Submitted for the Degree of B.Sc. In Computer Science, 2018-2019

**Low-cost gait monitoring**

Name: Petar Pilev Supervisor: Marc Roper

Registration Number: 201544399

Except where explicitly stated all the work in this report, including appendices, is my own and was carried out during my final year. It has not been submitted for assessment in any other context.



Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date:25/03/2019

**Abstract**

Nowadays running and walking are preferred ways of exercise for a lot of people. It is very important to do it properly as running or walking with a poor form consistently may lead to injuries.

Gait analysis can reveal important information related to health and well-being such as higher risk of falling or early disease detection in people with abnormal gait. Usually gait analysis is conducted in a lab on a treadmill and with a lot of attached sensors to your body. This not only disrupts normal gait as the conditions are not natural and the number of attached sensors make the person feel uncomfortable but often the costs to conduct such tests are quite high.

The Low-cost gait monitoring device (GTracker) was developed to overcome those problems. The project aims to provide a gait analysis device that can be used outdoors on a daily basis with the help of a mobile application. It also strives to be unobtrusive when mounted and to be as cheap as possible.  
  
Keywords: Gait analysis, Outdoors, Low-cost, Mobile Application

**Acknowledgments**

Firstly, I would like to thank my supervisor Marc Roper for his continuous support and invaluable help he offered throughout the project. The feedback and advice he gave during our meetings guided me and kept me on track with the set goals.  
I would also like to thank all my family members and my significant other for always encouraging me and not letting me give up in times of despair.

Contents

[Introduction 7](#_Toc4396820)

[1.1 Aims and Objectives 7](#_Toc4396821)

[1.2 Outcomes 7](#_Toc4396822)

[1.3 Report Structure 8](#_Toc4396823)

[Background Research and Related Work 9](#_Toc4396824)

[2.1 Gait Cycle and analysis 9](#_Toc4396825)

[2.2 Related work 13](#_Toc4396826)

[Problem description and specification 15](#_Toc4396827)

[3.1 Problem Description 15](#_Toc4396828)

[3.2 Requirements 15](#_Toc4396829)

[3.3 Problem approach 16](#_Toc4396830)

[System Design 17](#_Toc4396831)

[4.1 Design Methodology 17](#_Toc4396832)

[4.2 Communication Method 17](#_Toc4396833)

[4.2.1Bluetooth Classic vs Bluetooth Low Energy 18](#_Toc4396834)

[4.2.2 Power Consumption 19](#_Toc4396835)

[4.2.3 How It Works 19](#_Toc4396836)

[4.3 Hardware Selection 21](#_Toc4396837)

[4.3.1 Microcontroller Board 21](#_Toc4396838)

[4.3.2 Sensors 22](#_Toc4396839)

[4.3.3 Power Supply 23](#_Toc4396840)

[4.4 Phone Platform 24](#_Toc4396841)

[4.5 Task Delegation 24](#_Toc4396842)

[4.5.1 Microcontroller 24](#_Toc4396843)

[4.5.2 Mobile Application 25](#_Toc4396844)

[4.6 Hook-up 25](#_Toc4396845)

[4.7 User Interface 27](#_Toc4396846)

[Detailed Design and Implementation 29](#_Toc4396847)

[5.1 Implementation languages 29](#_Toc4396848)

[5.1.1 C++ 29](#_Toc4396849)

[5.1.2 HTML 29](#_Toc4396850)

[5.1.3 CSS3 29](#_Toc4396851)

[5.1.4 JavaScript 29](#_Toc4396852)

[5.2 Third Party Tools 30](#_Toc4396853)

[5.2.1 Bootstrap 30](#_Toc4396854)

[5.2.2 JQery 30](#_Toc4396855)

[5.2.3 Popper 30](#_Toc4396856)

[5.2.4 Plotly 30](#_Toc4396857)

[5.2.5 JSLint 30](#_Toc4396858)

[5.2.6 Plugins 31](#_Toc4396859)

[5.3 Prototyping 31](#_Toc4396860)

[5.4 I2C Communication Protocol 32](#_Toc4396861)

[5.5 Microcontroller Implementation Details 32](#_Toc4396862)

[5.5.1 Sensor Initialization 32](#_Toc4396863)

[5.5.2 BLE Initialization 33](#_Toc4396864)

[5.5.3 Reading Data 34](#_Toc4396865)

[5.4.4 Sending Data 34](#_Toc4396866)

[5.6 Mobile Application Implementation Details 35](#_Toc4396867)

[5.6.1 Managing Connection 35](#_Toc4396868)

[5.6.2 Initial Contact Detection 36](#_Toc4396869)

[5.6.3 Stride and Swing Estimation 37](#_Toc4396870)

[5.6.4 Live Graph 38](#_Toc4396871)

[5.6.5 Analysis Representation 38](#_Toc4396872)

[5.6.6 Download Analysis Plot 38](#_Toc4396873)

[5.6.7 Storing Results 39](#_Toc4396874)

[5.6.8 Old Results Page 40](#_Toc4396875)

[Verification and Validation 41](#_Toc4396876)

[6.1 Verification 41](#_Toc4396877)

[6.2 Validation 41](#_Toc4396878)

[6.2.1 Testing Overview 41](#_Toc4396879)

[6.2.2 BLE Test 41](#_Toc4396880)

[6.2.3 Application Error Handling 43](#_Toc4396881)

[Results and Evaluation 45](#_Toc4396882)

[7.1 Final System 45](#_Toc4396883)

[7.2 Analysis Algorithms Evaluation 46](#_Toc4396884)

[7.3 Price Comparison 59](#_Toc4396885)

[Summary and Future Work 61](#_Toc4396886)

[8.1 Summary 61](#_Toc4396887)

[8.2 Future Work Opportunities 61](#_Toc4396888)

[8.3 Conclusions 62](#_Toc4396889)

[Appendix A – References 63](#_Toc4396890)

[Appendix B – Maintenance Guide 66](#_Toc4396891)

[Appendix C – Technical Characteristics 68](#_Toc4396892)

[Appendix D – User Guide 71](#_Toc4396893)

[Appendix E – Detailed Evaluation Results 79](#_Toc4396894)

**Chapter 1**

# Introduction

## Aims and Objectives

The aim of this project is to develop a low-cost gait monitoring system that a person can easily mount and go for a run or walk in the park. Furthermore, a mobile application is to be developed so it can connect with the monitoring device gather the data from the walk/run, analyse it and store it for later reference. At the beginning, research was carried out in order to explore different ways of gait analysis using wearable sensors and ways of transmitting data from the portable device to the mobile app. The objectives based on this process and the initial description of the project are:

* Choose a way of gait analysis and buy appropriate sensors
* Choose a platform on which the mobile application will be developed and start implementing
* Look into different ways of wireless communication, establish the most suitable one and begin experimenting with transmitting data between the two devices
* Implement an analysis algorithm and present the results in an understandable fashion

## Outcomes

Over the course of this project, a fully functional system comprised of a mobile application and a hardware device was designed, implemented and tested for both walking and running gait analysis. The hardware device is easy to mount, and the mobile application is simple and straightforward. Although the analysis algorithms do not perform very well sometimes, they mostly showed plausible and promising results. Other than that, the application does its job quite well. Results can be stored or downloaded, it is also designed to preserve the battery of the gait analysis tool. The hardware system itself is designed to be usable in different weather conditions, to be protected from damage by being put in a protective case. Nonetheless, the components can be easily accessed if a problem occurs or if the battery needs charging.

## 1.3 Report Structure

In the following chapters of this report the methods followed during the development of the project will be discussed. The next chapter contains information about the carried-out background research and already developed technologies related to the project. After that, in Chapter 3 the problem is described thoroughly (requirements and plan to follow). Thereafter in Chapter 4, an overview is presented of the overall structure of the project with attention being paid to the hardware setup and mobile application. Chapter 5 provides insight into the implementation languages and libraries being used. Critical features are also mentioned as well as a prototype created during development is covered. In the next chapter the testing procedures used are described and it is checked to what extend the initially set requirements are met. Results and the evaluation of the system and final application are detailed in Chapter 7. In the final chapter reflection of the whole system is made and the possibilities of future development are examined.

**Chapter 2**

# Background Research and Related Work

This chapter of the report outlined the background research carried out. The aim is to provide insight into the process of gait analysis and give a general idea of the direction of the project. Several related works are also covered.

