an empirically chosen maximum and minimum amplitude.

Jimenez et al. [33] suggest a similar method (for a foot worn IMU) which calculates the local acceleration variance to remove gravity and highlight motion before thresholding to find steps. They then filter using the nature of foot swing. This is likely not useful for earables. There is no related work which completes step detection using earables.

Finally, Lu et al. [6] present a similar method to Xu et al. [32] using peak detection combined with prominence thresholding on a low-pass filtered signal of the norm of the acceleration.

Once we have found the timestamp for the steps, we find the distance moved per step. Methods for this include:

- Lu et al. [6] propose using **dead reckoning** to find the step length:

$$\mathbf{D} = K \cdot \iint \boldsymbol{a}(t) dt dt \tag{2.31}$$

  This assumes the velocity for each step starts at 0. The value of K is computed experimentally for each user.

- **Weinberg Model**: The Weinberg model [34] defines the following formula:

$$\text{Step\_Length} = k \cdot \sqrt[4]{a_{max} - a_{min}} \tag{2.32}$$

  where k is measured for each individual. This can also be automatically defined:

  - Ho et al. [35] define the following formula:

$$k = 0.68 - 0.37 \times \bar{v}_{\text{step}} + 0.15 \times \bar{v}_{\text{step}}^2 \tag{2.33}$$

  - Strozzi et al. [36] propose the following formula where β is a per sensor constant:

$$k = \beta \times \frac{1}{\sqrt[3]{a_{\max}(i)}} \tag{2.34}$$

- One can simply **measure the length of a stride** and assume that all strides are of the same length. This can use an empirical value or be **based on the user's height** [37].

$$\text{Step\_Length} = 0.43 \cdot \text{height} \tag{2.35}$$

Once you have the step time and step length, you can update the position at each timestamp.

For each step x occurring between timestamp i and j:

$$d_{t \in i,\ldots,j} = \frac{1}{\text{Step\_Length}_x} \tag{2.36}$$

Then, for each timestamp:

$$\mathbf{S}_t = \begin{pmatrix} S_{t_x} \\ S_{t_y} \end{pmatrix} = \mathbf{S}_{t-1} + \begin{pmatrix} d_t \cos\theta_t \\ d_t \sin\theta_t \end{pmatrix} \tag{2.37}$$

2.3.4 Estimation Theory

As part of tracking a user, two variables are fused to find the change in position: heading and distance moved in each timestamp. There is uncertainty in both variables. Therefore, I consider three methods to combine these variables probabilistically to find the most probable position.

Particle Filter

In a particle filter [38], one uses a large number of particles to simulate possible positions for the user. It consists of two stages: sampling and re-sampling.

In sampling, for each particle, at each timestamp, you sample a heading and distance moved. I assume a normal distribution, with an experimentally measured standard deviation. Therefore, these particles spread out over time to represent the divergence in possible positions.

Occasionally, you re-sample these particles, based on the probability of each existing particle. The probability of a particle could be based on an external (more reliable) measurement, like a GPS location. It could also be based on real-world information about the position. For example, a particle that is inside a building for an outdoor walker should have p=0. Finally, it could just be chosen to establish the particles at the centre of mass of the particles, but with a lower variance. This is the method used to smooth measurements.

Kalman Filter

A Kalman filter [39] has a similar approach but does not require the simulation of particles, estimating the positions with a normal distribution.

It has the two stages. In the prediction stage the Kalman filter produces an estimate of the current state variables (position) along with the uncertainty. This is based on a hidden variable. For localisation, this hidden state is the velocity.

One then uses the measurements of position given heading and distance moved at each timestamp (which have some level of noise) to combine with these estimates using a weighted average. This is the update stage. The weighting is based on the expected uncertainty of the prediction and measurement. This weighted average is then used to predict the hidden variable (the velocity) and the uncertainty in this value.

Markov Chain Monte Carlo

MCMC [40] makes use of a Bayesian assumption to make predictions of what *state* an item is in based on the *state* it was previously in and the transitional data. In this case, the hidden state is the location of the user. You observe the heading and distance moved, which are the transition variables, training to optimise

the transition probabilities between hidden states based on training data where the hidden state is known (when there is ground truth GPS location data).

Then, I could use the Viterbi algorithm to step through a new sequence of heading and distance transitions to parse the most probable path and identify the location at each time step.

Chapter 3

Earable Prototype Design and Implementation

To implement inertial navigation on an earable, we needed an earable containing an accelerometer, gyroscope and magnetometer. However, no commercial available off-the-shelf (COTS) earables contained a magnetometer. This is because the magnetometer, as previously discussed, was highly inaccurate when placed next to a speaker and Bluetooth circuitry. I wanted to test using a magnetometer with Ferlini et al.'s earable magnetometer calibration [5] which could allow tracking on an earable to be possible. For my prototype, I created a complete solution including a 3D printed enclosure, a custom PCB and software. This could act as an effective platform for future earable research containing an IMU, BLE transceiver, temperature sensor, pressure sensor, audio output and microphone.

3.1 Identifying Requirements

The requirements of the prototype were as follows:

- **Mechanics**
  - The physical prototype should contain all required electronics.
  - The prototype should allow for sufficient airflow to prevent overheating of the electronics (>45°C based on Ungar et al. [41]).
  - The prototype should be comfortable to wear.
- **Electronics and Software**
  - The device should sample accelerometer, gyroscope and magnetometer data with a frequency greater than 20 Hz.
  - The device should be able to connect to an Apple iPhone over Bluetooth to transfer data.
  - The device should have a battery life over five hours ensuring that it lasts for at least as long as an Apple AirPods [42].
  - The device should allow for audio output.

3.2 Electronics

3.2.1 Identifying Components

As part of the electronics I had a few decisions to make regarding components.

Microcontroller

I chose to use an Arduino Nano 33 BLE board to control the earables over other options such as using a custom chip and programming in C. Alternatively, I could have used a Raspberry Pi Zero, which offered greater computation power. However, its consumption of 0.4W compared to 0.05W for the Nano was significant

for a portable device. In using an Arduino board, I could utilise an open platform with a comparable level of portability, power usage and computation power to existing mobile devices. It also meant that the prototype could be used for future earable research beyond this thesis. The Nano also contained the required IMU (a 9-axis LSM9DS1) and BLE communications circuitry.

Battery and Power Management

Existing wearable systems have used various strategies for power storage, from harvesting solar energy [13] and friction [43] to varied battery technologies, such as Zinc Ion [44] and Lithium Ion [45]. I decided to use Lithium Polymer batteries, as used by Kos et al. [46] and Goncalves et al. [47], as they offered a good compromise of power density and cost. In particular, I chose a 250mAh (one cell) battery which was the correct shape to fit the designed casing.

I used the PAM2401 to act as a DC-DC Boost Converter and regulator to provide a steady 5V to the Arduino board and the MCP73831 to act as a charging regulator. These offered cost-effective solutions whilst being possible to hand solder.

Audio Output

To output music from a speaker, I would require significant additional circuitry, including an audio driver chip, SD card system and an Op-Amp. Hence, I decided that this should be left as an extension. Instead, I concentrated on producing audible tones using pulse width modulation (PWM) and a resistor. I also chose to include a digital switch, so that music could be switched between a connected iPhone and the Arduino. To do this, I used the TS3A24159DGSR, which had a 0.2Ω resistance between the input and output pins. Any higher resistance could impact the iPhone audio quality.

3.2.2 Schematic Design

These components were combined by looking at the datasheets to find the correct auxiliary circuitry for each chip, adding indicator LEDs for future debugging. Figure 3.1 shows this schematic, designed with EagleCAD [48].

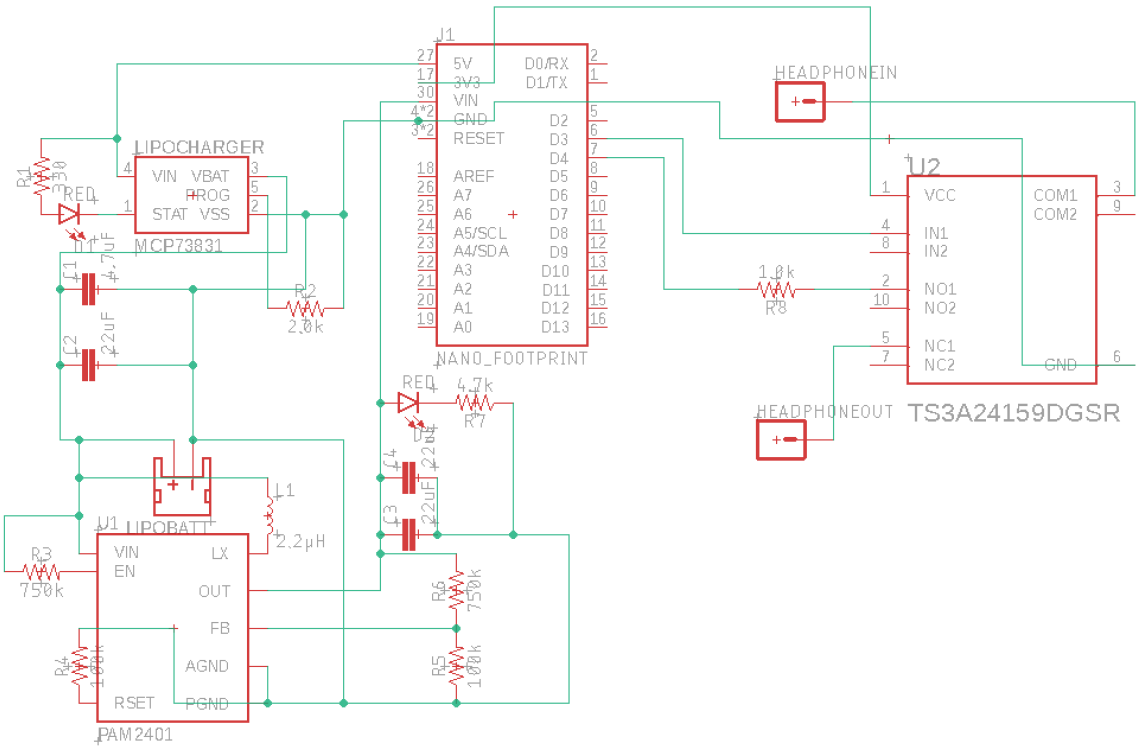


Fig. 3.1: Electronics Schematic

3.2.3 Printed Circuit Board (PCB)

The PCB was designed using EagleCAD, with components manually placed and nets routed on a two-layer PCB (43 x 18mm). I also placed a ground plane on the rear layer for safety reasons and to reduce the complexity of the routing. The final design can be seen in Figure 3.2. The Arduino Nano was soldered on top of the PCB. This could be done reversibly or conserving space (Figure 3.3).

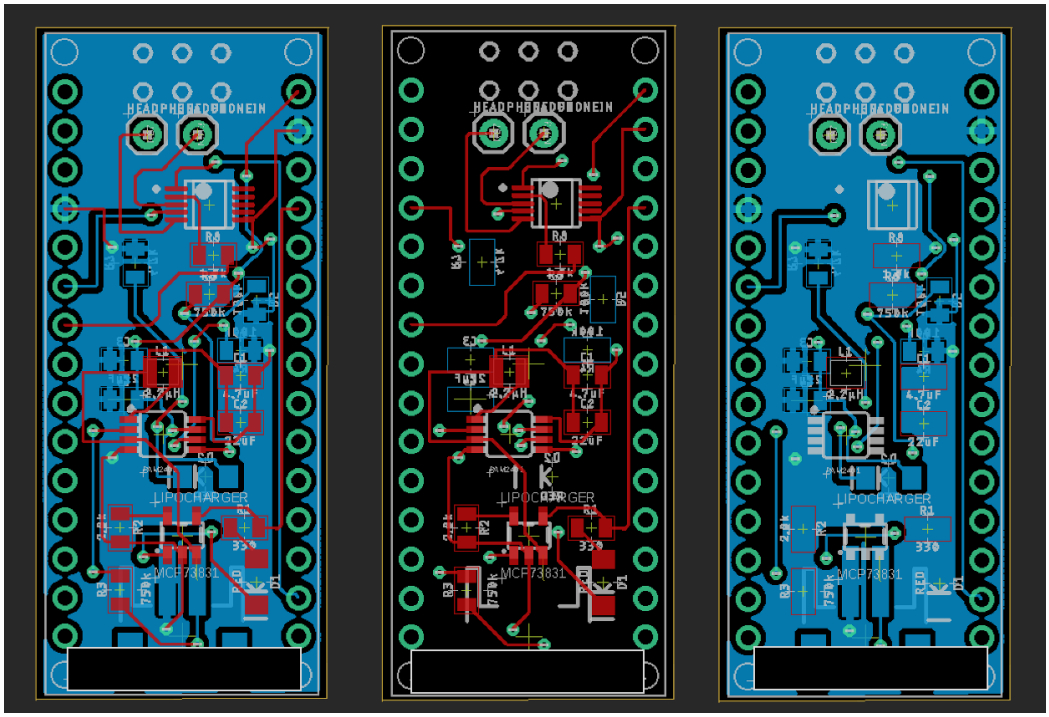


Fig. 3.2: PCB Design showing each layer

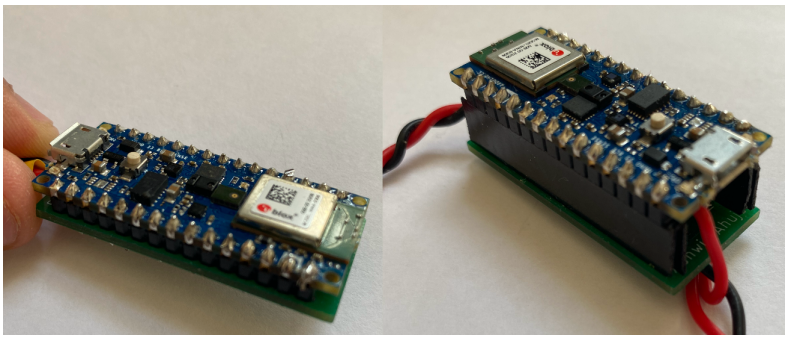


Fig. 3.3: Methods of board connection to PCB

3.2.4 PCB Assembly

Figure 3.4 shows a picture of the delivered PCBs. They were assembled by hand with care taken for the challenging QFN components. Figures 3.5 shows the complete soldered PCB.

In the future, reflow soldering could be used to ensure quicker and more reliable soldering.

To test the prototype, I considered each component in turn, using a multimeter to probe voltages. I verified that charging was working by monitoring the idle current draw. The current draw went from around 25mA to 500mA when plugged in, demonstrating the battery was charging. I demonstrated that the digital switch was working using an LED initially and then using an earphone. Figure 3.6 shows the complete system.