## 2.1 Gait Cycle and analysis

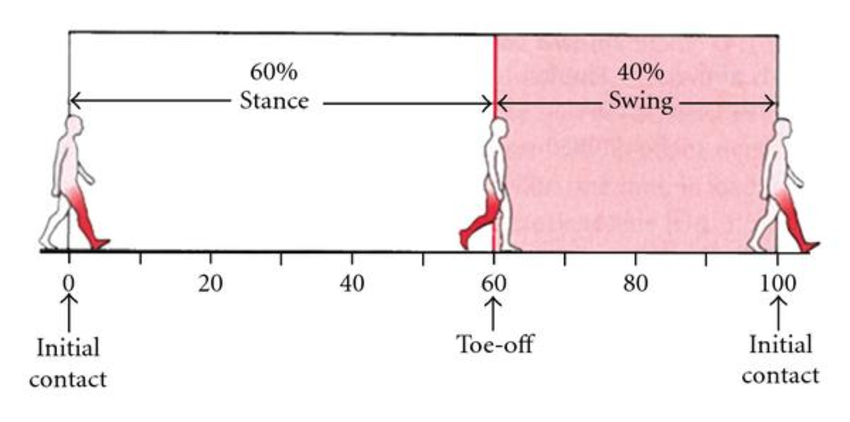
The gait is one of the most repeated daily activity to which humans usually pay little to no attention. However, the activity itself is complex and involves a large amount of different sub-systems each being rich in amount of research and complexity. “The gait can be defined as a movement consisting of a translation of the whole body permitted by a repetition of body segments while keeping the balance” [1]. Therefore, this part of the chapter focuses on explaining in detail the parts of gait cycle and gait analysis relevant to the project. Gait analysis is usually used for clinical and laboratory identification of deviations of normal gait. Detection of abnormal gait can help prevent injuries or spot early stages of diseases like Parkinson’s [2]. In order to conduct a gait analysis and interpret the results the normal gait cycle as well as its spatial and temporal components should be understood. Gait cycle could be described as a period of time between two equivalent events in the gait process. For example, a gait cycle can be explained as a single sequence of functions by one limb. It begins when the reference foot strikes the ground and ends with subsequent floor contact of the same foot (this is called initial contact). Generally, those are the accepted events that mark gait cycle. When observed this cycle of foot strikes can be divided into two distinct phases. A period of stance phase which usually is about 60% of the duration of the cycle and a period of swing phase which consists of the other 40% (Figure 1). Simply put stance phase is when the touches the ground and the swing is when it is in the air. 

Figure 1 The two phases of the human walking gait [3]

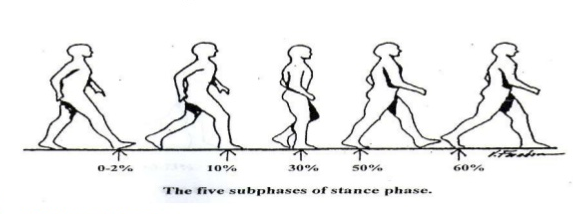
The stance commences with the initial contact of the foot and ends with the toe-off (TO) event and vice versa for the swing. Both phases can be further subdivided. The stance can be segmented into five parts (Figure 2) [4]

Figure 2 Subphases of the stance phase [5]

1. Heel Contact (Initial contact)
2. Loading Response (initial contact of forefoot with ground)
3. Midstance
4. Terminal stance (Heel-off)
5. Preswing (Toe-Off)

On the other hand, the swing phase can be divided into three functional sub-phases occurring in the following sequence. (Figure 3)

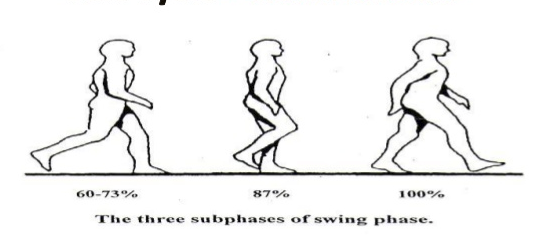


Figure 3 Subphases of the swing phase [5]

1. Initial Swing (Acceleration of the swinging leg)
2. Mid-swing (Swinging limb overtakes the limb in stance)
3. Terminal Swing (Deceleration of the swinging leg)

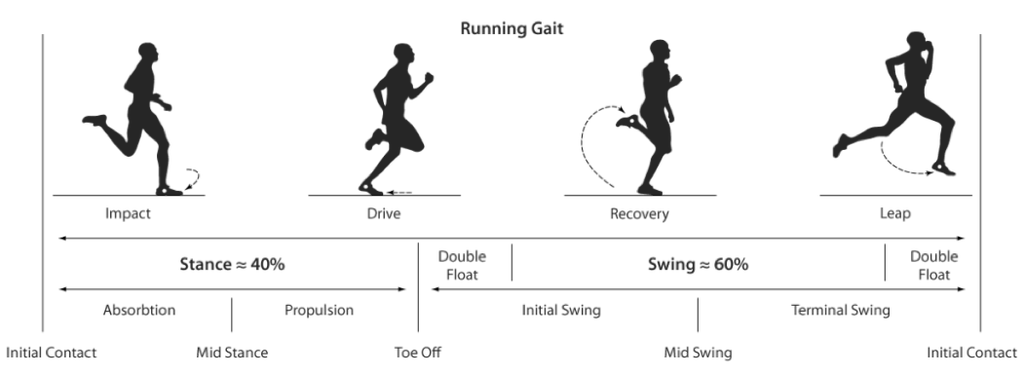
Moreover, the stance phase while walking can also be differently divided into two support phases. Single support is when only one foot is in contact with the floor and double support when both feet are in contact with the ground. They take 40% and 20% of the gait cycle respectively. Another important thing to take into consideration is that with increasing speed of walking the duration of the stance phase and double support decreases while the duration of the swing phase increases. Interestingly, opposed to walking when running the double support phase disappears and double swing (double float) takes its place. Also, the ratio of the stance and swing time reverses. (Figure 4)

Figure 4 Phases and subphases of the running gait [6]

A lot of gait analysis studies use the abovementioned division of gait cycle as a reference. Also, as it can be seen from the figures the graphics are usually normalized to the duration of the gait cycle and are represented as a percentage.  
Now that the gait cycle has been explained we can proceed to delving into gait analysis. As gait itself the analysis can be split into two parts. The first one being kinetic analysis. It studies the forces that cause the motion of the bodies and strives to characterize the forces that act upon the body segments and the body itself. This project mainly focuses on the other type that is Kinematic analysis. Kinematics study the motion of different body parts neither considering the internal nor the external forces that cause the body movement. Kinematics include the measurement of joint angles of lower and upper extremities as well as spatial (distance) and temporal (time) variables. We are mainly concerned with the variables. The spatial parameters (Figure 5) split into:

* Step length - the distance between corresponding successive points of heel contact of opposite feet (in normal gait bot step lengths are equal)
* Stride length - distance between successive points of heel contact of the same foot (double the step length in normal gait)
* Stride width (also known as walking base or base width) – distance between the two feet
* Degree of toe out – represents the angle of foot placement

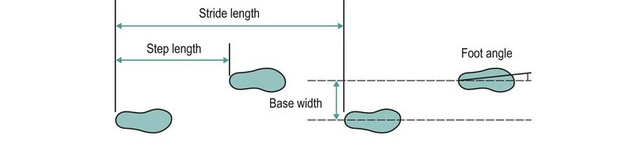


Figure 5 Spatial parameters of the human gait [7]

The temporal variables consist of:

* Step time – amount of time spent between the initial contact of one leg and the initial contact of the contra-lateral leg
* Stride time – amount of time spent between two consecutive initial contacts of the same leg
* Stance time – the amount of time that passes between the initial contact and the toe off event of a leg
* Swing time – the vice versa of the stance time (time between TO and IC)
* Single limb time – when only one leg touches the ground
* Double limb time – the amount of time passed when both legs touch the ground (present only when walking)
* Double swing time – the amount of time both the legs of the person are in the air (present only when running and appearing twice during swing phase of running)
* Cadence – number of steps per unit of time (usually measured in steps/min)
* Velocity – distance covered by the body (usually measured in m/s)

To sum up, after looking into a lot of different material about gait analysis it was concluded that it is bad practice to give a concrete feedback from a gait analysis. Even though there are some established understanding of what a normal gait looks like gait patterns can be diverse from person to person. Additionally, an individual can show different gait patterns by simply increasing or decreasing his/her speed. Therefore, focus was given to research how gait parameters can be measured as accurately as possible.

## 2.2 Related work

There are quite a few developed wearable systems for gait data collection and analysis outside the confines of the traditional laboratories. They make use of different sensors such as accelerometers, gyroscopes, force sensors, pressure sensors etc. Many systems vary in how the sensors are placed on the body. Some embed them in shoes or insoles while others attach them on different body parts such as arms, thighs, shanks or lower back. In this part of the chapter several systems will be examined.