3.3 Mechanics

I chose to produce an around-ear design since it was be easier to manufacture. I initially attempted to emulate commercial off-the-shelf (COTS) around-ear headphones, but since they are injection molded using

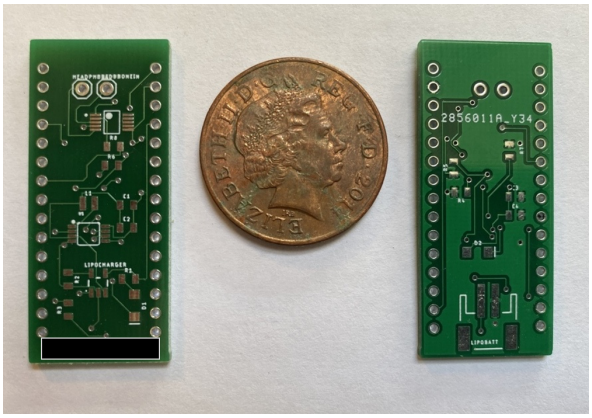


Fig. 3.4: Delivered PCB

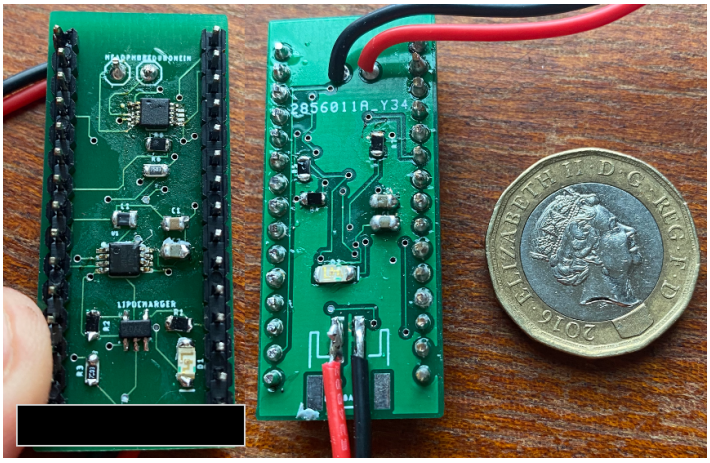


Fig. 3.5: Soldered PCB

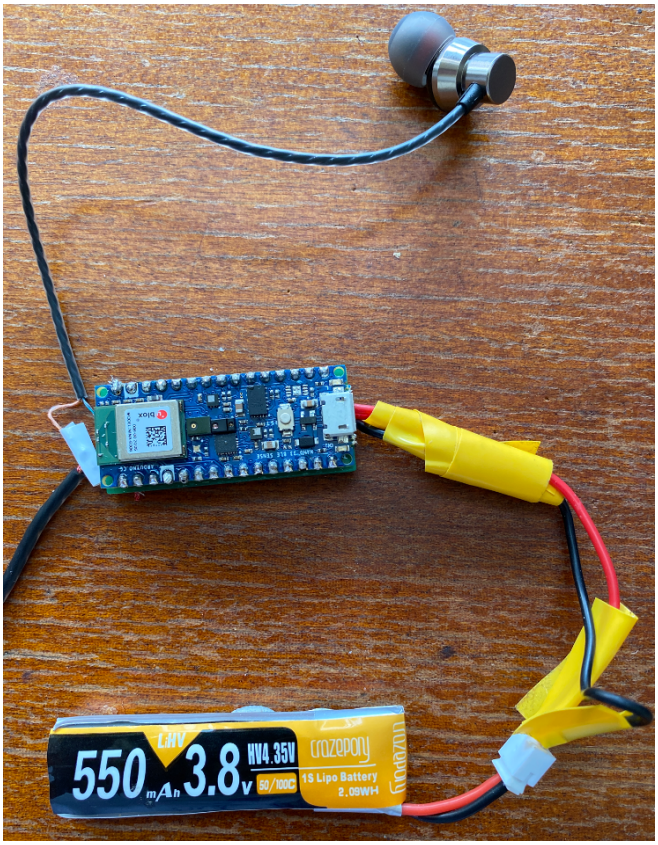


Fig. 3.6: Soldered full system

silicon, whilst my prototype was 3D printed, this proved challenging. The casing is designed using Autodesk Fusion 360 [49] and printed with an Ultimaker 2 using Polylactic Acid. A better material for a 3D printed earable would be the flexible Thermoplastic Polyurethane (TPU), however, this is challenging and expensive to print in. I leave testing my design with a TPU printer as future work.

The first iteration (Figure 3.7 and Figure 3.8) demonstrated the design idea. The ear dimensions were based on the author of the dissertation and Figure 3.9 demonstrates how it comfortably sat on the ears on the dissertation author. An initial test of the electronics inside the prototype demonstrated that the design was unnecessarily large. Additionally, with a larger battery, the weight balance of the device could mean that the prototype would fall off the author's ear.

In the second iteration (Figure 3.11 and Figure 3.10), I rotated the casing for the electronics to ensure the centre of gravity was as close to the centre of the ear as possible. The general size of the device was also shrunk.

The final iteration (Figure 3.12 and Figure 3.13) increased the size of the device slightly. This was prompted

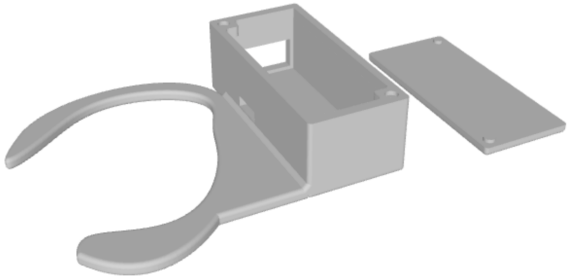


Fig. 3.7: First casing design

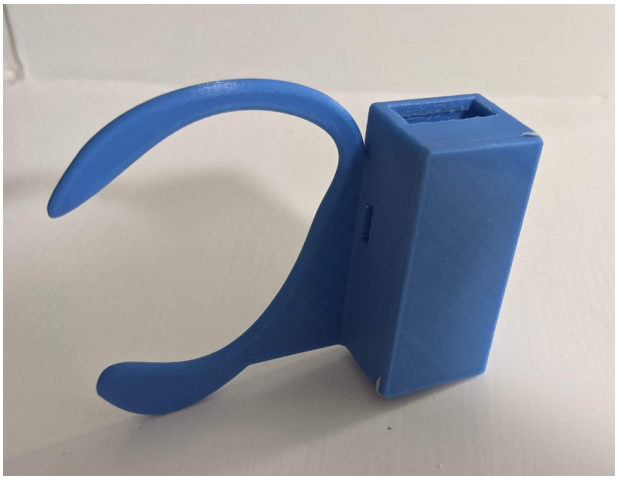


Fig. 3.8: First printed casing

Fig. 3.9: First casing design

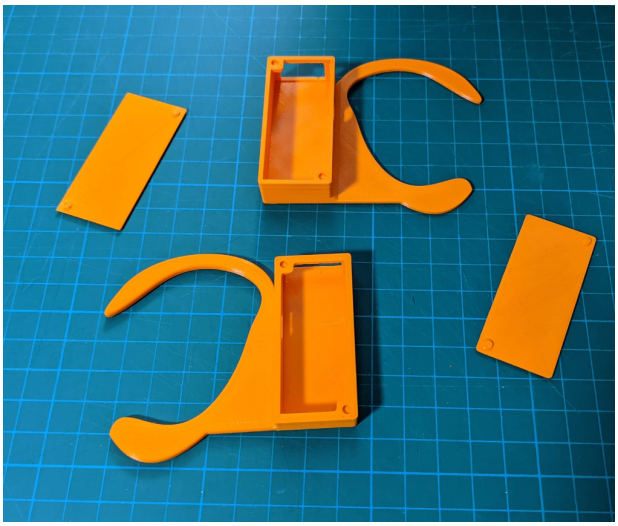


Fig. 3.10: Second casing printed

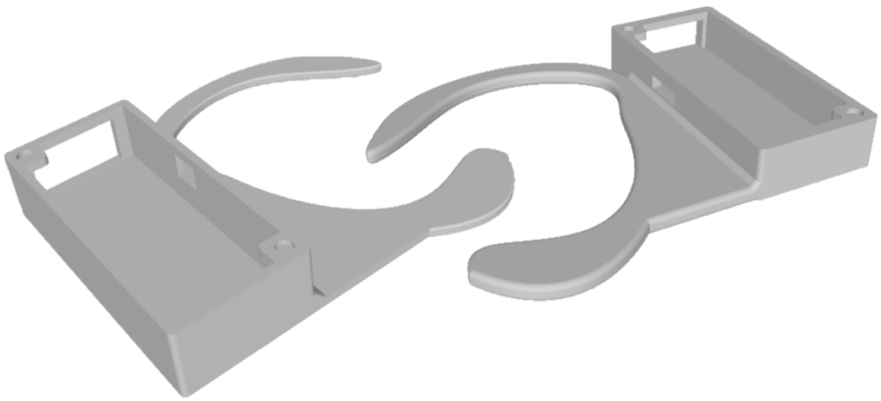


Fig. 3.11: Second casing design

by a change in the battery used, due to Brexit shipping delays. A larger 550mAh battery was acquired and the larger casing was made to accommodate this battery. Figure 3.14 shows the final device with all electronics inside. This was the device that I used for all the data collection.

3.4 Software

I aimed to use the prototype earables to collect data and transmit this to a connected iPhone for processing. However, the power of the Nano (and additional sensors) means that future research could use this prototype

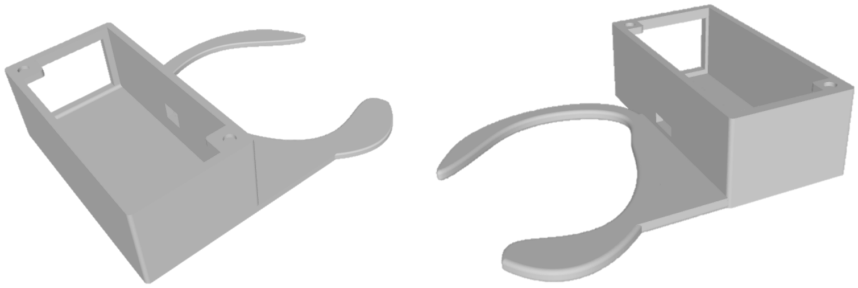


Fig. 3.12: Third casing design

Fig. 3.13: Third casing printed

Fig. 3.14: Final model with electronics

as a powerful earable platform. Bluetooth Low-Energy (BLE) was used for data transmission from earable to iPhone to minimise the power usage, since it uses less power than WiFi.

The earable software sampled IMU data (using a pre-written library [50]) at a specified rate and transmitted this as a 36 byte package (4 bytes per sensor per axis) to the iPhone. I assigned both of the earables as Bluetooth peripherals (see Figure 5.3), which the central, the iPhone application, connected to.

An iPhone application carried out data logging, as well as live tracking. More information on this is in Section 5.2. I tested data transmission at various sampling rates, using different sensors, measuring the current draw. With one earable connected, I found that the maximal transmission rate was 38.4Hz. With both devices connected, this was limited to 20Hz.

The results in Table 3.1 and Figure 3.15 demonstrate that there was a linear increase in current draw when increasing the frequency of updates. This difference is statistically significant[1] for all changes other than 2.5Hz to 5Hz. Changing from 5Hz to 40Hz reduced the battery life by two hours, a significant impact for earables.

The results for partial components of the IMU were unexpected. Individual parts never used less current

[1] using a paired t test with an α value of 0.05

Configuration		Actual Frequency / Hz	Current Draw / mA	Estimated Battery Life / hrs:mins
Completely Idle		N/A	15.4	35:42
Looking for connections		N/A	23.1	23:48
2.5Hz	**All**	2.48	19.5	28:12
5Hz	**Only Mag**	4.96	19.6	28:03
	Only Acc	4.96	19.6	28:03
	Only Gyro	4.96	19.7	27:55
	Mag + Acc	4.96	19.6	28:03
	Acc + Gyro	4.96	19.6	28:03
	Mag + Gyro	4.96	19.5	28:12
	All	4.96	19.6	28:03
10Hz	**All**	9.88	19.8	27:46
20Hz	**All**	19.59	20.2	27:13
30Hz	**All**	29.40	20.6	26:41
40Hz	**All**	38.44	20.9	26:18

Table 3.1: Table showing power draw testing results. Measurements (using UT658B) taken as average of 15 measurements taken every minute of 15 minutes running.

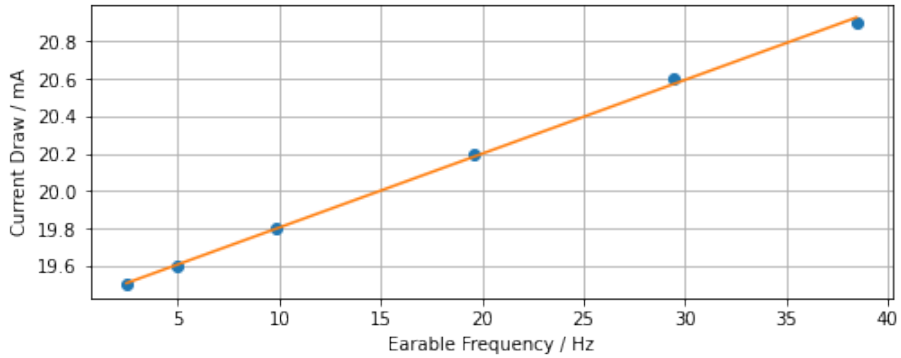


Fig. 3.15: Current draw for prototypes running at various frequencies

than the entire IMU. I discovered that the open-source LSM9DS1 library did not work as the datasheet instructed. When a part of the IMU was not initialised, it did not trigger the IMU 'off' signal. The LSM9DS1 datasheet indicates significant gains are possible if the IMU is controlled better.

In this chapter, I've created a new powerful, comfortable and portable earable prototype platform containing all the required sensors for tracking, as well as for future earable research.

Chapter 4

User Study

I performed a user study to collect IMU data from the earable prototypes under multiple situations. No similar has been collected and released. I hope to release this data widely as soon as possible.

This raw IMU data could be used to enable comparison of different calibration, heading estimation and displacement estimation algorithms discussed during Chapter 2. The experimental protocol is provided here. I received permission from all required parties (ethics committee, Computer Laboratory Safety Officer and college senior tutor) to conduct the experiment.

4.1 Experimental Protocol

I recruited six healthy subjects (3 male and 3 female) to complete the experiment. These were taken from the author's household bubble to allow an indoor test during the lockdown. The participants put on two prototype earables, one on each ear and an iPhone was provided to them. This logged IMU data from the earables and reference data from the iPhone as discussed in Section 5.2.1.

The experiment was split into two parts, involving data collection indoors and outdoors (Figure 4.1). This was motivated by the idea that the level and composition of noise would differ as discussed in Section 2.3.1. Additionally, the outdoor test provided a longer and more robust challenge. I could also capture ground truth location from GPS.

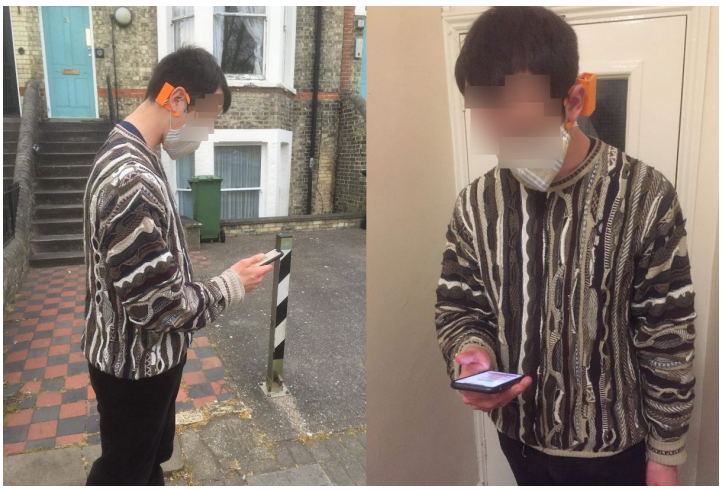


Fig. 4.1: Subject completing user study

Indoors, the participants were asked to walk up and down a corridor (of length 8m) five times, taking approximately four minutes. Outdoors, participants walked around a 750m route (≈11 minutes), visible in Figure 4.2.

In all tests, the users were asked to hold the iPhone in front of them to capture reference heading data. The following data was recorded from users:

- 9-axis IMU data from each earable
- Reference heading from iPhone

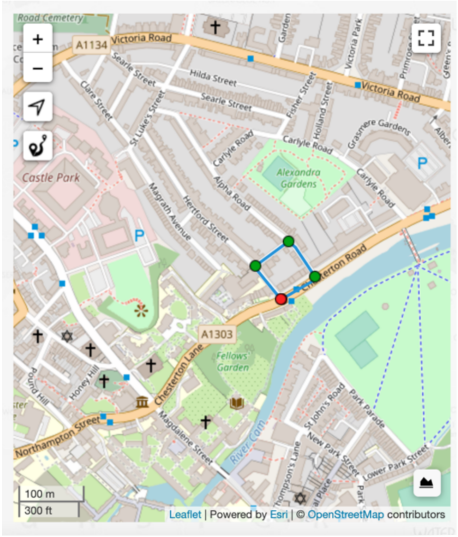


Fig. 4.2: Outdoor test route

Fig. 4.3: User study subject using headband to hold prototype earables

- 9-axis raw IMU data from iPhone
- GPS longitude and latitude from iPhone

I also tested with two different noise profiles for both indoor and outdoor tests. The first involved the user not listening to music and hence, had no earable magnetic interference other than background noise. The second involved the user playing music on headphones. Therefore, each participant perform four different tests:

- Indoors without music
- Indoors with music
- Outdoors without music
- Outdoors with music

Due to the COVID-19 pandemic, safety protocols were strict to ensure adherence to government guidance. For a list of precautions, see Appendix 8.1 which contains the full experimental application and consent form.

4.2 Issues encountered during user study

There were a few initial issues with the user study. The first problem was with attaching the earables to the participants' ears. As discussed in Section 3.3, the earables had been designed for the ear shape and size of the dissertation author. This meant that it did not fit all the participants. It would have been possible to print out specific designs for each participant, but this would have been time-consuming and costly. Therefore, I used a headband to hold up the earables where necessary (Figure 4.3).

The earable was also particularly fiddly for those with long hair or wearing glasses. This would be an important consideration for future mechanical designs. However, there was no quick fix and thus, care was taken to ensure that the earables were comfortable for all participants, by placing the earables underneath glasses and asking the participants to lift up their hair until the earables were put in place. No pain or discomfort was reported by any participant.