The first one is F-Scan from Tekscan [8]. It provides dynamic pressure, force and timing information for foot function by using an in-shoe pressure sensor. Even though measurements are supposed to be accurate the system has some limitation in terms of analysing the data live. Tekscan provides three connection types for its F-Scan system. Tethered which needs wires to be connected to a computer via USB letting you to read the data live but given the subject only a distance up to 30.5 metres. The second one is wireless, but it is again limiting as the subject needs to be in 100 meters range for data to be transmitted successfully and displayed live. The last one uses a datalogger which is better for outdoor activity, but the data needs to be uploaded on a computer to be analysed and displayed which makes the system harder to use when a person wants to take a walk in the park and review the results right away.

Next is the G-Walk produced by BTS Bioengineering [9]. It uses a wireless inertial sensor attached to the user’s lower back using a specialized belt. The system lets the patient to walk run and jump completely free as specified in their website. The sensor needs to be connected to a computer via Bluetooth to send the data. At the end of each analysis automatic report is being displayed containing spatio-temporal parameters, general kinematic parameters and more. The system uses triaxial accelerometer, gyroscope and magnetometer and a rechargeable via USB battery that works up to 8 hours. It looks easier to use and seems less obtrusive than the F-Scan system as it is completely wireless and uses Bluetooth but still doesn’t appear as if it has the option to be used outdoors without having to stay in 60m range of the computer with which connection is made.

The final system that will be investigated is Smart Insole [10]. To some extend it tries to combine the sensors used in the previous two works. It has a 3-axis accelerometer, 3-axis gyroscope and a 3-axis magnetometer as well as several electronic textile-based pressure sensors embedded into an insole. It looks as a comfortable and inobtrusive system as It uses Bluetooth Low Energy for communication and is wireless. It also uses a rechargeable battery to power the hardware. Furthermore, it presents the opportunity to review the data live and has software developed for mobile and computer which makes it suitable for your everyday walk/run outdoors. The system manages to calculate several temporal parameters such as stride, swing and stance time and cadence as well as several pressure related parameters.

**Chapter 3**

# Problem description and specification

## 3.1 Problem Description

“Running is now a very popular way of exercising, but many people persist in running with poor styles that may lead to injuries or long-term problems. Gait analysis is a service which may be offered by physiotherapists or sports injury clinics but is often conducted in a controlled environment (e.g. on a treadmill) which is very different to a normal running environment. It also typically involves quite expensive kit.   
The aim of this project is to investigate a develop a low-cost gait monitor which makes use of a smartphone and a few cheap wireless motion sensors which a runner can use during their normal runs. The exact choice of the hardware is up to the student, and this project could support more than one student if they chose very different sensors (e.g. hip-mounted accelerometers versus shoe-located pressure sensors). The system should provide them with the ability to download and analyse the gait data for the duration of their run and also investigate what feedback can be given to them during the run (such as warning them when their style is starting to deviate from the ideal and nudging them towards a smoother action for example). ”  
 This is taken from the project suggestion proposed by my supervisor, Marc Roper. From this description initial specifications can be derived which were used as a starting point for determining specifications and material to research.

## 3.2 Requirements

Initial functionality that the system needs to have was derived from the project description. This includes cheap hardware, sensors that do not disrupt the normal way of walking or running, mobile application with ability to analyse, display the data gathered from the sensors and save it. Further requirements were gathered during background research and survey of related work. List of high-level functional and non-functional requirements can be found below:Functional Requirements:

* Gather data from the sensors
* Transmit sensor data from the microcontroller to mobile application
* Analyse the gathered data and display it
* Allow the user to store the analysis and look at it later
* Possibility of live representation

Non-functional requirements:

* Easy to mount the device on your leg
* Not disruptive while walking or running
* The analysis should be as accurate as possible
* The mobile app should be intuitive, easy to use without help and well designed

## 3.3 Problem approach

In the early stages of the project a great deal of background research and looking into related work was needed to explore the possibilities of different sensors that can be used for gait analysis as well as their placement on different parts of the body and method of transmitting data. The initial steps after mostly finishing the abovementioned research were the familiarisation with the hardware and to start gathering data from the sensors by the microcontroller. At this point simultaneous research of the appropriate platform on which the mobile app will be based and how the chosen method of communication Bluetooth Low Energy (BLE) works and how to implement it was conducted. After successful troubleshooting of the BLE connection an initial skeleton app was made on the now chosen platform. The next step was to establish a connection between the mobile app and the device and after successful completion the analysis algorithm had to be started. The development process of the algorithm included a lot of experimenting with different positions marked from earlier research as possible and a prototype was developed. When the algorithm showed plausible output the front-end of the application had to be developed while simultaneously improving the algorithm.  
The approach used when developing the project was based mostly on one or two week long agile sprints as well as prototyping.

**Chapter 4**

# System Design

## 4.1 Design Methodology

The design process of the project was very iterative and incremental, with each successive stage being built on those prior to it. Furthermore, stages were frequently revisited and improved. The main reason for this approach is that mostly the nature of the project is research and experiment driven.   
As large amount of background research was needed for the project. It was decided that the research and development processes will go hand in hand throughout most of the time. Goal based approach was adopted and when a certain goal approached completion work towards the next one was started. Background research was undertaken to determine what is the best sequence of research and development goals. This process led to the work being split in different targets mostly described in the problem approach in the previous chapter.

## 4.2 Communication Method

Bluetooth was chosen to be the way the mounted device and the mobile application will communicate with each other. It is a wireless technology standard for exchanging data between devices over short distances using short-wavelength UHF (Ultra High Frequency) radio waves.

### 4.2.1Bluetooth Classic vs Bluetooth Low Energy

After looking into Bluetooth, it was discovered that Bluetooth Low Energy (which is Bluetooth 4.0) is the better choice for the project [12] [13]. They have a lot of similarities such as that they operate with the same pairing technology, authentication and encryption. Also, both operate in a classic master-slave model. First devices need to be paired and then the data transmission can happen. The important difference is that Bluetooth was initially designed for continuous streaming data applications which means that a large quantity of data can be transmitted. This makes Bluetooth good fit for consumer products such as headphones, speakers etc. On the other hand, BLE is ideally suited for connectivity in products that require only periodic transfer of data. As you can tell from its name Bluetooth Low Energy offers low power consumption of a magnitude of 1-50% of that of the Classic Bluetooth. Because of that the application using BLE can rely on less battery power thus making the size of the used battery smaller. It makes it perfect for a device that needs to be easily mounted. Table 1 illustrates the main differences between the two types of Bluetooth.

Figure 6 BLE overview [11]

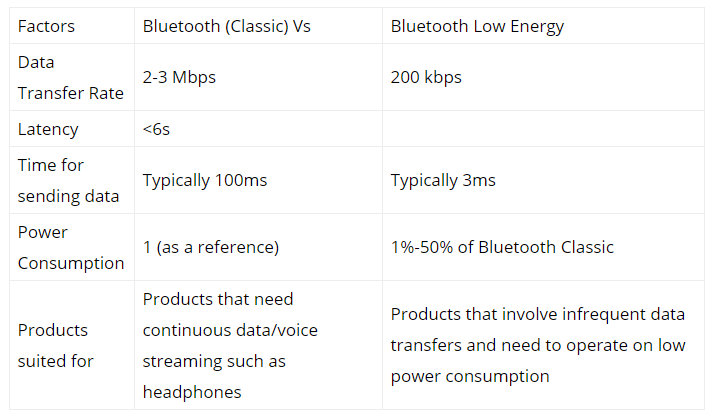


Table Bluetooth vs BLE comparison [13]

### 4.2.2 Power Consumption

The way BLE achieves low power consumption is its operating cycle. Most of the time when it is working it is in sleep mode. It wakes up and starts transferring data only when a connection with another device is initiated.