Finally, I planned to power up the earables using the power connector once the prototype was attached to the participants. This proved challenging as I had to ensure that no hair got into the power connector for those with long hair. To fix this, I added a SPDT power switch.

Chapter 5

Inertial Navigation

In this chapter, I discuss the software required to identify the best tracking framework for an earable. This includes implementing algorithms discussed in Chapter 2 for sensor calibration, heading estimation and displacement estimation with a Jupyter notebook. We discuss these implementations, motivating our choices of algorithms to implement. The results from running the data collected in the user study through this offline implementation are discussed in Chapter 6 and used to define the best calibration, heading estimation and displacement estimation methods for tracking an earable.

In this chapter, we also propose a real-time implementation of tracking on earables, using a connected iPhone application. This implementation would allow us to look at the required latency to obtain accurate tracking, as well as looking at the CPU and memory usage of continually tracking on an earable and comparing this to GPS tracking.

5.1 Offline INS implementation with Jupyter Notebook

The offline implementation used a Jupyter Notebook, allowing for simple and rapid experimentation and use of open-source libraries. Initially, the raw IMU data from the user study was loaded as a CSV. The logging process is discussed in Section 5.2.1. To process this data, I implemented the algorithms discussed in Chapter 2 and in this section I explain my reasoning behind the chosen implementations.

5.1.1 Sensor Calibration

Accelerometer Calibration

For accelerometer calibration, I demonstrated that Sipos et al.'s and Frosio et al.'s error model [2, 21] were equivalent, so I only implemented one, the simpler model by Sipos et al. I also implemented a simple offset method as discussed in Section 2.3.1. Both of these required a stationary clip which was recorded for all assembled earable prototypes.

Magnetometer Calibration

I used an existing Python implementation of Ferlini et al.'s calibration which used a least squares minimisation using trust region reflective algorithm (TRR). I also implemented tilt compensation as described in Section 2.3.1. I chose not to test other techniques as I wanted a user transparent method. Manual calibration (such as compass swinging techniques by Bowditch et al. [51]) are known to be error prone and cumbersome for users.

An important consideration when using Ferlini et al.'s calibration was when to capture calibration points (using the phone as a reference heading) and how to use these points. To emulate a user lifting their phone occasionally (leading to a phone-earable alignment), I captured potential calibration points every ten seconds.

The second aspect considered filtering points and when to discard old calibration points and complete a full re-calibration. Filtering could ensure that points were taken at time such that the calibration offsets were effective. I proposed filtering to ensure points was taken when the user was moving with constant velocity ($\|a\| \approx g$). This was motivated by a similar approach for calibration filtering by Shen et al [3]. As discussed in the results, this proved effective, but required significant tuning, since a good threshold for 'zero' acceleration proved user dependent.

The importance of discarding old calibration points was demonstrated by Ferlini et al. in Figure 2.1. The bias in magnetometer readings varies over time, due to variation in music and background magnetic properties. I considered a number of discarding methodologies:

1. **Never discarding points**: Never discard any calibration points.
2. **Rolling window of n points**: Use n points for each calibration, where the n was empirically chosen.
3. **Rolling window of s seconds**: Discard old points after a period of time.
4. **Discarding every full rotation**: This method yielded consideration based on primary experimentation. Discard all calibration points when the heading rolls over ($359° \rightarrow 0°$ or $0° \rightarrow 359°$).

Gyroscope Calibration

For gyroscope calibration, I considered using the method defined by Yang et al. [26]. However, I did not have access to a turntable. Therefore, I instead simply complete an offset calibration with a stationary clip.

I also implemented Shen et al.'s gyroscope heading calibration [3]. The algorithm was as follows:

```
for point in data:
    uncorrected_heading = update_heading(point.gyro_data)
    corrected_heading = uncorrected_heading − current_gyro_offset

    if(norm(point.acc_data) < n)):
        current_gyro_offset = angle_diff(uncorrected_heading − magnetometer_heading)
```

Algorithm 5.1: Shen et al. Heading Algorithm

where angle_diff finds the difference between two angles (Algorithm 5.3).

5.1.2 Heading Estimation

For heading estimation, I implemented most of the methods described in Chapter 2. To get the heading from the gyroscope I used the method described in Section 2.3.2, using an implementation from the AHRS library [52]. This library was also used to implement both Madgwick and Fourati filters. Finally, I implemented the heading from magnetometer and complementary filter as described in Chapter 2. I did not implement the Mahony filter, since the Madgwick and Fourati filters were expected to perform more accurately, with the Mahony filter prioritising efficiency over accuracy.

5.1.3 Offline Displacement Estimation

For displacement estimation, I implemented both the basic kinematics method and pedestrian dead reckoning method described in Section 2.3.3. For step identification, I adapted Lu et al.'s method [6]. The algorithm was:

```
Pass norm of accelerometer data through a low pass filter with a 3Hz cut−off frequency.
Find peaks (local maxima) of this filtered data using SciPy.
Find peaks that have a
    ↪ prominence above a certain empirically chosen value. The SciPy library was used to find peak prominences.
```

Algorithm 5.2: Lu et al. Step Detection Algorithm

I chose to implement this method since it was reasonably simple. Jimenez et al.'s method was specific for foot IMUs and would not transfer well to earables. Additionally, I found that whilst peaks were easy to spot in earable data (Figure 5.1), valleys were less distinct.

To find peaks, I used SciPy's find_peaks function [53] which finds maxima using a simple comparison of neighbouring values. The prominences were found using the find_prominences function. The histogram in Figure 5.2 was used to empirically set a peak prominence threshold of 0.2.

In preliminary testing, I found that complex stride length estimation models worked but not consistently across individuals. In particular, I experimented with using Weinberg's model with Ho et al.'s formula for

k. It appeared to require a scale factor tuning per user. Therefore, I estimated each user's stride based on their height, which performed much more consistently.

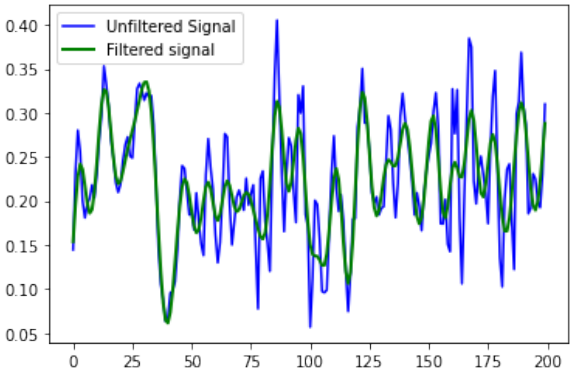


Fig. 5.1: Low-pass filtering of accelerometer signal

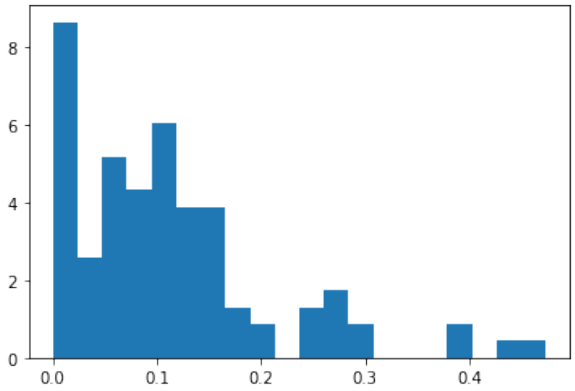


Fig. 5.2: Histograms of peak prominences

5.1.4 Estimation Theory

Of the three methods considered in Section 2.3.4, I implemented a particle filter and Kalman filter, since they were the simplest and most used methods in the sensor fusion community.

The particle filter was implemented manually, while, for the Kalman filter, the PyKalman library [54] was used.

5.2 Online INS implementation with iPhone Application

I decided to create the real-time tracking system for an iPhone application, since this was the primary smartphone owned by the author. Additionally, it allowed us to utilise some code written by Ferlini et al. [5] from their magnetometer calibration work. The existing code connected to a single earable, logging phone heading and magnetometer data.

5.2.1 Bluetooth Communications

Ferlini et al.'s code was extended to allow communications with two earables simultaneously. I used the Core Bluetooth library to establish the iPhone as the central device (Figure 5.3). This then searched for earable devices with a specific ID. The same ID was shared by both earables, which had the same ID but a different device name (*Left* or *Right*). Once the device with matching ID was found, a connection was automatically established. When a packet of IMU data arrived, it triggered an application interrupt to process the packet.

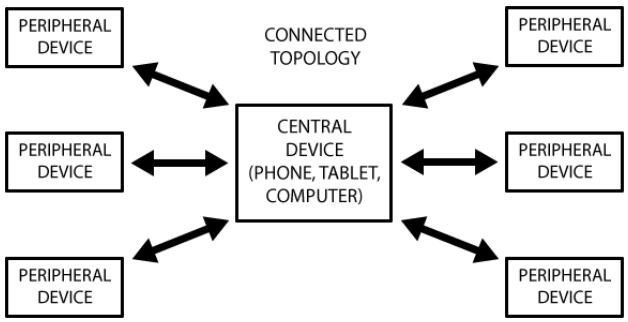


Fig. 5.3: BLE topology

As per Section 3.4, data was transmitted as a 36B packet from each earable. This was then split into nine 4B variables. For each packet, I log:

- Timestamp of packet arrival

- Earable name: 'left' or 'right'
- Earable IMU data: Magnetometer (x, y and z), Gyroscope (x, y and z) and Accelerometer (x, y and z)
- Phone IMU data: Magnetometer (x, y and z), Gyroscope (x, y and z) and Accelerometer (x, y and z)
- Phone Heading (found by fusing GPS and magnetometer)
- Phone GPS latitude and longitude

On a disconnection event, the log was processed and converted into a CSV file, saved on the iPhone. They then completed the experiment, before disconnecting the earables by turning them off. The CSV could be exported by emailing the file or plugging in the phone. I considered using a server to continuously upload data, but this was unnecessarily time-consuming for minimal gain, since I intended to complete real-time tracking on the iPhone, rather than on a server.

5.2.2 User Interface Designs

I designed four screens for the application. The first (Figure 5.4) was a home screen. This was designed according to Gestalt's Laws [55]. Foreground and background was made clear using colour. The similar buttons were symmetrically grouped to abide by continuity, symmetry and similarity laws.

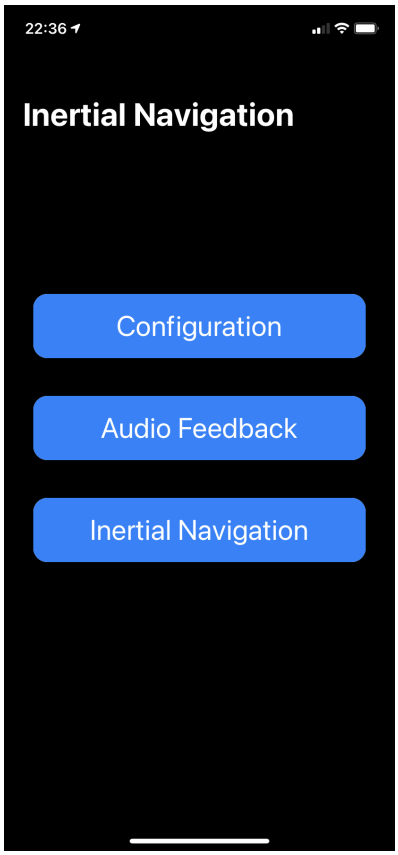


Fig. 5.4: Main screen

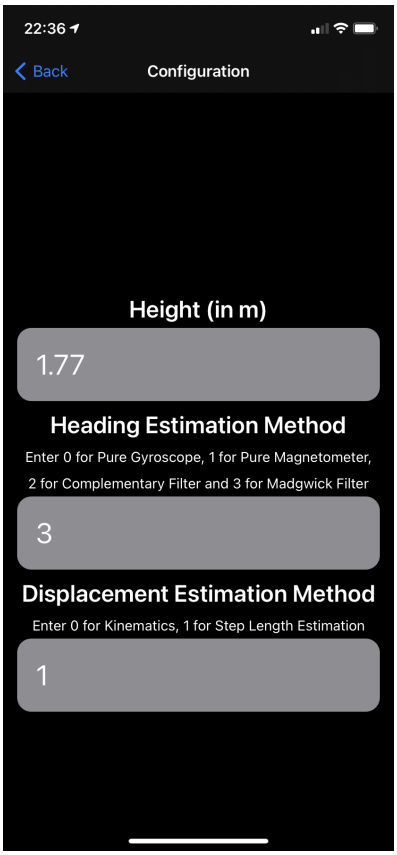


Fig. 5.5: Configuration screen

Clicking the 'Configuration' button took the user to a screen (Figure 5.5), where they could set the configuration of the application. The first box asked for the user's height, which was used during pedestrian dead reckoning. The screen also offered several heading estimation options and two different displacement estimation methods. The latter are both discussed in Section 5.1.3. Meanwhile, the heading estimation options were a subset of those discussed in the offline implementation. I implemented the gyroscope-only, magnetometer-only, Madgwick filter and complementary filter strategies. I intended the option selection to use robust and reliable drop-down menus. However, this was not a native feature in iOS and requires significant work to implement. Thus, I used a text input field.

Clicking the 'Audio Feedback' button took the user to a new screen (Figure 5.6). This had two main parts.

The top half showed the calibrated heading, the heading aim and two buttons to start and stop the audio feedback. Section 5.2.4 discusses what happens on this screen.

At the bottom of the screen, the connected earables were displayed. The timeline of connecting devices (and screens for these parts) is shown in Figure 5.8.

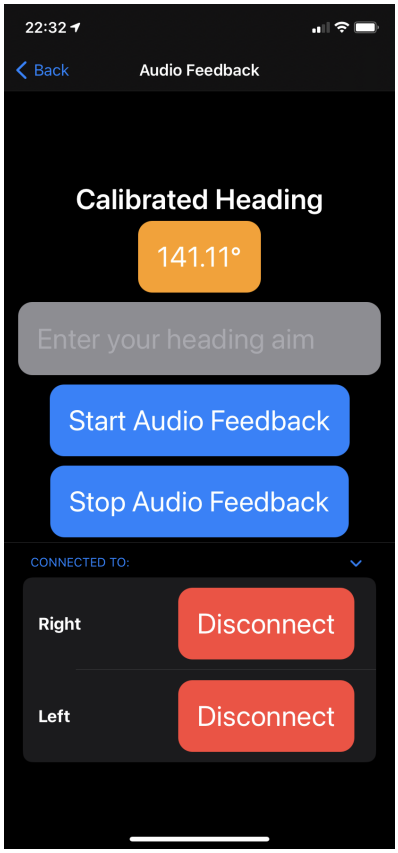


Fig. 5.6: Audio feedback screen

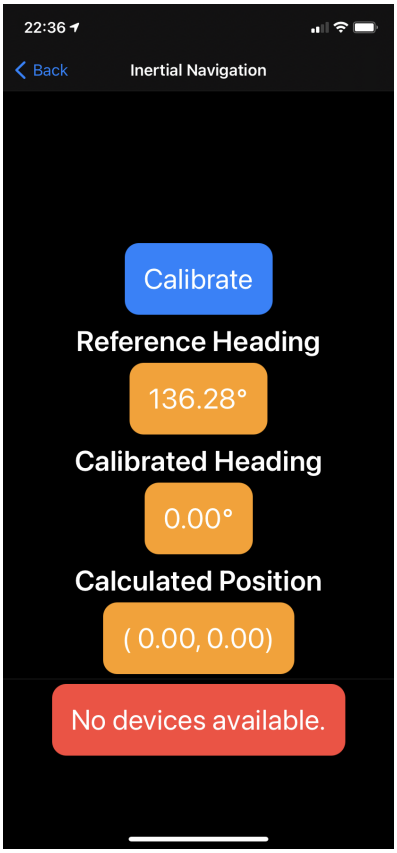


Fig. 5.7: INS screen

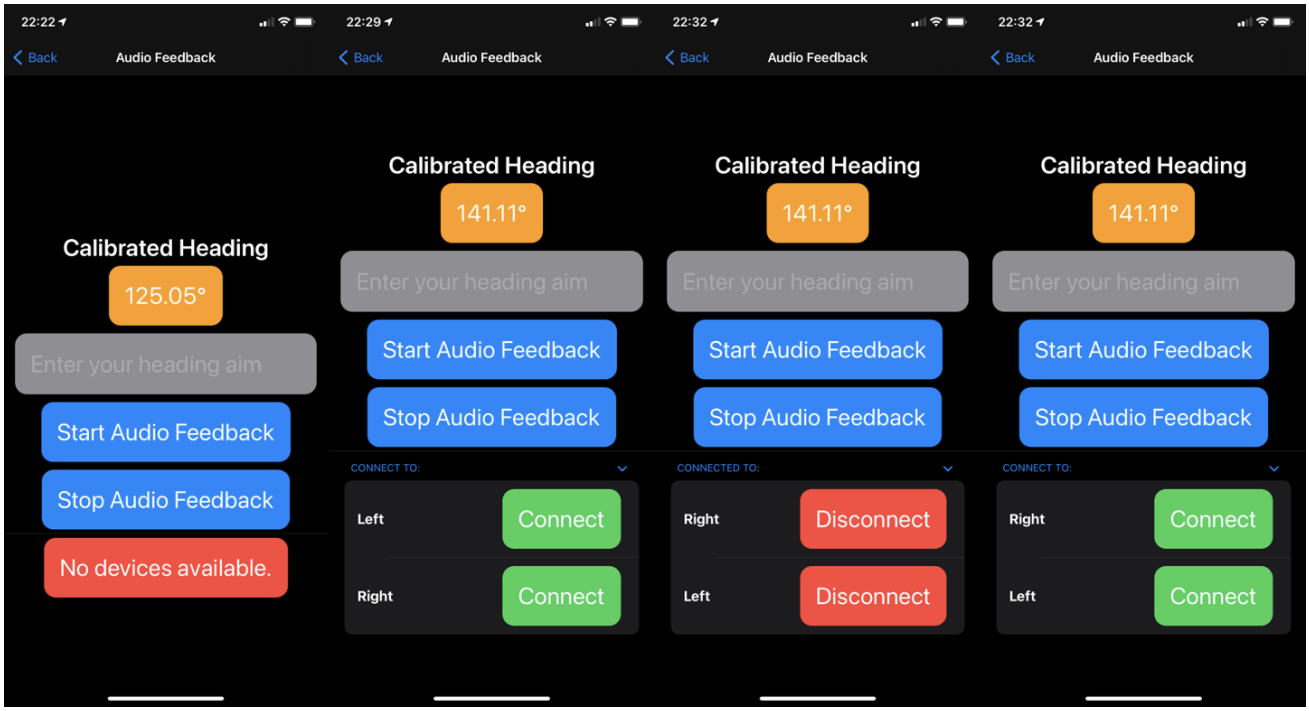


Fig. 5.8: Timeline of connections

Finally, the inertial navigation screen (Figure 5.7) also had two halves. The bottom half was the same as that in the audio feedback screen, and followed the same timeline. The upper half contained the reference

heading, calibrated heading and the calculated position. These were found using the methods chosen by the user in the configuration screen. Additionally, a 'Calibrate' button allowed users to indicate they are looking at the screen, to calibrate the magnetometer.

5.2.3 Online Heading Estimation

For heading estimation, I implemented a few of the methods described in Section 5.1.2. The choices of methods was based on primary offline testing discussed in Chapter 6.

For accelerometer calibration, I implemented Sipos et al.'s calibration model [2], copying across the calculated correction parameters. For magnetometer calibration, I used a filtered approach to generate a calibration point once the user pressed the calibrate button as long as the device was static, throwing away points after a rolling points window of ten points. This was shown in my initial testing to be a very effective and reliable method. For least squares minimisation, I used an existing online implementation [56].

For the gyroscope-only heading estimation method and Madgwick filter, I manually converted the offline Python code into Swift. Finally, for the complementary filter, I chose to set $\alpha = 0.8 - \frac{t}{400}$, based on preliminary testing.

5.2.4 Audio Feedback

To demonstrate an advantage of earables in tracking in addition to potentially higher accuracy, I created a demonstration application. Here, I used a calibrated heading from one of the methods discussed above to find the user's attention heading. Given a target heading, the application outputs a tone corresponding to how close a user was looking to this target. This could be useful in settings such as a complicated intersection to help identify which road to walk down.

I outputted the audio directly from a phone, rather than passing audio via the Arduino earable prototypes. This method was much simpler, not requiring two directional traffic between the iPhone and earables. I defined the frequency of the tone as:

$$f = 2750(\frac{180 - \text{diff}}{180}) + 250 \tag{5.1}$$

where diff is the difference between current calibrated heading and the heading aim:

```
def diff(a, b):
    if (a > b):
        return min(a−b, 360+b−a)
    else:
        return min(b−a, 360+a−b)
```

Algorithm 5.3: Angle Difference Algorithm

This mapped the difference to a frequency between 250 and 2750Hz. These tones empirically sounded clear, audible and comfortable.

5.2.5 Online Displacement Estimation

I implemented both the kinematics method and pedestrian dead reckoning methods. The kinematics method was simple to convert to real time, as it had no time reliance. Given the state and the current data, I could find the next predicted position.

The pedestrian dead reckoning method was a bit more challenging to convert into real-time. It required finding peaks in the real-time accelerometer signal. Therefore, I chose to add a delay of two seconds, empirically chosen to allow us to reliably find peaks whilst minimising the delay. To complete low-pass filtering, I manually implemented a first-order filter using the following formula:

$$\text{output\_value} = \frac{1}{3}\text{input\_value} + \frac{2}{3}\text{previous\_value} \tag{5.2}$$

I chose to use a first-order filter due to its simplicity and efficacy. Figure 5.9 demonstrates at a glance that a first-order filter does nearly as well as a fifth-order filter in helping to identify peaks in the norm of the acceleration.

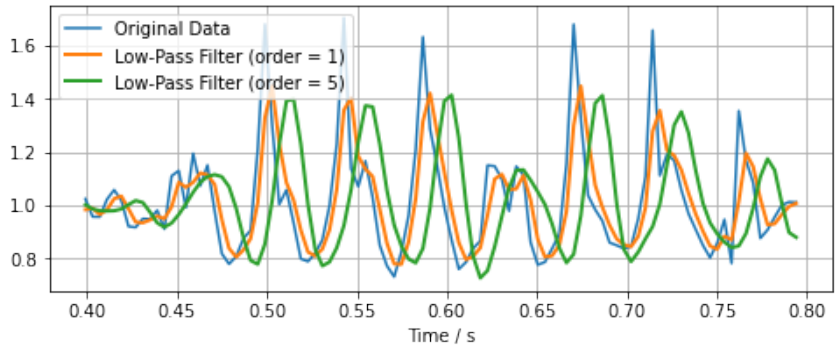


Fig. 5.9: Output of low-pass filters

To find peaks, I utilised a smoothed z-score algorithm used by Bernardi et al. [57] and implemented in Swift online [58].

In this chapter, I demonstrated how to capture raw data transmitted by the prototype earables. This data could either be processed offline using the Jupyter notebooks with a wider range of possible sensor calibration, heading estimation and displacement estimation methods. The offline software is used in Chapter 6 to evaluate the implemented methods using data collected from the user study. The best methods are combined to define a new framework for tracking on an earable. The transmitted data can also be processed to track the heading of a user in real-time and position of a user with a latency of two seconds, using a new iPhone application.

Chapter 6

Results and Discussion

In this chapter, I discuss the various calibration and tracking algorithms implemented. These were evaluated using the data collected in the user study, with metrics described in Section 6.1. Throughout, I completed significance testing using a paired t test comparing results for the same user. The null hypothesis is that the two compared systems are the same, and I used an α value of 0.05.

I firstly look at the impact of data frequency on the heading and displacement error, finding that the higher the frequency, the lower the errors. This appears to be a linear relationship until 10Hz. I then consider the effect of sensor calibration, finding that Shen et al.'s gyroscope and Ferlini et al.'s magnetometer calibration perform the best, whilst Sipos et al.'s calibration and an offset calibration perform the same. I also propose a new calibration point strategy for Ferlini et al.'s method, combining the best performing tested strategies. Following this, I look at heading estimation and displacement estimation methods, finding that magnetometer-only methods on average perform the best but with a lower reliability. Pedestrian dead reckoning meanwhile, outperforms kinematics methods by 93.3%, with a significant velocity drift noted in the kinematics method. Finally, I investigate the impact of the environment, demonstrating that music does not significantly affect the accuracy, whilst the heading error is significantly reduced in the longer, outdoor test. These results allow me to define the most effective tracking strategy for one earable.

I also complete two other tests, showing that earable tracking outperforms inertial iPhone raw data tracking by over 35.6%, whilst also showing that the real-time tracking application has a low battery impact maintaining a reasonably good accuracy, performing 12% worse than the offline implementation.

6.1 Relevant Metrics

Each variable was tested in turn in this section against a set of tests. These included a variety of different heading estimation methods. The choice of tests was based on preliminary testing and is defined as follows.

- **Physical Environment and Noise Profile**: I tested the system indoor (without music).
- **Accelerometer Calibration**: I opted for Sipos et al.'s calibration.
- **Magnetometer Calibration**: I implemented Ferlini et al.'s magnetometer calibration with the following calibration point strategies:
  - Never discarding calibration points
  - Rolling window of ten calibration points
  - Rolling window of a minute
  - Discarding every full rotation
- **Gyroscope Calibration**: I leveraged offset calibration.
- **Heading Estimation Methods**: I utilised the magnetometer-only and gyroscope-only method as well as a complimentary filter with an empirical function of $\alpha = 0.8 - \frac{t}{400}$. The value of 400 was chosen based on preliminary testing and I consider the impact of multiple values in Section 6.4. I also consider the Madgwick and Fourati filters.

- **Displacement Estimation Methods**: I used pedestrian dead reckoning.

For each heading estimation method, I calculated the difference between the predicted heading and the ground truth reference heading. These were averaged across all the study volunteers. An example of the headings from all methods is shown in Figure 6.1.

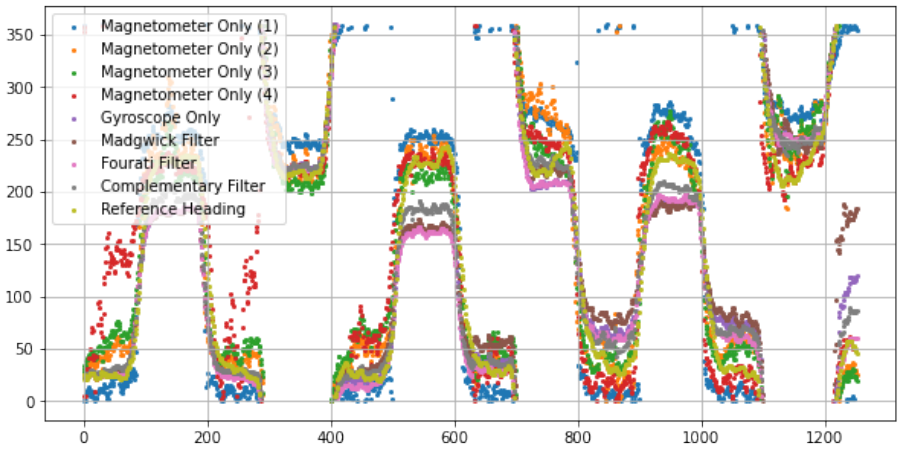


Fig. 6.1: Example of heading estimation

I also looked at the drift in tracking position over time. Indoors, there was no GPS to provide ground truth position. Therefore, I the test to start and end at the same point. Hence, I could find the drift as the normal of the final position divided by the testing time. An example path taken is shown for an indoor test is shown in Figure 6.2.

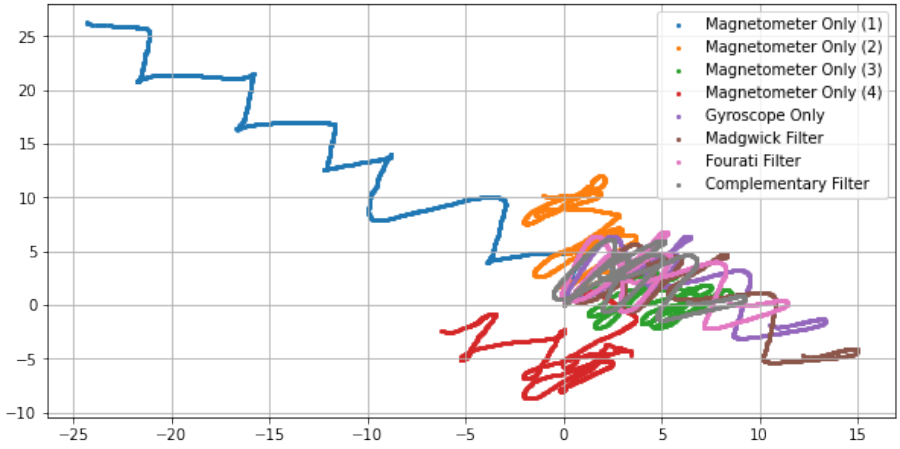


Fig. 6.2: Example of displacement estimation

6.2 Data Frequency

I considered the impact of earable data frequency on the heading accuracy and tracking drift. I found the data for different frequencies by discarding data. This was not necessarily equivalent to recording data slower, since sensors provide more accurate, better smoothed data when recording slower. Thus, it is possible that in real life, recording slower could perform better than the results showed.

The results in Table 6.1 demonstrate that increasing the frequency of data increases all heading accuracies and reduces the drift. Figure 6.5 demonstrates this trend appears linear. These differences were statistically significant for 2.5Hz to 5Hz (16.2%) and 5Hz to 10Hz (19.6%). However, the difference between 10Hz and 20Hz was not statistically significant.

In general, the standard deviations are quite high. This standard deviation arose from different users and not different methods. Therefore, the paired t test could still show statistically significant differences in average heading and displacements. The high standard deviations of magnetometer-only heading errors arose from

Frequency / Hz		2.5 Heading Error / °		Displacement Error / ms^{-1}		5 Heading Error / °		Displacement Error / ms^{-1}	
		Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Magnetometer Only	Never discarding points	30.1	26.6	0.3030	0.01007	30.4	25.1	0.1673	0.1000
	Rolling window of ten points	23.1	19.0	0.3010	0.01020	20.2	16.4	0.1671	0.0098
	Rolling window of a minute	27.4	22.3	0.3028	0.01010	22.4	18.3	0.1673	0.0100
	Discarding every rotation	32.0	28.6	0.3030	0.00983	23.3	22.3	0.1675	0.0099
Gyroscope only		29.6	18.6	0.3019	0.00892	26.0	19.8	0.1674	0.0101
Madgwick Filter		32.0	21.6	0.3014	0.00934	24.0	20.0	0.1675	0.0101
Fourati Filter		32.8	21.7	0.3017	0.00938	27.0	19.8	0.1677	0.0100
Complementary Filter		26.4	18.1	0.3018	0.00924	22.6	17.5	0.1673	0.0100
Frequency / Hz		10 Heading Error / °		Displacement Error / ms^{-1}		20 Heading Error / °		Displacement Error / ms^{-1}	
		Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Magnetometer Only	Never discarding points	29.5	24.1	0.1084	0.00407	24.3	19.8	0.4370	0.1780
	Rolling window of ten points	14.2	12.0	0.1084	0.00409	15.1	15.7	0.1330	0.0354
	Rolling window of a minute	15.7	11.8	0.1083	0.00412	19.4	18.6	0.2310	0.1010
	Discarding every rotation	17.2	17.2	0.1082	0.00395	15.9	15.9	0.2510	0.0686
Gyroscope only		21.4	15.5	0.1080	0.00416	24.5	24.0	0.1950	0.0290
Madgwick Filter		19.7	16.1	0.1081	0.00423	18.8	14.3	0.2170	0.0353
Fourati Filter		21.9	16.2	0.1080	0.00411	21.8	15.9	0.1800	0.0265
Complementary Filter		17.8	15.1	0.1081	0.00415	21.9	17.7	0.2450	0.0721

Table 6.1: Results comparing heading error and displacement error for different data frequencies

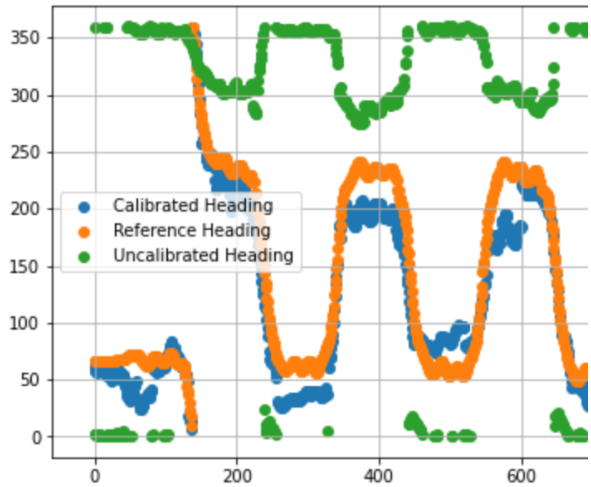


Fig. 6.3: Good magnetometer calibration

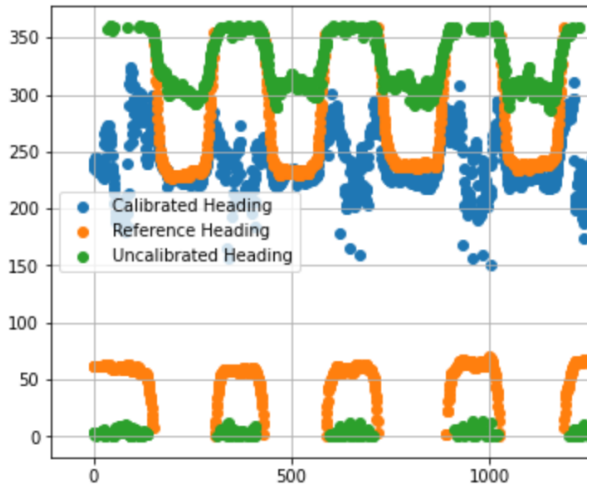


Fig. 6.4: Bad magnetometer calibration

'failed' calibrations. 5% of the calibrations failed (Figures 6.3 and Figure 6.4), but this significantly increased the standard deviation. The standard deviation for successful calibrations was under 15°. For the rest of the results, I chose to use 20Hz as the sampling frequency which performed the joint best.