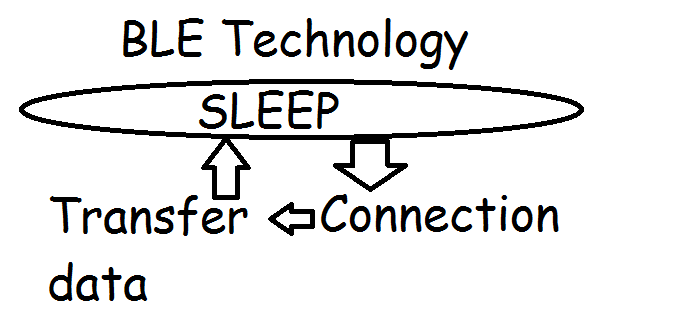


Figure 7 BLE operational cycle [13]

### 4.2.3 How It Works

With BLE there are two types of devices Server and Client. In the case of this project the microcontroller acts as a Server and the mobile phone acts as a Client. The Server advertises its existence so it can be found by other devices and holds the data to be transmitted. The Client scans the nearby devices and connects to the Server that it is looking for. After establishing connection, the Server listens for incoming data. This type of communication is called point-to-point communication. There are two other modes supported from BLE. Broadcast (Server transmits data to several Clients) and Mesh (many-to-many connection). In this project only point-to-point communication is used. Important term in understanding BLE is GATT which stands for Generic Attributes. It defines a hierarchical data structure or put simpler it defines the way two BLE devices send and receive standard messages (Figure 8). The top level of the hierarchy is a Profile which is usually composed from one or more Services. Every Service contains at least one Characteristic and it can also reference other Services. A Characteristic is always owned by a Service. The Characteristic has two mandatory attributes and one optional one. The first one holds properties which contain the meta data about the data and the second one is the value. The optional one is a descriptor which further expands on the meta data from the properties. Each Service, Characteristic and descriptor has a Universal Unique Identifier (UUID) which is a unique 128-bit number used to distinguish them from each other. An overview of GATT and Standard Services, Characteristics and descriptors with shorter UUIDs can be found on the official Bluetooth page [14]. The user can also define his own Services and Characteristic if they are not present in the standard list on the Bluetooth page. Furthermore, Characteristics may have different modes to determine how the data they hold can be manipulated such as read, write, notify etc. For example, a read property allows the data to be read at this exact moment while if the Characteristic has a property notify it sends a notification each time the value inside has been changed. To sum up, when a Client establishes connection it starts reading data by using the UUID to connect to the Service/s and Characteristic/s that it is interested in.

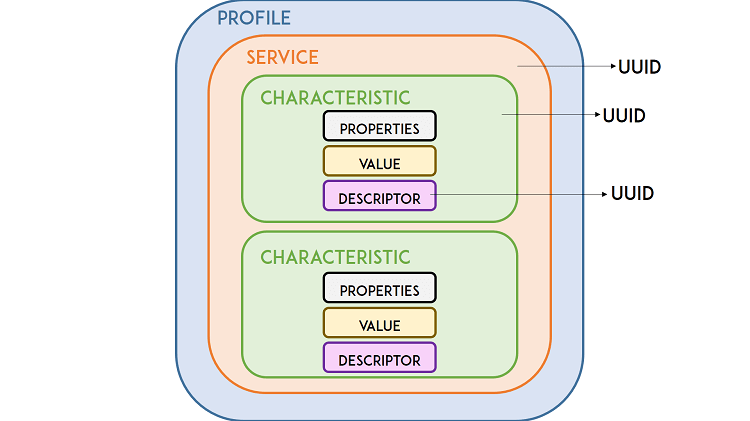


Figure 8 Hierarchical data structure of BLE [15]

## 4.3 Hardware Selection

After establishing the main direction of the project as well as the method of communication, a research was undertaken to find out the suitable hardware components. The needed parts to accomplish the set goals were a microcontroller board, sensor/s and power supply. The extensive list of the components’ technical characteristics can be found in Appendix C

### 4.3.1 Microcontroller Board

When talking about a microcontroller people usually make the mistake by assuming that it refers to the whole board. In reality the microcontroller is only the chip acting as a small computer which is attached to a bigger board which has voltage regulators, resistors, all the pins used for connecting sensors and other parts that are important for the complex electrical circuit to work. Therefore, research was undertaken to choose the appropriate chip first because usually there are quite a few different breakout boards using the same microcontroller chip. When choosing a chip and a board several online guides were examined [16] [17]. The main qualities looked for were that the board is compact, the chip had built in Bluetooth with the capability of using BLE to avoid buying a standalone module that needs to be connected to the board separately and last that it was at an acceptable price. The ESP32 chip was chosen because of its robust design capable of functioning reliably in different environments, its ultra- low power consumption suited for a device that needs to be provided with its own power supply [18]. After careful consideration of the available boards using this chip the Adafruit HUZZAH32 – ESP32 Feather board was chosen. Even though cheaper alternatives were available the main advantages pushing the decision towards this board its small dimensions and yet having all the needed pins for connection with sensors. Extensive documentation and good user feedback further solidified the choice. Also, it has a built in JST jack for a battery making it easy to supply it with power. A Lithium Ion/Polymer charger also allows the battery to be charged while still connected to the board.

### 4.3.2 Sensors

Choosing the correct sensors was yet another crucial part needed for the success of the project. Two types of sensors were considered in this part of the research namely force sensitive resistors (FSR) and motion sensors. Initially FSR sensors were investigated. The idea was to put them in the sole of the shoe and capture different gait events and extract some gait parameters based on them as the sensors allow you to detect physical pressure and weight. Even though the research looked promising as they were simple to use, the idea was discarded . The chosen FSRs that could measure and withstand the pressure and force applied from a walking or running person were too expensive and took too long to arrive as most of them were shipped from overseas. The second idea was to use motion sensors such as accelerometers and gyroscopes on different parts of lower body to capture their movement and register different important events of the gait. Accelerometers are devices that measure proper acceleration (or rate of change of velocity) of a body which is different from acceleration in a fixed coordinate system. The main difference is that accelerometers consider in their measurements the Earth’s gravity. A gyroscope is a device used for measuring or maintaining orientation and angular velocity. There are several different types gyroscopes, but we are interested in Micro-Electro-Mechanical System (MEMS) gyroscopes. They sense angular velocity from the Coriolis force applied to a vibrating element [19]. It was decided that one or two sensors should be bought to make initial tests and experiments before buying more. At this stage different possibilities were looked into and IMU (Inertial Measurement Unit) sensor was discovered. This is an electronic device that measures and reports a body’s specific force, angular rate and sometimes magnetic field surrounding the body using a combination of accelerometer, gyroscope and magnetometer sensors. They have several applications including industry quality control, medical rehabilitation, navigation systems and more [20]. A single IMU sensor was chosen namely the SparkFun MPU-9250 IMU Breakout. It is a 9 DoF (Degrees of freedom) which refers to the different ways that sensor is able to move throughout the 3D space. It contains a 3-axis gyroscope, 3-axis accelerometer and a 3-axis magnetometer. Even though the magnetometer is redundant in the case of this project this sensor was chosen because of the extensive documentation, good user feedback, relatively good price and small size.

### 4.3.3 Power Supply

Because the microcontroller’s JST jack accepts 4.2/3.3V Lithium Polymer (Lipo/Lipoly) or Lithium Ion (Lilon) battery only those types were considered when investigating for an appropriate one. The next step was to roughly estimate the power consumption of the whole system in order to choose the appropriate mAh needed. Looking at the sensor’s electrical characteristics in its datasheet it is stated that the most supply current it needs is 3.7 mA. Also, according to the Adafruit Bluefruit LE UART Friend test [21] of current measurement of Bluetooth Low Energy a 1200mAh battery shows that its expected life is 645 hours with peak current 15.2mA. Even though this hardware component is not part of this project’s system it gives an idea how much power does the BLE consume while streaming data. With the stated currents was estimated that a 500mAh 3.7V Lipo battery would survive around more than 25 hours which is enough for the magnitude of this project. Moreover, this battery is compact and rather cheap which makes it an even better choice.

## 4.4 Phone Platform

Before starting the development process of the mobile application, a suitable platform had to be chosen. The considered options were to either develop an Android native app using the Android Studio or use a hybrid framework [22]. The second option was the preferred one and more specifically the Phonegap framework was selected. The reasoning behind it was that even though Android Studio gives easier access to the phone’s native functions such as vibration or Bluetooth the hybrid framework gives you the opportunity to produce an Android or an IOS native applications with the same base code. Another advantage against other framework and native Android is that Phonegap applications are developed in HTML, CSS and JavaScript which I have used and have knowledge in. With the other options I would have to get familiar with first before developing.

## 4.5 Task Delegation

As it can be seen the project is divided into two parts development wise this part of the report focuses on a high-level explanation of the tasks each part must do.

### 4.5.1 Microcontroller

The tasks assigned to the microcontroller can be split into two parts. Gathering all the data from the sensor and transmit it to the mobile application. The first part can further be split down to initializing the sensor, calibrating it, gathering the raw data and doing calculations with them in order to derive the exact data needed for the analysing algorithm. The next step is to initialize a BLE server create all the Servers and Characteristics with their corresponding UUIDs and make the microcontroller visible for other devices. After all the above-mentioned tasks are completed the data is converted appropriately to values which the BLE Characteristics accepts and advertised to the Client.

### 4.5.2 Mobile Application

The duty of the app is to firstly establish a connection with the ESP32 microcontroller. Secondly, to subscribe to the different Services and Characteristics so it can be notified every time new data is available. After that the user clicks a button to start his/her run/walk and algorithm starts analysing the received data on the go. In the end when the user finishes the walk/run he is taken to another page where he can view the results of the analysis and either download an image of the graph representation or store the whole analysis on the app. The stored results can be later reviewed or deleted. The mobile app also handles the unsubscribing and disconnecting from the peripheral device. In order to utilise BLE every time the user leaves the main page where the live analysis happens the app disconnects from the ESP32 if previously connected.