6.3 Calibration Methods

In this section, I tested the various implemented calibration methods for accelerometers, gyroscopes and magnetometers. The best methods can then be combined to define a new calibration framework for earable IMUs.

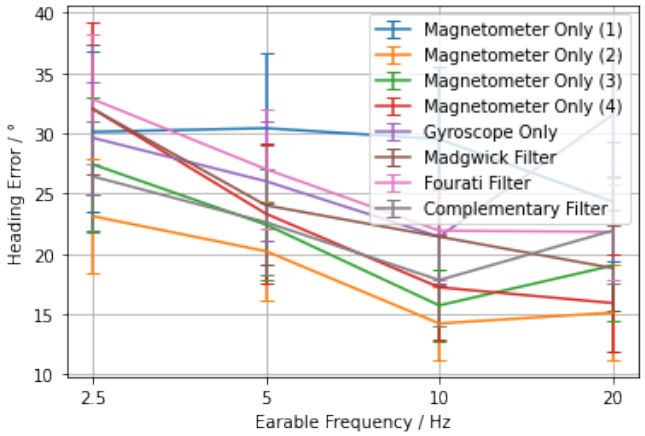


Fig. 6.5: Line graph showing heading error vs device frequency

6.3.1 Accelerometer Calibration

I compared the Sipos et al., offset calibration and no calibration.

Accelerometer Calibration Method		No Calibration Heading Error / °		No Calibration Displacement Error / ms^{-1}		Offset Calibration Heading Error / °		Offset Calibration Displacement Error / ms^{-1}	
		Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Magnetometer Only	Never discarding points	24.3	19.8	0.437	0.178	24.3	19.8	0.437	0.178
	Rolling window of ten points	15.1	15.7	0.234	0.0354	15.1	15.7	0.133	0.0354
	Rolling window of a minute	19.4	18.6	0.488	0.101	19.4	18.6	0.231	0.101
	Discarding every rotation	15.9	16	0.343	0.0686	15.9	16	0.251	0.0686
Gyroscope only		24.5	24.4	0.302	0.117	24.5	24.1	0.195	0.029
Madgwick Filter		26.7	19.7	0.431	0.105	19.5	15.7	0.228	0.0423
Fourati Filter		26.5	17.9	0.267	0.1	21.9	16	0.181	0.0275
Complementary Filter		25.8	19.2	0.328	0.105	22.7	18.5	0.247	0.0767

Sipos et al. Calibration Heading Error / °		Sipos et al. Calibration Displacement Error / ms^{-1}	
Average	Standard Deviation	Average	Standard Deviation
24.3	19.8	0.437	0.178
15.1	15.7	0.133	0.0354
19.4	18.6	0.231	0.101
15.9	15.9	0.251	0.0686
24.5	24	0.195	0.029
18.8	14.3	0.217	0.0353
21.8	15.9	0.18	0.0265
21.9	17.7	0.245	0.0721

Table 6.2: Results comparing heading error and displacement error for different accelerometer calibration methods

The results in Table 6.2 demonstrate a few things. The offset calibration (which uses only a bias and no scale factor) performed significantly better (p=0.004) than no calibration in terms of heading and displacement error. There was however, no statistically significant difference (p=0.23) between the offset calibration and Sipos et al.'s calibration. This suggests that the scale factor and non-orthogonality error was not particularly significant for the accelerometer. Secondly, whilst the clearest improvements are in the heading error for methods which use the accelerometer, there was also a statistically significant change in displacement error between no calibration and either calibration. This arose from the fact that the accelerometer was used for step detection.

6.3.2 Magnetometer Calibration

Here, I compare Ferlini et al.'s calibration with no calibration at all.

Magnetometer Calibration		Ferlini et al. Calibration				No Magnetometer Calibration			
		Heading Error / °		Displacement Error / ms^{-1}		Heading Error / °		Displacement Error / ms^{-1}	
		Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Magnetometer Only	Never discarding points	24.3	19.8	0.437	0.178	30.5	19.8	0.581	0.0567
	Rolling window of ten points	15.1	15.7	0.133	0.0354				
	Rolling window of a minute	19.4	18.6	0.231	0.101				
	Discarding every rotation	15.9	15.9	0.251	0.0686				
Gyroscope only		24.5	24	0.195	0.029	24.5	24	0.195	0.029
Madgwick Filter		18.8	14.3	0.217	0.0353	19.5	15.7	0.227	0.0417
Fourati Filter		21.8	15.9	0.18	0.0265	21.9	16	0.183	0.0283
Complementary Filter		21.9	17.7	0.245	0.0721	22.8	18.6	0.248	0.0205

Table 6.3: Results comparing heading error and displacement error for different magnetometer calibration methods

Table 6.3 shows that for the Magnetometer Only method, the Ferlini et al. calibration offered a significant improvement (38%). This improvement did not carry through to the Madgwick Filter and Fourati Filters. I believe this is due to their filters most significantly use the accelerometer and gyroscope for heading estimation, with little use of the magnetometer.

Of the strategies for choosing calibration points, the rolling window of ten points significantly outperformed the other strategies. I note that discarding points every rotation also performed very well. These two approaches could be combined to produce a rolling window which refreshes at every rotation.

Finally, I considered the impact of the filtering methods discussed in Section 5.1.1. This suggested excluding points if the user encounters significant acceleration. This was tested for the never discarding points method, offering a statistically significant improvement of 7%. However, it required significant tuning (per user) to select an accelerometer threshold. Too high a threshold would mean too few points, and often only outliers could be used for calibration, leading to failed calibrations. Too low a threshold meant that the impact of the filtering was minimal. With an effective tuning, however, filtering could be added to the best calibration point strategy discussed. In particular, I hypothesise that introducing smoothing and a threshold relative to peaks would be an effective automatic tuning method.

6.3.3 Gyroscope Calibration

I compared the offset calibration and Shen et al.'s calibration with no calibration.

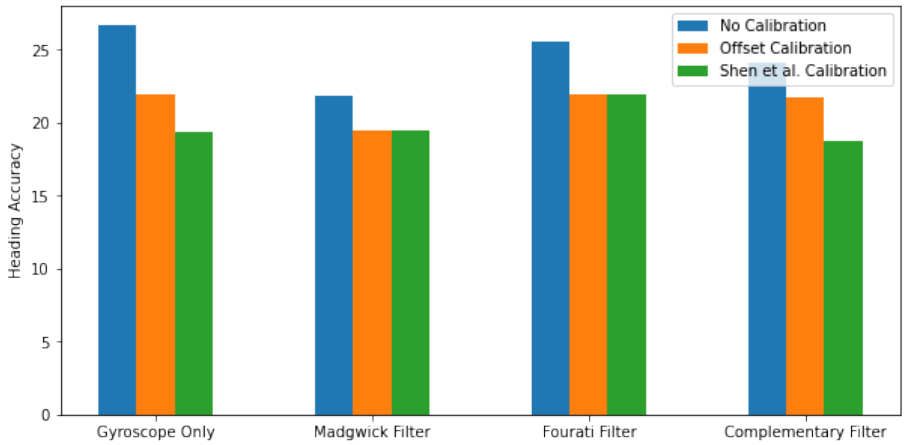


Fig. 6.6: Bar chart showing the impact of various gyroscope calibration methods on average heading accuracy

Gyroscope Calibration	No Calibration				Offset Calibration			
	Heading Error / °		Displacement Error / ms^{-1}		Heading Error / °		Displacement Error / ms^{-1}	
	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Gyroscope only	26.7	17.8	0.177	0.0429	24.5	24.0	0.195	0.029
Madgwick Filter	21.8	16.7	0.207	0.0408	18.8	14.3	0.217	0.0353
Fourati Filter	25.6	17.3	0.162	0.0381	21.8	15.9	0.18	0.0265
Complementary Filter	24.1	19.9	0.236	0.0521	21.9	21.7	0.245	0.0721

Shen et al. Calibration			
Heading Error / °		Displacement Error / ms^{-1}	
Average	Standard Deviation	Average	Standard Deviation
19.4	19.7	0.158	0.0477
19.5	15.7	0.227	0.0417
21.9	16	0.18	0.0277
18.8	20.4	0.136	0.0507

Table 6.4: Results comparing heading error and displacement error for different gyroscope calibration methods

The results (Table 6.4) were as expected, with the offset calibration performing better than no calibration (by 13%) and the Shen et al. calibration performing better than offline calibration (by 7%).

6.4 Heading Estimation Method

Using the best earable calibration framework defined above, I compared the various heading estimation algorithms.

		Heading Error / °		Displacement Error / ms^{-1}	
		Average	Standard Deviation	Average	Standard Deviation
Magnetometer Only	**Never discarding points**	24.3	19.8	0.437	0.179
	Rolling window of n points	15.1	15.7	0.133	0.0354
	Rolling window of n seconds	19.4	18.6	0.231	0.101
	Discarding every rotation	15.9	15.9	0.251	0.0686
Gyroscope only		24.5	24.0	0.195	0.0290
Madgwick Filter		18.8	14.3	0.180	0.0265
Fourati Filter		21.9	16	0.181	0.0275
Complementary Filter	**0.25**	15.8	16.5	0.172	0.0314
	0.5	18.9	19	0.202	0.0386
	0.75	21.2	17.9	0.193	0.0268

Table 6.5: Results comparing heading error for different heading estimation methods

Table 6.1 demonstrates that the magnetometer methods had the potential to perform the best, but was less reliable, with the highest relative standard deviation. It was continuously calibrated directly against the reference heading, with minimal drift, therefore the magnetometer method performed very well. I do note however, that with Shen et al.'s calibration, methods that use a gyroscope performed comparably. Shen et al.'s method seemed to use the low frequency magnetometer accuracy (especially when well calibrated) and high frequency gyroscope accuracy. In particular, it removed the gyroscopic drift exhibited by the Madgwick and Fourati filters. Figure 6.7, which came from an outdoor test, demonstrates that Fourati exhibit a lower drift than Madgwick filters, however, on indoor tests, the Madgwick filter consistently outperformed the Fourati filter. This suggests that the method for the Fourati filter performs better in the long term, but less well in the short term.

For the magnetometer methods, the variance in results arose from the occasional failed calibration. I can conclude that the best method for a short, indoor test, is either the magnetometer only method (with the new strategy defined), or Shen et al.'s complementary filter method.

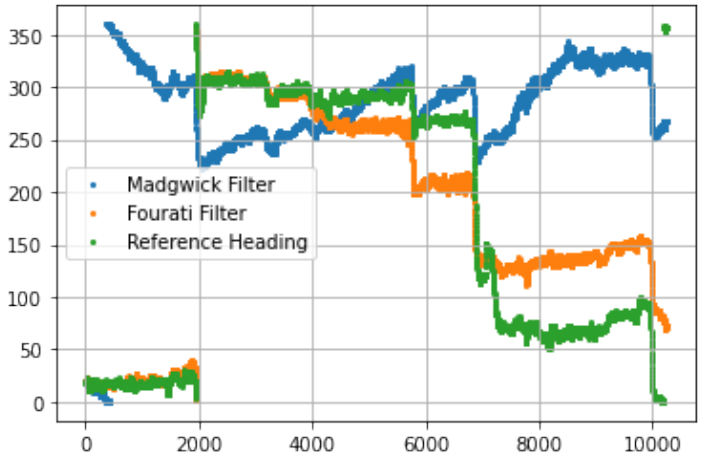


Fig. 6.7: Graph of heading from an outdoor no music test which demonstrates the gyroscopic drift in the Madgwick and Fourati filters

6.5 Displacement Estimation Method

I compared the kinematics method and pedestrian dead reckoning.

		Kinematics Displacement Error / ms^{-1}		Step Length Estimation Displacement Error / ms^{-1}	
		Average	Standard Deviation	Average	Standard Deviation
Magnetometer Only	**Never discarding points**	6.16	3.18	0.437	0.178
	Rolling window of ten points	2.32	0.769	0.133	0.0354
	Rolling window of a minute	5.47	1.3	0.231	0.101
	Discarding every rotation	2.49	1.12	0.251	0.0686
Gyroscope only		2.69	1.11	0.195	0.0290
Madgwick Filter		3.66	1.55	0.217	0.0353
Fourati Filter		2.42	1.14	0.180	0.0265
Complementary Filter		2.63	0.538	0.245	0.0721

Table 6.6: Results comparing displacement error for different displacement estimation methods

Table 6.6 shows that PDR significantly outperforms the kinematics-based method. By looking at a path generated by the method (Figure 6.8), we can identify that the kinematics method exhibits a significant velocity error that builds up over time.

Figure 6.9 shows that the method observed test subjects reaching $4ms^{-1}$ ($\approx 12km/h$) which is far higher than could be true. It is possible this could be fixed with further calibration.

6.6 Test Environment

Firstly, I looked at the impact of audio playback on the results (Table 6.7). I found there was no statistically significant change in the heading errors (p=0.29). Interestingly, whilst there was no significant difference in displacements for any of the magnetometer methods, there was one for the gyroscopic methods. This was highly unexpected, since the magnetic noise of music and speakers should only affect magnetometer methods. Therefore, I hypothesise that the results arose from user actions when listening to music. There was likely more noise, with users looking to the sides more. This is a possible issue with the tracking method. The user's head must be facing in the direction of motion for tracking to work. This does not mean that the user cannot look away, as they can move their eyes without moving their head. However, when looking around, users generally move their heads a little as well.