## 4.6 Hook-up

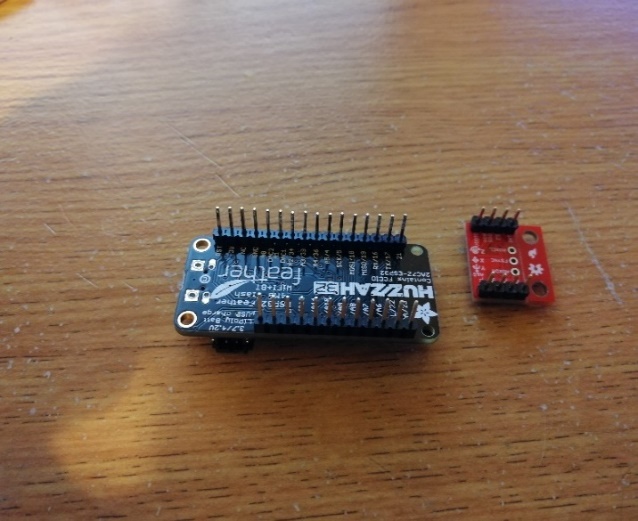
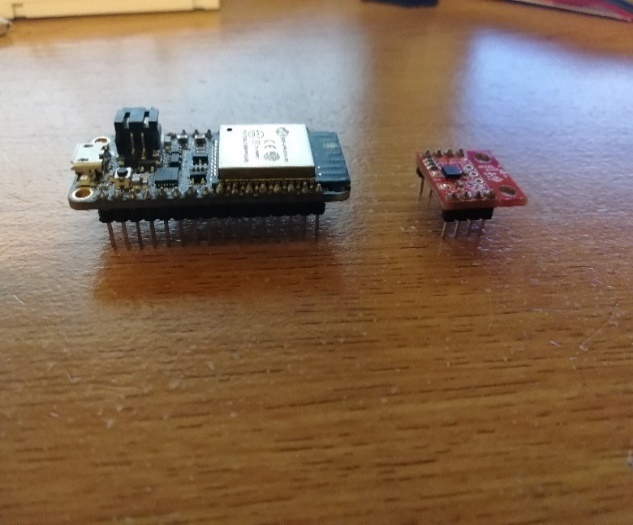
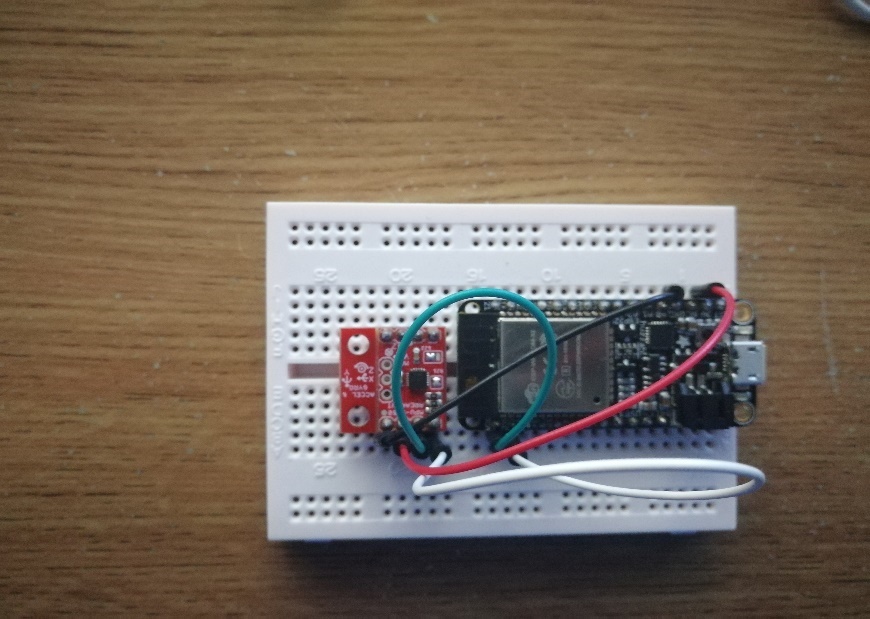
The initial step of the hook-up was the soldering part. Pins were soldered on both the ESP32 microcontroller and the sensor (Figures 9 and 10)

Figure 9

Figure 10

The pins were needed in order to put the components on a breadboard and connect the sensor to the microcontroller with jumper (signal) wires (Figure 11). A breadboard essentially is a board for making an experimental model of an electric circuit.



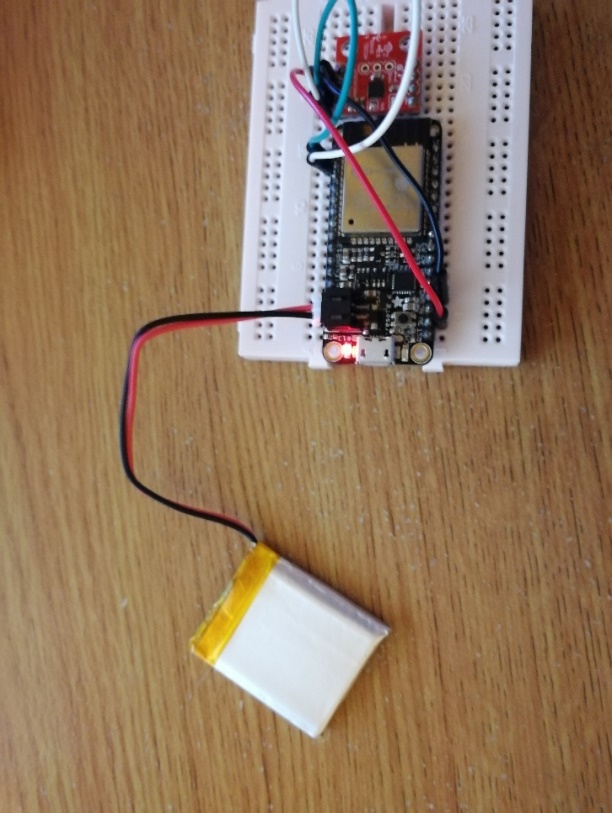
  
The wires are connected as follows. The red one connects the VDD pins (which provides the sensor with power), the black one connects the GRD (ground) pins, the green one connects the SDA (Serial Data Line) pins and finally the white one connects the SCL (Serial Clock Line) pins. Lastly, the power supply of the whole device comes from 3.7V Li-Po battery which is connected to the microcontroller via the designated JST jack (Figure 12).

Figure 11 Wiring of sensor and microcontroller

Figure 12 Battery connection

## 4.7 User Interface

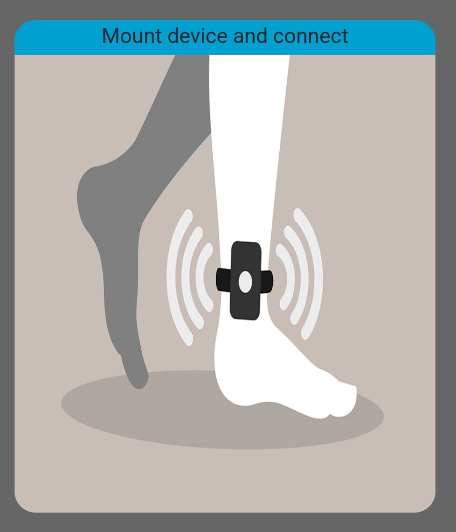
As mentioned in the non-functional requirements in an earlier chapter the desired properties of the mobile app are that it is easy to use and intuitive. Therefore, user interface design is focused on simplicity and clarity. Moreover, consistency of design is maintained throughout the different pages as well as a simple yet attractive visual representations of the current state of the app is presented. Also, all the navigational and functional buttons are sequentially organised and grouped accordingly. The mobile application consists of three different pages. The main one which appears when you start it up consists of visualised instructions (Figure13) in terms of a short sentence and a picture or a gif. It also provides functionality to observe step detection live. After you finish a cycle on the main page you are taken to the results page. There the results of the gait analysis are displayed. On top you have a box plot representing information on different temporal gait parameters with the option to change the parameter displayed from a dropdown menu. Below the plot average values of the gait parameters are displayed. Lastly there are two buttons that present the opportunity to record the analysis either by downloading an image of the box plot or storing all the results for later reference. The last page shows all the previously saved results in a clear way. It shows the type of the recorded analysis (walk or run) and the time and date of when it was saved in order to distinguish between them. The recorded analyses are sorted by the most recent one. The user is presented with two options: view and delete. On all pages there is a navigation sidebar placed on the top left corner that allows the user to navigate throughout the application. Lastly, not all control buttons are available to use at all time. The button appearing greyer and more transparent marks the button currently disabled (Figure 14). The intention of this is to not let the user willingly or not to disrupt the natural flow of the application. More pictures of the application can be found in Appendix D.

Figure 13 Visual instructions

Figure 14 Disabled Buttons

**Chapter 5**

# Detailed Design and Implementation

## 5.1 Implementation languages

### 5.1.1 C++

C++ is a general-purpose programming language. It has imperative, object oriented and generic programming features, while also providing facilities for low-level memory manipulation. The Arduino IDE in which the code for the microcontroller is written supports bot C and C++ but, in this project, C++ is used

### 5.1.2 HTML

HTML5 is the latest generation of HyperText Markup Language. It provides new functionality that is necessary for a good-looking web application. It has excellent cross-browser support which includes browsers like Google Chrome, Mozilla Firefox, Apple Safari, Opera and Edge. Only Internet Explorer has limited HMTL5 support, but it is being gradually replaced with Edge, so it is not a big problem.

### 5.1.3 CSS3

The use of Cascading Style Sheets (CSS) allows web design to be flexible and scalable as well as separate the website layout from its content. The language compliments HTML5, allowing complex and diverse styling with minimal amounts of code.

### 5.1.4 JavaScript

JavaScript is a very powerful and widely used language. In this project it is used for developing the back-end of the mobile application. It handles Bluetooth connectivity, analysis, displaying it and saving it.

## 5.2 Third Party Tools

### 5.2.1 Bootstrap

Bootstrap [23] is an open source toolkit for developing with HTML, CSS and JS. It is arguably the world’s most popular front-end component library. It is used to develop a responsive, mobile-first project, allowing the same page to be displayed appropriately on screens with different sizing. Moreover, Bootsrap components follow the same style convention, which means that the project is consistent and clean on all web browsers and mobile platforms. It was the library of choice because of the good documentation available, the overall popularity between developers and extensive list of components available to use of the shelve.

### 5.2.2 JQery

jQuery is a fast, small, and feature-rich JavaScript library. Its motto is “write less do more” and in this project it is mainly used by the previously mentioned Bootstrap**.**

### 5.2.3 Popper

JavaScript library used only by Bootstrap in this project. It is used to manage poppers which are elements on the screen that “pop out” from the natural flow in web applications. Common examples are tooltips, popovers and drop-downs.

### 5.2.4 Plotly

Plotly [24] is company that maintains the fastest growing open-source visualisation libraries for R, Pyhton and JavaScript. The JavaScript library is used in the project which is a high-level, declarative charting library. It has 20 chart types, including 3D charts, statistical graphs, and SVG maps.

### 5.2.5 JSLint

JSLint is a static code analysis tool used in software development for checking if JavaScript source code complies with coding rules. On this project it is used on top of InteliJ IDE and searches for flaws in the code like breaking style conventions or structural problems. The alternative considered were JavaScript Ling and JSure.

### 5.2.6 Plugins

The mobile application uses Phonegap and Cordova plugins (Phonegap is built on top of Cordova). The purpose of the plugins is to provide a JavaScript interface to access native components and functions beyond what is available to pure web applications.

## 5.3 Prototyping

During the development phase of the algorithm a prototype was developed which main purpose was to test a possible positioning of the sensor (on the front part of the thigh) and a way of calculating some of the temporal gait parameters namely swing and stance periods. As described previously stance is when some part of the foot touches the ground and swing is when it is in the air. The prototype used Yaw and Pitch values (that were computed on microcontroller) which showed its current position. Those values are usually used in aviation to determine the current position of the aircraft in the 3D space. It was then visualised on the app by drawing a line and moving it correspondingly. Afterwards, a simple algorithm was developed to monitor the movement and output the current phase of the leg depending on which way is the leg currently moving. The results from the prototype were not plausible enough. It was observed while walking that the outputted values are not consistent. When the leg was in the air and thus the correct output value being swing a lot of times sudden and short changes to stance were observed and vice versa. The main issue being that the values used for computing the phases were too noisy thus showing inconsistency. Furthermore, it was noted that the range of values received were in a rather short range. This means that the sensor positioned there is handicapped because that part of the leg does not move that much back and forth. All in all, the prototype was helpful in determining that this is not a feasible way of calculating swing and stance and that the positioning of the sensor should be reconsidered and changed.

## 5.4 I2C Communication Protocol

The communication between the sensor and the microcontroller is handled by the Inter-integrated Circuit (I2C) protocol. The SparkFun tutorial was used to gather information on how it works [25]. The protocol is intended to allow multiple “slave” chips to communicate with one or more “master” chips. As it can be seen from the hook-up section of the report it requires only two signal wires to exchange information (SDA and SCL). SDA wire transports data while the SCL wire is the clock signal. The clock signal is always generated by the current bus master. Some slave devices may force a delay on the master sending more data by using the clock. On a basic level messages sent are broken up into two types of frame. The address frame specifies to which slave the message is being sent and one or several data frames which 8-bit data messages are passed from master to slave or vice versa. The data is written on the SDA when the SCL line goes low and is read when the SCL line goes high. The time between data read/write varies from chip to chip.

## 5.5 Microcontroller Implementation Details

The flow of the microcontroller’s program splits into two main self-explanatory parts: setup and loop. In this chapter I will explain how different parts of the software work and to which part of the flow do they belong to.

### 5.5.1 Sensor Initialization

In the beginning connection with the sensor is established. A reference to the MPU9250 is initialized. In the library it is defined on which address the sensor should be found. The first step of communicating with the sensor is to check if it can be found on the specified address. After that some self-tests are performed in order to calibrate the sensor’s accelerometer and gyroscope. The next part is checking the magnetometer calibrating it as well (Figure 15). All the above happens in the setup part.

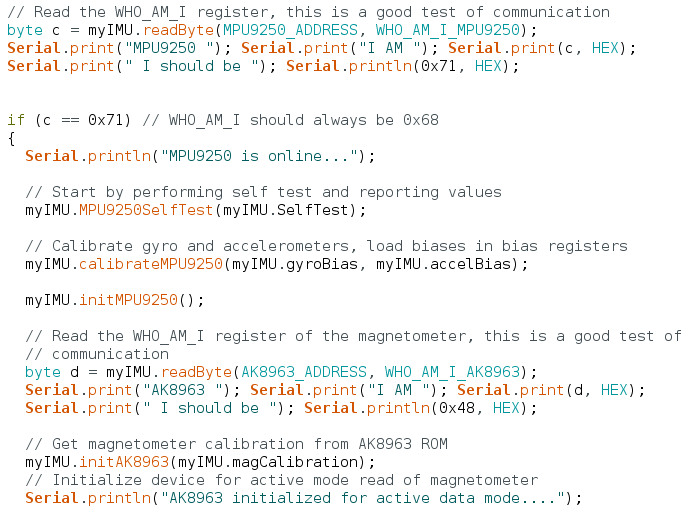


Figure 15 sensor calibration

### 5.5.2 BLE Initialization

Outside of the loop and setup the UUIDs of the different Services and Characteristics and a Boolean value stating the connectivity status are defined. Also, outside of the two parts callback functions are defined (Figure 16). They are used for changing the connectivity status. Then in the setup a Bluetooth server is initiated with the desired name that will show when other devices perform a scan. The Services are then created, and the corresponding Characteristics are also created, and their properties are defined. In the end the server starts advertising itself to other devices.

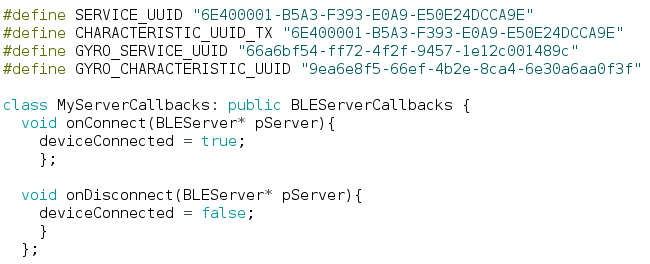


Figure 16 BLE setup

### 5.5.3 Reading Data

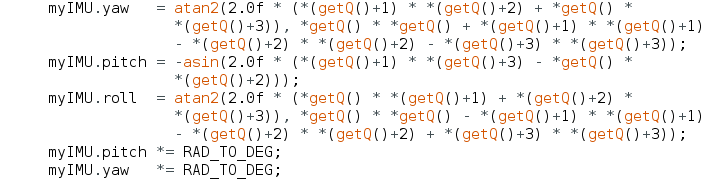
Reading the data from the sensor happens in the loop. It uses the earlier initialized MPU9250 object to access its attributes (the axes of the accelerometer, gyroscope and magnetometer). Each read value is manipulated the appropriate way. The x,y and z values of the accelerometer are transformed into mg (miliG), the gyro’s axes values into deg/s and the magnetometer values into miliGaus and are stored back into the MPU9250 object. Quaternions are then used with some of the transformed values to obtain the current orientation which is represented by the Yaw, Pitch and Roll. Optionally all the read values can be displayed in the serial monitor for debugging purposes. 