The results from the outdoor tests showed the algorithms having a lower heading error, but a slightly higher

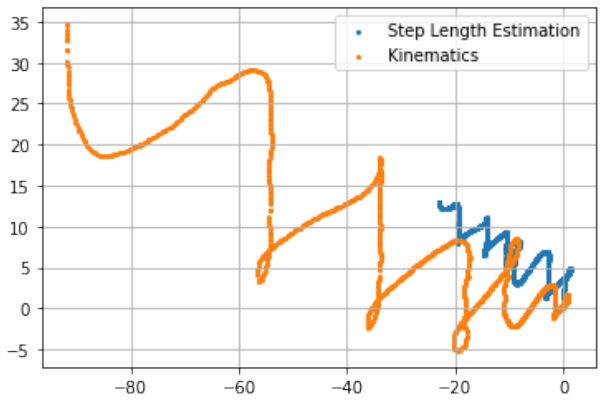


Fig. 6.8: Path comparison between kinematics method and pedestrian dead reckoning for indoor no music test

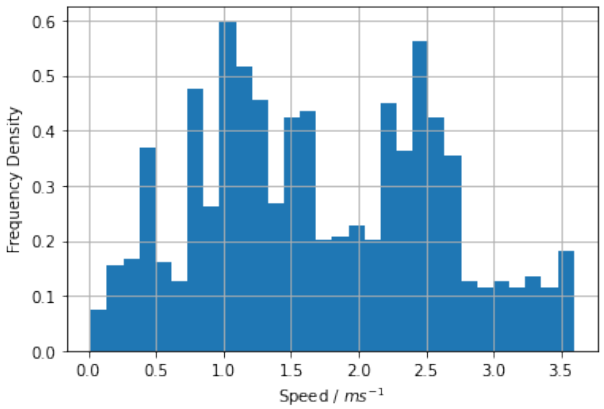


Fig. 6.9: Histograms demonstrating the speed of movement in kinematics method

		Inside No Music Heading Error / °		Displacement Error / ms^{-1}		Inside Music Heading Error / °		Displacement Error / ms^{-1}	
		Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Magnetometer Only	Never discarding points	24.3	19.8	0.437	0.1790	24.8	19.0	0.371	0.1770
	Rolling window of ten points	15.1	15.6	0.133	0.0354	17.1	18.0	0.145	0.0680
	Rolling window of a minute	19.4	18.6	0.231	0.1010	20.0	17.9	0.180	0.0800
	Discarding every rotation	15.9	15.9	0.251	0.0686	17.0	16.8	0.214	0.0757
Gyroscope only		24.5	24.0	0.195	0.0290	21.4	19.0	0.190	0.0703
Madgwick Filter		18.8	14.3	0.217	0.0353	22.4	20.1	0.286	0.0500
Fourati Filter		21.8	15.9	0.180	0.0265	20.5	22.0	0.205	0.0621
Complementary Filter		21.9	17.7	0.245	0.0721	20.7	23.1	0.205	0.0661

		Outside No Music Heading Error / °		Displacement Error / ms^{-1}		Outside Music Heading Error /°		Displacement Error / ms^{-1}	
		Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Magnetometer Only	Never discarding points	9.6	7.0	0.29	0.0900	9.93	9.36	0.283	0.0672
	Rolling window of ten points	15.1	17.2	0.172	0.0505	13.6	11.4	0.192	0.0577
	Rolling window of a minute	13.0	11.8	0.216	0.110	12.6	13.0	0.208	0.0602
	Discarding every rotation	17.6	16.9	0.231	0.118	10.9	9.48	0.292	0.114
Gyroscope only		37.6	19.4	0.44	0.154	36.9	18.1	0.528	0.149
Madgwick Filter		29.7	17.9	0.49	0.168	22.4	15.8	0.438	0.111
Fourati Filter		20.2	11.6	0.48	0.106	20.7	13.6	0.382	0.103
Complementary Filter		17.9	12.8	0.30	0.108	18.9	15.1	0.333	0.0960

Table 6.7: Results comparing heading error and displacement error for different environments and noise profiles

displacement error. The lower heading error came from the simpler testing situation. Whilst every ≈10 seconds there was a 180° turn in the indoor test, this did not exist in the outdoor test. This led to far fewer failed magnetometer calibrations. For the higher displacement error, I suspect that this came from a higher accelerometer noise that impacted the accuracy of step detection.

Outside, there was not a clear correlation between heading estimation accuracy and displacement error. For example, the no-splitting magnetometer-only heading method has the lowest heading error. This did not correspond to the lowest displacement error. This suggests that there was a level of over-calibration, where the iPhone did not necessarily exactly point in the direction of motion, whilst the earables did. The methods

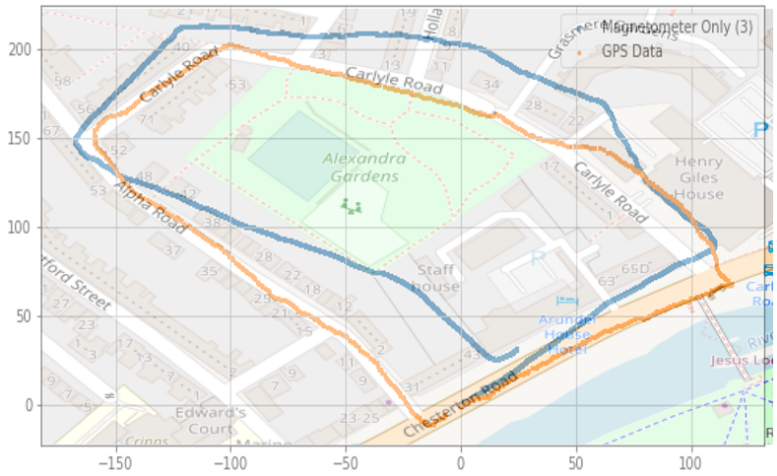


Fig. 6.10: Example of outdoor tracking path

with occasional discarding have a reduced over-calibration effect.

For gyroscopic methods, the length of the test showed the issues with gyroscopic drift. Sometimes, the drift was not too significant (Figure 6.12), whilst sometimes it was particularly bad (Figure 6.11). It appeared that the gyroscope drift may be user dependent and could be fixed by a per-user calibration. Using a longer stationary clip, it might be possible to characterise this per-user drift and correct against it whilst walking. As seen in Figure 6.8, the Fourati filter dealt with the gyroscopic drift far better than the Madgwick filter. Shen et al.'s method dealt with this drift effectively by calibrating the gyroscope with the magnetometer.

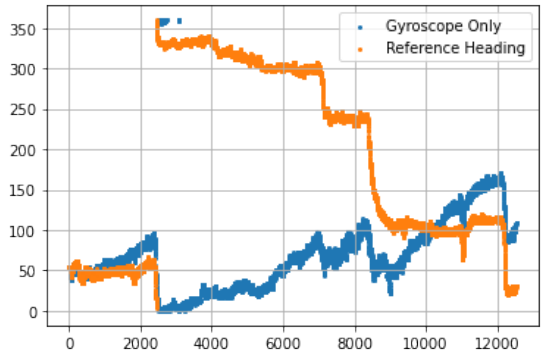


Fig. 6.11: An example of significant gyroscopic drift in an outdoor test

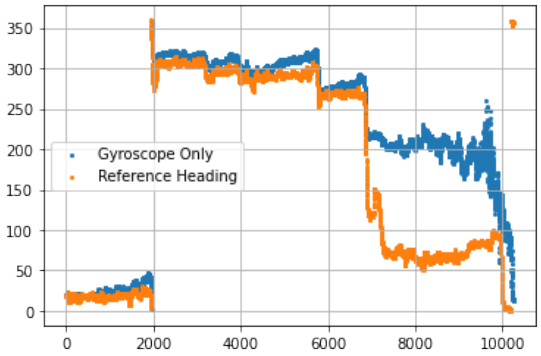


Fig. 6.12: An example of minimal gyroscopic drift in an outdoor test

6.7 Impact of Particle and Kalman Filters

		Original Heading Error / °		Original Displacement Error / ms^{-1}		Particle Filter Heading Error / °		Particle Filter Displacement Error / ms^{-1}		Kalman Filter Heading Error / °		Kalman Filter Displacement Error / ms^{-1}	
		Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Magnetometer Only	Never discarding points	24.3	19.8	0.437	0.178	25.6	20.7	0.428	0.183	25.6	20.7	0.427	0.184
	Rolling window of ten points	15.1	15.7	0.133	0.0354	16.3	16.9	0.157	0.083	16.3	16.9	0.158	0.083
	Rolling window of a minute	19.4	18.6	0.231	0.101	20.8	18.5	0.238	0.101	20.8	18.5	0.25	0.01
	Discarding every rotation	15.9	15.9	0.251	0.0686	15.8	15.5	0.148	0.0603	15.8	15.5	0.154	0.0615
Gyroscope only		24.5	24.0	0.195	0.0290	25.9	21.7	0.194	0.0202	25.9	21.7	0.194	0.0217
Madgwick Filter		18.8	14.3	0.217	0.0353	19.5	15.4	0.226	0.0438	19.5	15.4	0.228	0.0439
Fourati Filter		21.8	15.9	0.180	0.0265	21.9	15.9	0.179	0.0287	21.9	15.9	0.18	0.0286
Complementary Filter		21.9	17.7	0.245	0.0721	18.9	14.6	0.17	0.0396	18.9	14.7	0.169	0.0391

Table 6.8: Particle and Kalman filter results for heading error and displacement error

Table 6.8 shows that there was no gain made by the particle or Kalman filter over assuming both measurements were equally accurate. This suggests that the inaccuracy in tracking arises from the calculated heading

and displacement, not their fusion. They did however, produce an interesting measure of the uncertainty of the position at any given point (Figure 6.13). This measurement of uncertainty could be used to help to identify when to collect a GPS ground truth position measurement. It could also be fed into a calibration system, using a 'more certain' measurement to calibrate a 'less certain' measurement for example.

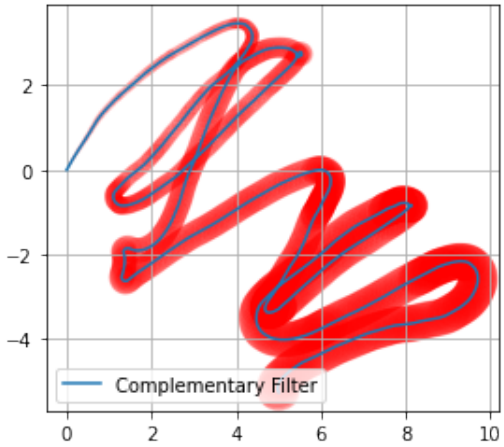


Fig. 6.13: An example of uncertainty visualised through a particle filter

6.8 Evaluating Online Inertial Navigation

I evaluated the online inertial navigation in two ways. Firstly, I looked at the accuracy of the system for the indoor tests and then looked at the systems aspect, looking at the CPU and memory usage on the iPhone.

6.8.1 Accuracy

To evaluate the accuracy of the online solution which was written after the data collection, I wrote code to allow the iPhone application to process previously collected data in simulated real time.

	Pedestrian Dead Reckoning Final Displacement Error / ms^{-1}	
	Offline (Jupyter Notebook)	**Online (iPhone)**
Magnetometer only	0.133	0.181
Gyroscope only	0.251	0.243
Madgwick Filter	0.217	0.356
Complementary Filter	0.187	0.213

Table 6.9: iPhone Accuracy vs Jupyter Notebooks for INS

Table 6.9 demonstrates that the system performed 18% worse than the equivalent offline system. The majority of this comes from the Madgwick filter, which likely has an implementation error. If I exclude the Madgwick filter, the difference was 12%. The results could likely be improved with more time spent tuning the many empirical thresholds in the application.

6.8.2 System Performance

To test the system performance, I ran the iPhone application, completing inertial navigation with two earables producing real-life data. Using the Xcode debugging panel, I took a CPU and memory usage measurement every minute for ten minutes of testing. During this process, the CPU usage stayed constant at 27%. The memory usage started at 41.7MB, then climbing 1MB every 190 seconds. This was described as 'low' battery impact according to Xcode. I compared this to the usage when running Google Maps, which used 57% of CPU usage and 230MB of memory, described as 'high' battery impact. This means that we can track the position of a user with inertial navigation for a day without consuming excess phone battery on the phone or draining the earables batteries.

6.9 Comparison with iPhone tracking

Finally, I compares the results of tracking between earables and an iPhone. Table 6.10 show how tracking on a smartphone is more challenging. The heading accuracy was very low (with the best heading estimation algorithm achieving an error just under 30°). This arose from significantly more noise for the magnetometer and gyroscope (Figure 6.15). This explains why the iPhone also includes GPS to find the phone's 'true heading'. Hence, I can conclude that tracking using raw IMU data performed better on an earable than on a smartphone. However, more work should be carried out to investigate the source of the higher iPhone tracking error.

	Heading Error / °	**Displacement Error /** ms^{-1}
Reference Heading	0	0.0439
Magnetometer Only: Rolling window of ten points	29.5	0.205
Gyroscope Only	41.7	0.45
Madgwick Filter	32.7	0.309
Fourati Filter	31.7	0.401

Table 6.10: Results for tracking algorithms on iPhone raw IMU data

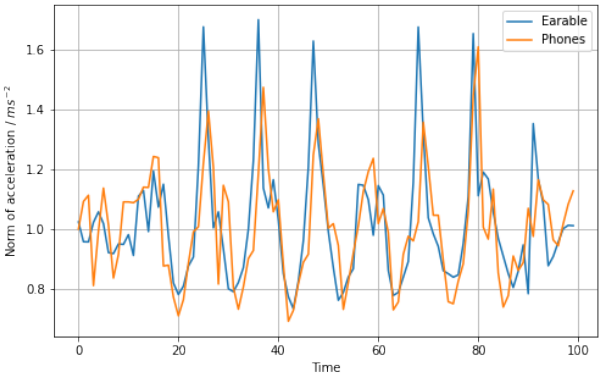


Fig. 6.14: Comparison of accelerometer readings of phone and earable

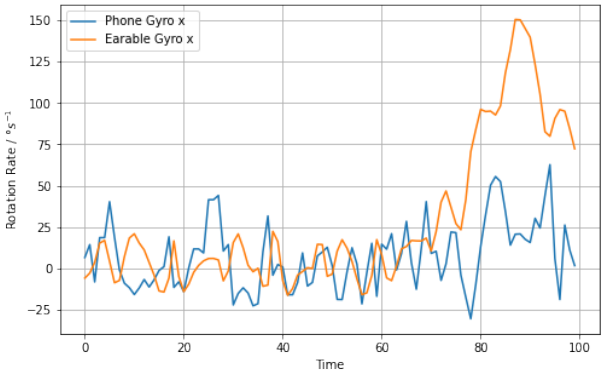


Fig. 6.15: Comparison between gyro readings of phone and earable

6.10 Initial Conclusions

In this chapter, I defined a best method for tracking a user from a single earable. I showed that this tracking performed much better than tracking on an iPhone, whilst consuming minimal iPhone CPU time, 53% less than GPS navigation. The best tracking firstly consists of a novel earable IMU calibration framework consisting of using:

- **Accelerometer Calibration**: Sipos et al. and offset calibration performed the same so either could be used.
- **Gyroscope Calibration**: Shen et al. calibration should be used.
- **Magnetometer Calibration**: Ferlini et al. calibration with the following method to choose when to collect and discard points: Reference points should be collected if the user is stationary. Points should be discarded using either a rolling window of ten points or when a full rotation has occurred.

For heading estimation, magnetometer-only methods performed best but were not the most reliable. Therefore, a method consisting of a complimentary filter of both a magnetometer heading and Madgwick filter would perform the best. Finally, for displacement estimation, pedestrian dead reckoning should be used.

These methods were used to produce the first end-to-end tracking system for an earable, with a heading error of under 10° for an outdoor test and tracking drift of under $0.15ms^{-1}$. In the next chapter, we propose novel methods to combine data from both earables to further reduce the tracking drift and power usage of the complete system.

Chapter 7

Experiments with Sensor Fusion

Throughout this dissertation, I have applied existing tracking algorithms to a prototype earable. In this chapter, I consider one of the major advantages of earables - two independent sets of measurements.

I propose a number of novel methods to combine all data to maximise the accuracy of tracking. I find the best strategy to combine the headings of each earable using a particle filter. This leads to an average reduction in drift of 27%.

I also devise methods to minimise current draw, whilst maintaining the accuracy improvement of fusing the earables. The first uses one IMU component from one earable and two from the other. This does not yield particularly good results. The second suggests mixing the frequency of updates from each earable. I demonstrate that it it possible to drop the frequency of one earable to 5Hz for indoor tests whilst still offering an improvement over using a single earable.

Finally, I theorise a new method of accurate (<15m error) tracking on a smartphone utilising earables to reduce the number of required GPS updates. This yields a 65% reduction in power usage on the iPhone.

7.1 Experiment 1: Combining all data from two earables

In this experiment, I attempted to maximise the accuracy of the results by using all data from both earables. I considered a number of methods to do this:

- **A**: For each data point, average the raw IMU data coming from each earbud.
- **B**: Use each data point from either earable as if from the same earable. This doubles the data rate from 20Hz to 40Hz.
- **C**: For each earable, independently calibrate and find the heading. Then average the heading of the two earables.
- **D**: For each earable, independently calibrate and find the heading. Then use a particle filter to fuse the headings.
- **E**: For each earable, independently find the predicted position. Then average these positions.
- **F**: For each earable, independently find the predicted position. Then use a particle filter to fuse these positions.

As opposed to Chapter 6, where I tested with a large number of possible algorithms, I chose to test the system on the best of the heading and displacement estimation methods defined in Section 6.10.

Table 7.1 and Figure 7.1 show the results for Experiment 1. I first noted that the right earable performs better than the left earable. This suggested that the calibration clip may be of a higher quality for this earable. The left earable used the larger removable Arduino board design (Figure 3.3), whilst the right used the more compact design. It is possible that the compact design reduced random rotation and hence performed better. In the future, if a similar discrepancy was noticed, I could weight the contribution of each earbud.

For **A**, I noted that for the indoor setting test, this method performs as well as using individual earables. However, it performed significantly worse for the outside tests. I can likely put this down to greater

	Inside with Music Final Displacement Error / ms^{-1}		Inside without Music Final Displacement Error / ms^{-1}		Outside with Music Final Displacement Error / ms^{-1}		Outside without Music Final Displacement Error / ms^{-1}	
	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Left Earable	0.180	0.129	0.173	0.096	0.199	0.044	0.226	**0.025**
Right Earable	0.182	0.152	0.135	0.081	0.180	0.144	0.139	0.082
Reference Heading	0.069	0.041	0.077	0.038	0.173	0.136	0.168	0.061
Experiment 1a	0.166	0.109	0.144	**0.087**	0.373	0.120	0.367	0.068
Experiment 1b	0.270	0.248	0.249	0.178	0.383	0.158	0.416	0.142
Experiment 1c	0.160	0.159	0.135	0.095	0.143	**0.074**	0.129	0.033
Experiment 1d	**0.118**	**0.105**	**0.119**	0.089	**0.113**	0.076	0.106	0.039
Experiment 1e	0.178	0.145	0.144	0.089	0.115	0.094	0.103	0.048
Experiment 1f	0.173	0.141	0.156	0.098	0.114	0.092	**0.094**	0.046

Table 7.1: Results for experiment 1

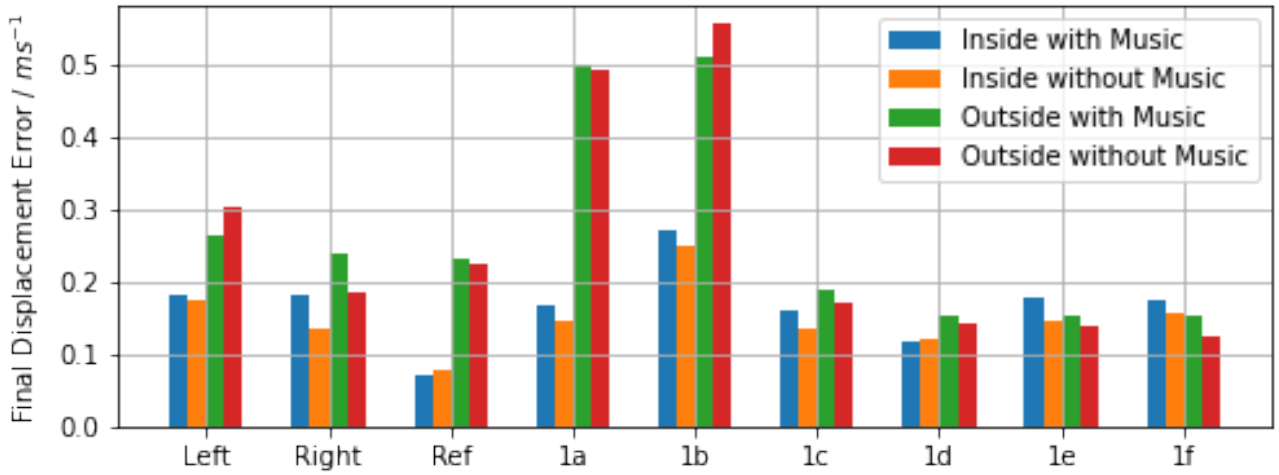


Fig. 7.1: Graph showing final displacement error for each experiment.

background noise, which could affect each device at slightly different times.

For all environments, using all data (**B**) performed worse than using individual earables. I hypothesise that this is based on an issue of BLE timing. Packets are handled in spurts from each device and are labelled with a timestamp on handling. It would be better to instead send a dispatch timestamp using the Arduino's crystal oscillators.

Averaging the heading (**C and D**) performed better than using each earable separately (by 28%). Using a particle filter to fuse headings performed even better. It was statistically significantly better than all methods inside and as good as position fusing methods outside.

Finally, averaging the position (**E and F**) performed much better than individual earables. Using a particle filter did not significantly change the results.

I can conclude that the best all-around performing method is using a particle filter to fuse headings from the two earables after calibrating and finding the headings independently.

7.2 Experiment 2: Combining IMU components from each earable

In this experiment, I considered using one IMU component from one earable and the two other components from the other earable. This is motivated by the idea of reducing the current draw without significantly hampering the tracking accuracy. Initially, I noted the results from Section 3.4 that say that using parts of the IMU do not reduce the current draw.

However, a full design that leverages this concept could instead split up the gyroscope, magnetometer, and

accelerometer with separate chips to distribute the current draw. Therefore, I tested pairing up data packets and taking one element of the left earable and combining with two from the right earable and vice-versa.

	Inside with Music Final Displacement Error / ms^{-1}		Inside without Music Final Displacement Error / ms^{-1}		Outside with Music Final Displacement Error / ms^{-1}		Outside without Music Final Displacement Error / ms^{-1}	
	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Left Earable	0.180	0.129	0.173	0.096	0.199	0.044	0.226	0.025
Right Earable	0.182	0.152	0.135	0.081	0.180	0.144	0.139	0.082
Reference Heading	0.069	0.041	0.077	0.038	0.173	0.136	0.168	0.061
Experiment 2a (Acc from one earable, Gyro and Mag from other earable)	0.265	0.232	0.233	0.128	0.447	0.087	0.455	**0.114**
Experiment 2b (Gyro from one earable, Acc and Mag from other earable)	0.303	0.219	0.249	0.126	0.475	**0.086**	0.479	0.128
Experiment 2c (Mag from one earable, Acc and Gyro from other earable)	**0.175**	**0.109**	**0.148**	**0.092**	**0.387**	0.125	**0.394**	0.142

Table 7.2: Results for experiment 2

Table 7.2 demonstrated that using the accelerometer or gyroscope on one earable performed significantly worse for all environments (32%, 39%, 58% and 61%). This makes sense, since for the Madgwick filter, the accelerometer and gyroscope are used closely together. The results for the magnetometer were more complicated. For an indoor environment it performed only as well as a single earable, but not statistically significantly better. Outdoors, where there is more noise, the method performed much worse than a single earable. Therefore, this idea does not merit further testing.

7.3 Experiment 3: Mixing sampling rates of earables

Section 3.4 showed that reducing the data frequencies reduces the current draw of an earable. Therefore, I compared the performance of both earables running at full data frequency (as observed in Experiment 1), with one earable running at full speed and one running slower. This would offer battery life improvements. To track, I used a particle filter to fuse headings from each earable, which Section 7.1 showed performed the best.

I considered the following scenarios:

- **A**: One earable at 20Hz and other at 10Hz
- **B**: One earable at 20Hz and other at 5Hz
- **C**: One earable at 20Hz and other at 2.5Hz

	Inside with Music Final Displacement Error / ms^(-1)		Inside without Music Final Displacement Error / ms^(-1)		Outside with Music Final Displacement Error / ms^(-1)		Outside without Music Final Displacement Error / ms^(-1)	
	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Left Earable	0.180	0.129	0.173	0.096	0.199	0.044	0.226	0.025
Right Earable	0.182	0.152	0.135	0.081	0.180	0.144	0.139	0.082
Reference Heading	0.069	0.041	0.077	0.038	0.173	0.136	0.168	0.061
Experiment 1d	0.118	0.105	0.119	0.089	0.113	0.076	0.106	0.039
Exp 3a	0.131	0.107	0.128	0.094	0.348	0.269	0.227	0.102
Exp 3b	0.168	0.100	0.156	0.081	0.349	0.138	0.204	0.049
Exp 3c	0.159	0.111	0.185	0.119	0.302	0.121	0.242	0.119

Table 7.3: Results for experiment 4.

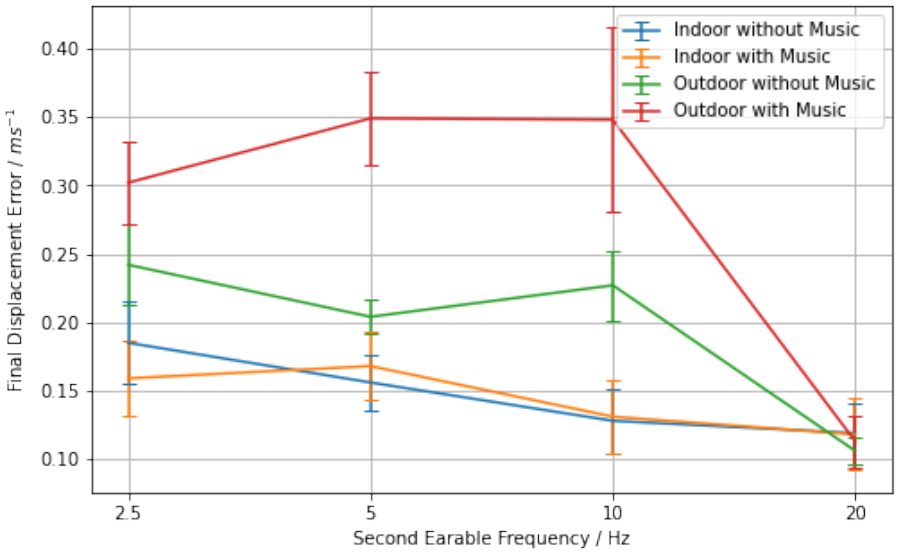


Fig. 7.2: Graph showing final displacement error for experiment 4

The results (Table 7.3 and Figure 7.2) demonstrated that dropping the data frequency of one earable generally adversely impacted the displacement error. For indoor tests, there was a statistically significant difference going from 20Hz to 10Hz and 10Hz to 5Hz, however, there was no difference going from 5Hz to 2.5Hz. Additionally, the results for 10Hz and 5Hz still performed better than one earable individually.

For the outdoor tests, the changes in data frequency had a less obvious effect. There was no clear trend with the reducing the second earable's frequency. However, for all frequency reductions, there was a significant negative impact. I suspect that this arose from an increased background noise that came from the outdoor environment. This noise meant that additional measurements were important for tracking accuracy.

7.4 Experiment 4: Using occasional GPS updates

GPS uses around 40mA, which is higher than the total current draw of the earable prototype. I theorised a method to reduce the power consumption of phone-based tracking by combining IMU data from the earables with GPS from the phone to keep the tracking within a particular accuracy. I implemented this with a particle filter which used GPS updates for particle re-sampling. To re-sample the particles given a GPS measurement, I took the error model of GPS to be a Gaussian distribution with $\sigma = 3.9m$ since the US claim a 95% accuracy of 7.8m [59].

I looked at the required frequency of GPS updates to stay within a particular accuracy (15m). I found that I required a GPS update an average of every thirty seconds to maintain the 15m accuracy outdoors. Using current draw figures from a NEO-6, the ability to return the GPS to idle[1], before using at high performance for one second would reduce power usage by 65%.

[1] when a GPS lock is maintained

Chapter 8

Conclusion

This work has assessed the viability of tracking without GPS on a miniature earable prototype with significant success. The dissertation has considered a number of the challenges of earables that exist today including the lack of earable platforms and noisy data. The former was dealt with by producing a new powerful custom, portable earable platform. The latter was considered by defining a new complete earable IMU calibration framework. The dissertation also considered inertial navigation specific problems, looking at heading estimation and displacement estimation methods. I found that magnetometer heading methods performed best, but were not always reliable, therefore should be fused with gyroscopic headings using a complementary filter. With one earable, I achieved a minimum average heading error of 10° and tracking drift of $0.15ms^{-1}$. The testing was conducted on a new in-the-wild dataset run on six test subjects with different environments and noise profiles.

Following this, I completed research into fusing data from two earables. When maximising tracking accuracy, I reduced the drift by 27% using a particle filter to fuse individual headings. I also proposed a technique to complete accurate (<15m error) smartphone tracking using earables with a 65% power reduction compared to continuous GPS tracking.

8.1 Future Work

This dissertation, whilst expansive, has significant potential for future research:

1. **Construction of a new device**: There are several aspects of the device which can be revisited:
   (a) The size of the device can be reduced using a smaller battery and further miniaturised circuitry assembled using a pick-and-place machine.
   (b) The mechanics can be redesigned to be more comfortable for more users and use better materials such as TPU.
   (c) A full audio driver chip can be included to allow for full audio output through the headphones, rather than just tones.
2. **INS algorithms**: Through this dissertation, I considered a large number of INS algorithms. There are however, even more to look at. In particular, with the larger new dataset that I collected, it would be possible to consider Chen et al.'s [60] deep learning approach.
3. **Systems research**: Further work could move the tracking to the earables. This would require significant software engineering work to change the communications topology and translate the various algorithms currently on the iPhone to run on the Arduino board.
4. **Further evaluation**: There are a number of other evaluation comparisons that could be made:
   (a) Benchmarking the earable system, comparing our results to other wearable INS methods, such as using a smartwatch.
   (b) Completing a user study and more comprehensive evaluation of online inertial navigation on the iOS application.

Bibliography

[1] Andrea Ferlini, Alessandro Montanari, Andreas Grammenos, Robert Harle, and Cecilia Mascolo. Enabling In-Ear Magnetic Sensing: Automatic and User Transparent Magnetometer Calibration. 2021.

[2] Martin Sipos, Pavel Paces, Jan Rohac, and Petr Novacek. Analyses of triaxial accelerometer calibration algorithms. *IEEE Sensors Journal*, 12(5):1157–1165, 2011.

[3] Sheng Shen, Mahanth Gowda, and Romit Roy Choudhury. Closing the gaps in inertial motion tracking. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, pages 429–444, 2018.

[4] Justin Kavanagh, Rod Barrett, and Steven Morrison. The role of the neck and trunk in facilitating head stability during walking. *Experimental brain research*, 172(4):454, 2006.

[5] Andrea Ferlini, Alessandro Montanari, Cecilia Mascolo, and Robert Harle. Head motion tracking through in-ear wearables. In *Proceedings of the 1st International Workshop on Earable Computing*, pages 8–13, 2019.

[6] Chuanhua Lu, Hideaki Uchiyama, Diego Thomas, Atsushi Shimada, and Rin-ichiro Taniguchi. Indoor positioning system based on chest-mounted imu. *Sensors*, 19(2):420, 2019.

[7] LASEREF Inertial Reference Systems. `https://aerospace.honeywell.com/en/learn/products/sensors/laseref-inertial-reference-systems`. (Accessed on 05/14/2021).

[8] Alberto Serra, Tiziana Dessì, Davide Carboni, Vlad Popescu, and Luigi Atzori. Inertial navigation systems for user-centric indoor applications. *Networked and Electronic Media Summit, Barcelona*, 2, 2010.

[9] Alejandro Correa, Estefania Munoz Diaz, Dina Bousdar Ahmed, Antoni Morell, and Jose Lopez Vicario. Advanced pedestrian positioning system to smartphones and smartwatches. *Sensors*, 16(11):1903, 2016.

[10] Darrell Loh, Shaghayegh Zihajehzadeh, Reynald Hoskinson, Hamid Abdollahi, and Edward J Park. Pedestrian dead reckoning with smartglasses and smartwatch. *IEEE Sensors Journal*, 16(22):8132–8141, 2016.

[11] P Mihajlik, M Guttermuth, K Seres, and P Tatai. DSP-based ultrasonic navigation aid for the blind. In *IMTC 2001. Proceedings of the 18th IEEE Instrumentation and Measurement Technology Conference. Rediscovering Measurement in the Age of Informatics (Cat. No. 01CH 37188)*, volume 3, pages 1535–1540. IEEE, 2001.

[12] Alexander Fiannaca, Ilias Apostolopoulous, and Eelke Folmer. Headlock: A wearable navigation aid that helps blind cane users traverse large open spaces. In *Proceedings of the 16th international ACM SIGACCESS conference on Computers & accessibility*, pages 19–26, 2014.

[13] Young Hoon Lee and Gerard Medioni. Wearable RGBD indoor navigation system for the blind. In *European Conference on Computer Vision*, pages 493–508. Springer, 2014.

[14] Fahim Kawsar, Chulhong Min, Akhil Mathur, Alessandro Montanari, Utku Günay Acer, and Marc Van den Broeck. eSense: Open Earable Platform for Human Sensing. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*, pages 371–372, 2018.

[15] Hiroki Ota, Minghan Chao, Yuji Gao, Eric Wu, Li-Chia Tai, Kevin Chen, Yasutomo Matsuoka, Kosuke Iwai, Hossain M Fahad, Wei Gao, et al. 3d printed "earable" smart devices for real-time detection of core body temperature. *ACS sensors*, 2(7):990–997, 2017.

[16] Nhat Pham, Taeho Kim, Frederick M Thayer, Anh Nguyen, and Tam Vu. Earable–An Ear-Worn Biosignal Sensing Platform for Cognitive State Monitoring and Human-Computer Interaction. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, pages 685–686, 2019.

[17] Robert P Leland. Mechanical-thermal noise in vibrational gyroscopes. In *Proceedings of the 2001 American Control Conference.(Cat. No. 01CH37148)*, volume 4, pages 3256–3261. IEEE, 2001.

[18] Oliver J Woodman. An introduction to inertial navigation. Technical report, University of Cambridge, Computer Laboratory, 2007.