Figure 17 Yaw Pitch and Roll calculation

### 5.4.4 Sending Data

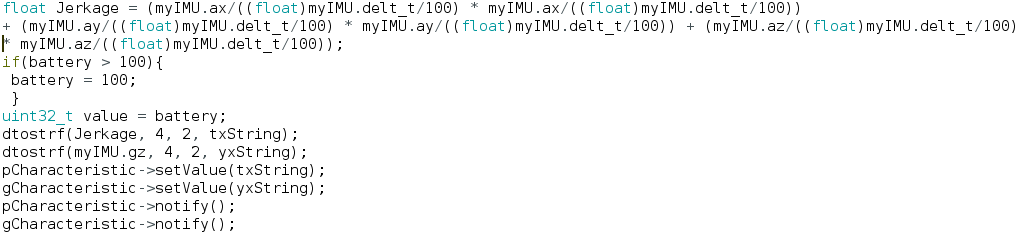
In the end of each loop it is checked if a connection has been established with another device. If it is a procedure to send the needed data is started. A Jerkage (explained in later subsection) value that is used in the initial detection algorithm is calculated from the values received from the accelerometer. After all the necessary calculations are made the values are transformed into character arrays because this is the type of values that the Characteristic accepts. When the values are placed in the corresponding Characteristics a notify function is called in order to notify the connected device that the value has changed (Figure 18).

Figure 18 Calculation of Jerkage and sending the data

## 5.6 Mobile Application Implementation Details

### 5.6.1 Managing Connection

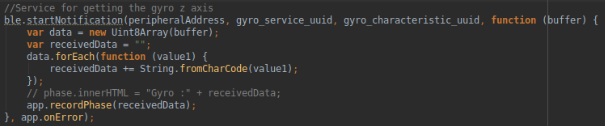
When the user clicks the connect button the process of connection begins. The mobile app uses a plugin to handle the process as JScript itself doesn’t have a built-in solution for that. Firstly, the user is prompted to enable his Bluetooth if he hasn’t done it. A connect function of the plugin is called and the peripheral device’s address, a success function and a fail function are passed as parameters. It tries to establish a connection for specified period and if it can’t the fail function is called which creates a pop-up informing the user of the failure. Otherwise a successful connection is established and in the front-end the user again is notified. At the same time in the backed the plugin’s startNotification function is used and the UUIDs of the Services and Characteristics are passed as parameters alongside a success and failure functions (Figure 19). When the process is successful the front-end view is changed. Disconnect replaces the connect button and some previously unavailable buttons are now made available for use. If then the user clicks disconnect the process is repeated the other way around. First the application stops the notifications then it disconnects from the peripheral device. It also disables the buttons again and changes the view of the pressed button. In order to preserve the battery of the mounted device when the user changes the page the application automatically breaks the Bluetooth connection. 

Figure 19 Start notification for receiving the gyroscope value

### 5.6.2 Initial Contact Detection

In order to understand how the algorithm works the value used for detection needs to be explained. In physics the time derivative of acceleration or rate of change of acceleration is referred to as jerk [26]. A person can feel the effects of jerk in lifts or trains. They are deliberately designed to minimise the felt effect but still it is not perfect. For example, when a lift suddenly starts moving the person senses a sudden loss of balance perfectly described by the name jerk. In the following paper [27] it was found that the sum of the squares of the jerk on all three axes gives a clean indication of initial contact. The researchers also observed that keeping the values squared helps ensuring a strong signal to noise ratio. They have named this calculated value “Jerkage” and to be clear the same name is used in this project when referring to it. The detection algorithm takes it as an input and performs a peak detection. A threshold is used to determine if the current value can be considered as an initial contact. If it passes through the set threshold it is compared with the current recorded maximum value (initially max=0). If it a larger value than the maximum is detected it is replaced. As soon as a value that passes the threshold but is lower than the maximum is detected it is considered that the maximum value is was a contact and is recorded. At this point a timer is started. The purpose of the timer is to filter out false positives by checking if enough time has passed since the last time an initial contact is detected (Figure 20). With the average cadence being 100- 115 steps/min [28] it was estimated that roughly two consecutive strides happen every 1 second. Taking into account the different walking patterns for different people the time threshold between initial contacts was set to 0.8s for walking and 0.5s for running. Through testing a problem with the algorithm was found. If a value above the threshold was detected but the following values fall below threshold the algorithm fails to record the stride. For this reason, a special case was developed to check for such occurrences and act properly. After each detection the maximum value and the timer are reset.

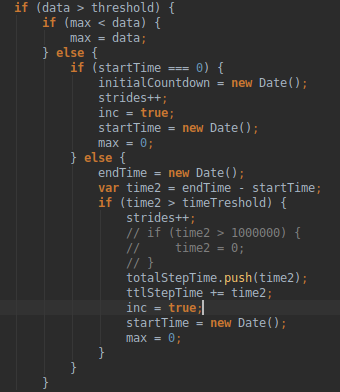


Figure 20

### 5.6.3 Stride and Swing Estimation

This part of the algorithm uses the gyroscope value to know in which phase is the leg. When the leg is on the ground and is in stance phase it moves backwards and when the leg is in the air it moves forward. The gyroscope value is positive for moving forward and negative when it moves backwards. A timer is used to estimate how much time has passed during each phase. When a transition occurs, the previous value gets overwritten. At each recorded initial contact, the last recorded values are saved for analysis. With experimentation it was found that the estimation is rather off. It starts the timer for the swing phase a bit later than the stance phase. Therefore, a manual correction of 0.15s were added to swing variable and 0.15s were subtracted from the stance variable before recording them.

### 5.6.4 Live Graph

When connection is established the user is given opportunity to observe the received data. Using the plotly library a line graph is created. It represents the initial contact of the foot with the ground. Each time the mobile app receives new information it is added to the graph live. In order to keep it readable a counter is initialized to keep track of the points represented on the graph. When they become more than twenty the graphs start sliding to show only the last twenty values received. On the top left corner there are zoom in, zoom out and auto scale buttons for the user to adjust the plotted graph (Figure 22).

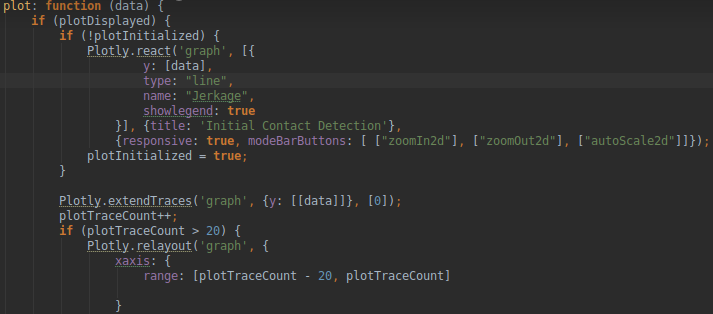


Figure 21 Live plot initialization

### 5.6.5 Analysis Representation

The analysed data is represented in two ways. Firstly, using the plotly library several graphs are created by passing arrays of data. Those arrays contain the stored data from the analysis. The graphs are box plots containing the maximum, minimum, median, upper and lower quartile values. They are not manually calculated but rather generated from the library itself. Secondly, after making the plots the mean values of the different parameters are manually calculated using the values from the arrays. Additionally, cadence is also calculated. Then using JavaScript, a table gets filled with all the data for the user inspect.

### 5.6.6 Download Analysis Plot

The user can download the currently displayed box plot as a PNG. A pop-up prompts what is the desired name of the picture to be downloaded. After a name is selected a function from the plotly library that returns the graph’s URL is called. The URL returned contains information about the picture in terms of its type, base encoding and the encoded data itself. An XMLHttpRequest object is created whose methods transfer data between a web browser and a web server is initialized. Its open function is used with a GET method. It is required to be invoked before sending the request in order to validate and resolve the URL or URI used. The open function is forced to return a blob type object. A blob object basically represents a file-like object of immutable raw data. The destination of the image that is going to be downloaded is specified and a success and fail responses are created. The XMLHttpRequest is then send and the user is notified of the outcome either positive or negative.