[19] Kirill Shcheglov, Christopher Evans, Roman Gutierrez, and Tony K Tang. Temperature dependent characteristics of the JPL silicon MEMS gyroscope. In *2000 IEEE Aerospace Conference. Proceedings (Cat. No. 00TH8484)*, volume 1, pages 403–411. IEEE, 2000.

[20] Grantham Pang and Hugh Liu. Evaluation of a low-cost MEMS accelerometer for distance measurement. *Journal of Intelligent and Robotic Systems*, 30(3):249–265, 2001.

[21] Iuri Frosio, Federico Pedersini, and N Alberto Borghese. Autocalibration of triaxial MEMS accelerometers with automatic sensor model selection. *IEEE Sensors Journal*, 12(6):2100–2108, 2012.

[22] Jorge J Moré and Danny C Sorensen. Newton's method. Technical report, Argonne National Lab., IL (USA), 1982.

[23] Nobuo Yamashita and Masao Fukushima. On the rate of convergence of the Levenberg-Marquardt method. In *Topics in numerical analysis*, pages 239–249. Springer, 2001.

[24] 44 Smartphone Addiction Statistics for 2021. `https://www.slicktext.com/blog/2019/10/smartphone-addiction-statistics/`. (Accessed on 05/14/2021).

[25] Using LSM303DLH for a tilt compensated electronic compass. `https://www.pololu.com/file/0J434/LSM303DLH-compass-app-note.pdf`. (Accessed on 05/14/2021).

[26] Haotian Yang, Bin Zhou, Lixin Wang, Haifeng Xing, and Rong Zhang. A novel tri-axial MEMS gyroscope calibration method over a full temperature range. *Sensors*, 18(9):3004, 2018.

[27] Attitude from angular rate — AHRS documentation. `https://ahrs.readthedocs.io/en/latest/filters/angular.html`. (Accessed on 05/24/2021).

[28] Joan Sola. Quaternion kinematics for the error-state kalman filter. 2017.

[29] Sebastian Madgwick. An efficient orientation filter for inertial and inertial/magnetic sensor arrays. *Report x-io and University of Bristol (UK)*, 25:113–118, 2010.

[30] Robert Mahony, Tarek Hamel, and J-M Pflimlin. Complementary filter design on the special orthogonal group SO (3). In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 1477–1484. IEEE, 2005.

[31] Hassen Fourati. Heterogeneous data fusion algorithm for pedestrian navigation via foot-mounted inertial measurement unit and complementary filter. *IEEE Transactions on Instrumentation and Measurement*, 64(1):221–229, 2014.

[32] Limin Xu, Zhi Xiong, Jianye Liu, Zhengchun Wang, and Yiming Ding. A novel pedestrian dead reckoning algorithm for multi-mode recognition based on smartphones. *Remote Sensing*, 11(3):294, 2019.

[33] Antonio R Jimenez, Fernando Seco, Carlos Prieto, and Jorge Guevara. A comparison of pedestrian dead-reckoning algorithms using a low-cost MEMS IMU. In *2009 IEEE International Symposium on Intelligent Signal Processing*, pages 37–42. IEEE, 2009.

[34] Harvey Weinberg. Using the ADXL202 in pedometer and personal navigation applications. *Analog Devices AN-602 application note*, 2(2):1–6, 2002.

[35] Ngoc-Huynh Ho, Phuc Huu Truong, and Gu-Min Jeong. Step-Detection and Adaptive Step-Length Estimation for Pedestrian Dead-Reckoning at Various Walking Speeds Using a Smartphone. *Sensors*, 16(9), 2016.

[36] Nicolò Strozzi, Federico Parisi, and Gianluigi Ferrari. A Novel Step Detection and Step Length Estimation Algorithm for Hand-held Smartphones. In *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–7, 2018.

[37] Stepping Science: Estimating Someone's Height from Their Walk - Scientific American. `https://www.scientificamerican.com/article/bring-science-home-estimating-height-walk/`. (Accessed on 05/14/2021).

[38] Fredrik Gustafsson. Particle filter theory and practice with positioning applications. *IEEE Aerospace and Electronic Systems Magazine*, 25(7):53–82, 2010.

[39] Greg Welch, Gary Bishop, et al. An introduction to the Kalman filter. 1995.

[40] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of Markov Chain Monte Carlo*. CRC press, 2011.

[41] Eugene Ungar and Kenneth Stroud. A new approach to defining human touch temperature standards. In *40th International Conference on Environmental Systems*, page 6310, 2010.

[42] AirPods Pro - Technical Specifications - Apple. `https://www.apple.com/uk/airpods-pro/specs/`. (Accessed on 05/16/2021).

[43] Anki Reddy Mule, Bhaskar Dudem, Harishkumarreddy Patnam, Sontyana Adonijah Graham, and Jae Su Yu. Wearable single-electrode-mode triboelectric nanogenerator via conductive polymer-coated textiles for self-power electronics. *ACS Sustainable Chemistry & Engineering*, 7(19):16450–16458, 2019.

[44] Hongfei Li, Cuiping Han, Yan Huang, Yang Huang, Minshen Zhu, Zengxia Pei, Qi Xue, Zifeng Wang, Zhuoxin Liu, Zijie Tang, et al. An extremely safe and wearable solid-state zinc ion battery based on a hierarchical structured polymer electrolyte. *Energy & Environmental Science*, 11(4):941–951, 2018.

[45] Jing Ren, Ye Zhang, Wenyu Bai, Xuli Chen, Zhitao Zhang, Xin Fang, Wei Weng, Yonggang Wang, and Huisheng Peng. Elastic and wearable wire-shaped lithium-ion battery with high electrochemical performance. *Angewandte Chemie*, 126(30):7998–8003, 2014.

[46] Marko Kos and Iztok Kramberger. A wearable device and system for movement and biometric data acquisition for sports applications. *IEEE Access*, 5:6411–6420, 2017.

[47] Carlos Gonçalves, Alexandre Ferreira da Silva, João Gomes, and Ricardo Simoes. Wearable e-textile technologies: A review on sensors, actuators and control elements. *Inventions*, 3(1):14, 2018.

[48] EAGLE - Electrical Schematic and Electronic PCB Design Software. `https://www.autodesk.co.uk/products/eagle`. (Accessed on 05/16/2021).

[49] Fusion 360 — 3D CAD, CAM, CAE & PCB Cloud-Based Software — Autodesk. `https://www.autodesk.com/products/fusion-360/`. (Accessed on 05/16/2021).

[50] Arduino_LSM9DS1: LSM9DS1 Library for Arduino. `https://github.com/arduino-libraries/Arduino_LSM9DS1`. (Accessed on 05/16/2021).

[51] Jonathan Ingersoll Bowditch. *American practical navigator*. Number 9. US Government Printing Office, 1903.

[52] Ahrs · pypi. `https://pypi.org/project/AHRS/`. (Accessed on 05/15/2021).

[53] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald,

Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[54] Pykalman: Kalman filter, smoother, and em algorithm for python. `https://github.com/pykalman/pykalman`. (Accessed on 05/15/2021).

[55] Dempsey Chang, Laurence Dooley, and Juhani E Tuovinen. Gestalt theory in visual screen design. In *7th World Conf. Comp. Edu., 2002*, 2002.

[56] Function which returns the least-squares solution to a linear matrix equation - Stack Overflow. `https://stackoverflow.com/questions/37836311/function-which-returns-the-least-squares-solution-to-a-linear-matrix-equation`. (Accessed on 05/15/2021).

[57] Dario Bernardi. A feasibility study on pairinga smartwatch and a mobile device through multi-modal gestures, 2019.

[58] Peak signal detection in realtime timeseries data - Stack Overflow. `https://stackoverflow.com/questions/22583391/peak-signal-detection-in-realtime-timeseries-data`. (Accessed on 05/14/2021).

[59] Brent A Renfro, Miquela Stein, Nicholas Boeker, and Audric Terry. An analysis of global positioning system (GPS) standard positioning service (SPS) performance for 2017. *See https://www. gps. gov/systems/gps/performance/2014-GPS-SPS-performance-analysis. pdf*, 2018.

[60] Changhao Chen, Peijun Zhao, Chris Xiaoxuan Lu, Wei Wang, Andrew Markham, and Niki Trigoni. Deep-Learning-Based Pedestrian Inertial Navigation: Methods, Data Set, and On-Device Inference. *IEEE Internet of Things Journal*, 7(5):4431–4441, 2020.

Chapter A

Ethics Committee Application and Consent Form

Title of Study: Designing an end-to-end inertial navigation system using earables (Part III Project)

Dates: 10/03/2021 ==> 31/05/2021

Study Type: Controlled Experiment

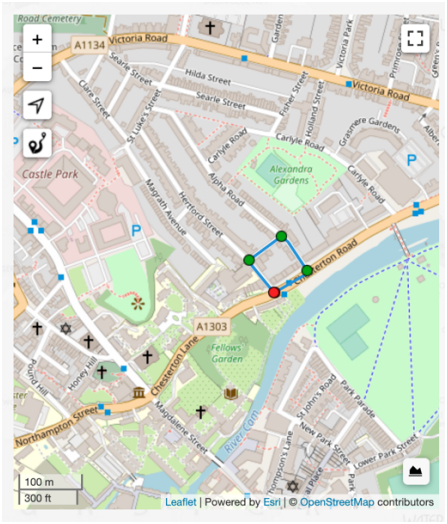
Description

The aim of this research is to explore the feasibility of tracking using smart earbuds which contain an Inertial Measurement Unit (accelerometer, gyroscope and magnetometer) and no GPS. Data will be collected when participants walk around two pre-planned routes wearing logging hardware.

Participants will wear two custom built prototype printed circuit boards (designed and assembled by the investigators). The boards contain an Arduino Nano 33 BLE Sense (https://store.arduino.cc/arduino-nano-33-ble-sense) with an IMU as well as power delivery and management. The components are held within a custom 3D printed casing to hold the prototype to the participant's ear. If required, a headband, beanie hat or adhesive tape will be available to hold the device in place. The prototype has been thoroughly tested, including leaving the device running for six hours with no perceivable change in temperature of the casing. Each participant will wear a prototype on each ear. An image of the prototype to be worn is shown below:

The prototype aims to simulate a smart pair of earhook-style earbuds and are required since no existing earbuds contain a magnetometer. They will also hold an iPhone and wear an Apple Watch (both provided by the investigators). The participant will be required to walk around two routes:

- Up and down the first-floor corridor in 35/37 Chesterton Road five times. This is the primary researcher's house, and all participants will be recruited from this household (same bubble) to ensure minimal COVID-19 risk.
- Around a specific path, shown in the below image.



In both cases, participants will be asked to use the phone naturally. We expect that the phone will mostly be in the participants pockets, with them occasionally taking it out of their pocket to check messages etc. When taking the phone out of their pocket, the participant will be asked to tap a button on an app. For each route, the participant will complete twice, once whilst wearing in-ear headphones (provided by the participant to reduce COVID contamination risk) and listening to music, and once without in-ear headphones. The test will take half an hour, and the following data will be recorded:

- IMU (accelerometer, gyroscope and magnetometer) data from the prototype boards (2x)
- IMU and GPS data from the iPhone as well as timestamps of taps of a button on an app
- IMU data from the Apple Watch

Around 5-10 participants will be recruited, all taken from the primary researcher's household. In addition, due to current legislation, the participants will be asked to conduct the outdoor portion of the test as part of their allowable daily exercise. All participants will be undergraduate or postgraduate students of different ages and genders. Each participant will be briefed about the research study before the tests. Social distancing will be maintained at all times and all items used for the experiment (the prototypes, iPhone and Apple Watch) will be sterilized externally before each experiment. Hand sanitizer will also be provided so that both the participant and researcher can sanitize their hands before and after the experiment. The participants will also be asked to wear a face-mask at all times throughout the data collection.

All information collected in this study is confidential. The participants will be identified with anonymized user labels (e.g. user 1, user 2, etc.). All the data will be saved on a secured server in the Department. Only researchers in the research team will be able to access the collected data initially. We will release the sensing data (without any identifying information) to the research community via online public platforms such as GitHub after participation. The results obtained from the work with this dataset will be disseminated by inclusion in a Part III dissertation, PhD thesis and by publishing in scientific journals and conference proceedings.

Precautions

Participants will be volunteers, each signing a consent form specifying their rights prior to the study (see attached form).

One day prior to the experiments, the participants will be contacted to examine whether they have any COVID symptoms or been in contact with anyone who has. The participant will not be recruited if either one of the answers is 'yes'.

No personal data (like gender and age) will be recorded.

Participants will be asked to wash their hands before and after data collection.

Participants will be asked if they have allergies to any form of tape before the experiment and will be asked again if tape is required to hold the prototype to the participants' ears.

The hardware (prototypes, iPhone and Apple Watch) will be sterilized using cleaning spray and wipes before and after the experiment.

The beanie hat and headband will be machine washed after every use.

At all times, social distancing will be maintained, and face masks will be worn.

Portable batteries used in the project will be checked before each experiment, and the participants will be informed of electrical safety.

Participants will be informed of the heat testing that has been performed and told to stop the test if at any time they feel the device has warmed up.

No sound will be recorded.

Participants will be briefed and made aware of the aims of the study and the fact that they have the right to withdraw at any time. They will be able to find out the results of the study by contacting the researchers.

Participants will not be subject to physical or mental harm or deceived.

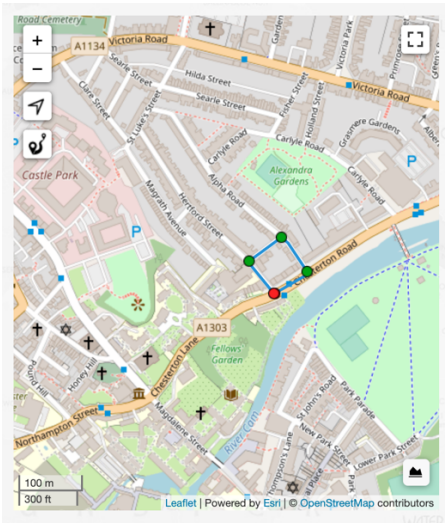
When walking using the prototype, the researcher will walk behind to warn them of any possible danger.

Designing an Inertial Navigation System using Earables Experiment - Consent Form

Experiment Purpose & Procedure

The purpose of this experiment is to determine the error in location and heading tracking using earables when walking around.

The experiment consists of 2 parts. In the first part, you will be asked to walk up and down the first-floor corridor of your house (35/37 Chesterton Road) five times. In the second part, you will be asked to walk around the block nearest the house, going East on Chesterton Road, turning left onto Carlyle Road, following it and then turning left onto Alpha Road, finishing by turning left back onto Chesterton Road. The researcher will walk the route with you, maintaining social distancing at all times. This route is shown below:



You should walk each route twice, once whilst listening to music and once whilst not listening to music. You are requested to provide headphones to listen to music, but sanitized earphones can be required by the researcher on request. For both routes, you will be required to wear a prototype location logging device on both ears and an Apple Watch. You will also be required to use an iPhone naturally. This should involve the phone mostly being in your pocket or bag but perhaps being checked on occasion to emulate responding to a message or checking the news. When taking the phone out of your pocket, a button on the active iPhone app should be tapped. The prototype, Apple Watch and iPhone will all be sterilized before the experiment and hand sanitizer are provided. You should wear a facemask throughout the experiment. The second part of the experiment should be conducted as part of your legally allowable daily exercise.

You should ask any questions to the researcher. Please note that none of the tasks is a test of your personal intelligence or ability. The objective is to test the accuracy of our research systems.

Confidentiality

The following data will be recorded:

- Motion data (accelerometer, gyroscope and magnetometer) from the prototype on each ear
- Motion data (accelerometer, gyroscope and magnetometer) from the Apple Watch
- Motion data (accelerometer, gyroscope, magnetometer and GPS provided location) from the iPhone
- The timestamp of user taps of the screen to indicate them looking at their device

All data will be coded so that your anonymity will be protected in any research papers and presentations that result from this work.

Finding out about result

If interested, you can find out the result of the study by contacting the researcher x, after date 31/05/2021. His phone number is x and his email address is crsid@cam.ac.uk.

(the following section can be torn off, and retained by researcher, with participant keeping above information)

Record of Consent

Your signature below indicates that you have understood the information about the inertial navigation on earables experiment and consent to your participation. The participation is voluntary and you may refuse to complete any part of the experiment and withdraw from the study at any time with no penalty. This does not waive your legal rights. You should have received a copy of the consent form for your own record. If you have further questions related to this research, please contact the researcher.

_____	_____
Participant	Date
_____	_____
Researcher	Date