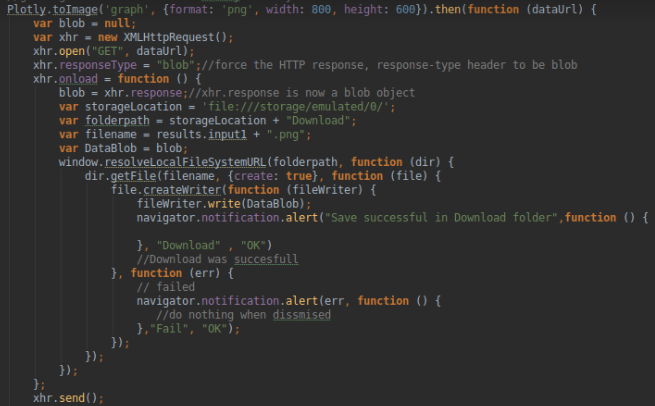


Figure 22 Process of downloading the box plot

### 5.6.7 Storing Results

The other option a user can take advantage of is to store the results. Local storage is used to save the results accompanied with a timestamp and the type (walk or run) in order to make it easier for the user to distinguish between the different stored results. After successful storage a message is generated, and the button is disabled to avoid duplication.



Figure 23 Process of saving results

### 5.6.8 Old Results Page

The contents of the saved results page are generated on the go. A function loops through the local storage and creates a window for every entry with its corresponding timestamp and type of analysis (run/walk) in chronologic order. The user can then choose to view the results and is redirected to the page showing the results in detail or delete an entry. If there are no saved results a window is generated to notify the user with a button redirecting him to the main page where he can start a new analysis.

**Chapter 6**

# Verification and Validation

## 6.1 Verification

This section aims to demonstrate that the system developed during the project meets the requirements set in Chapter 3. Both the functional and non-functional requirements are mosltly satisfied. The system successfully gathers the data from the sensor and sends it to the mobile application. The user can watch a live graph representation of the data used for the initial contact detection and can distinguish when such an event happens. The data is then analysed and displayed in box plots and a table which can be stored and/or downloaded on the phone. The device is easily mounted and is seemingly not obtrusive while walking or running. Furthermore, the application itself is straightforward and easy to use with clear and concise controls. To conclude, even though the analysis algorithm can be improved it still shows plausible results.

## 6.2 Validation

### 6.2.1 Testing Overview

Throughout the development process of this project it has been extensively tested and the different components have proven to be robust and capable. To show that the system works effectively series of pictures will be shown each demonstrating individual components properly working.

### 6.2.2 BLE Test

This test can be split down to two. The first one being if the BLE configuration on the microcontroller works appropriately and sends the data. The second one is the actual connection between the microcontroller and the mobile application and the correct transmitting of data.

1.Microcontroller  
To test the BLE connection of the microcontroller a third-party app called NRF [29] Connect was used. It is a powerful generic tool that allows you to scan and explore Bluetooth Low Energy devices and communicate with them. It supports many different functionalities but the ones that we are concerned with are the ability to scan for BLE devices, connect with them and parse the advertisement data so we can see the value sent. The following pictures show the troubleshooting process. (Figures 26,27)

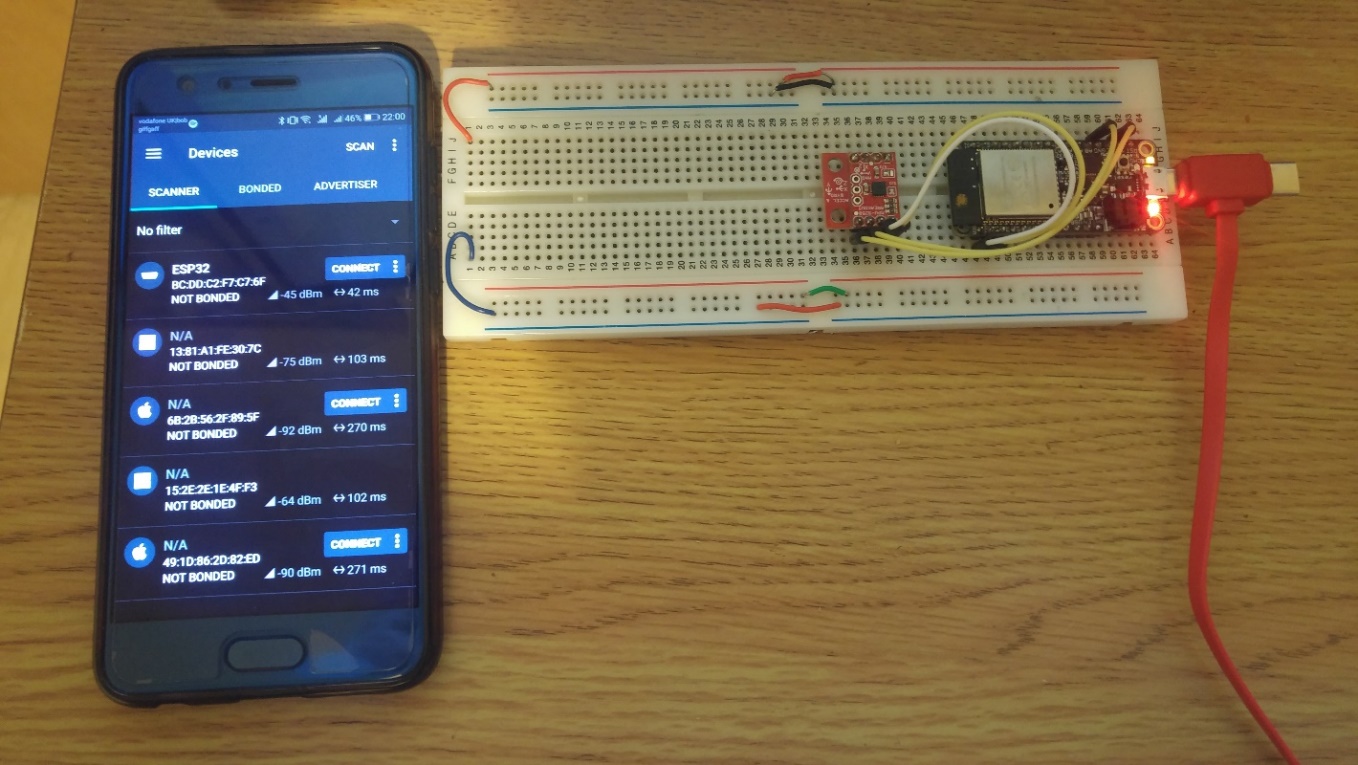


Figure 25 Scanning for the ESP32 microcontroller

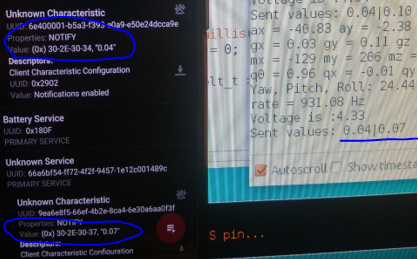


Figure 26 Sending and receiving data

2. Microcontroller to App

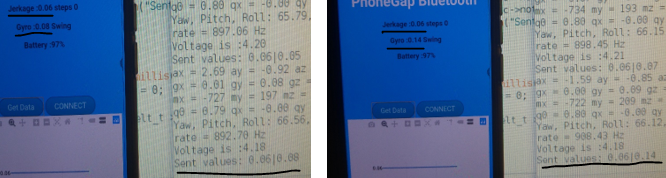
The developed skeleton app uses cordova plugin to establish a connection with the microcontroller and then subscribes to be notified every time the value to which it has been subscribed to changes. The pictures below represent a process of connection and then the values received compared to the values sent from the microcontroller (Figures 28).

Figure 27 Two tests representing the values sent and the values received

### 6.2.3 Application Error Handling

Here it will be demonstrated how the application handles unexpected errors such as sudden loss of connection with the peripheral device (Figure 29) or if the application tries but is not able to connect for a long period of time (Figure 30).

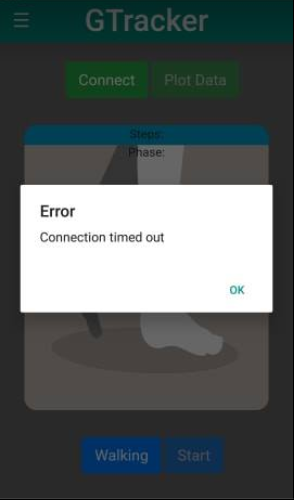
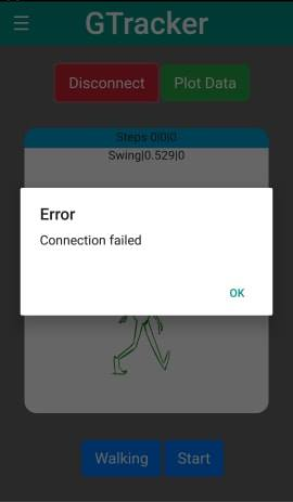


Figure 28 Connection fail Figure 29 Connection timeout

**Chapter 7**

# Results and Evaluation

In this chapter the final product is presented. The evaluation of the project was carried out in several different ways that try to show how accurate is the analysis of the application as well as how it compares pricewise to other gait analysis options.

## 7.1 Final System

In the end the hardware components had to be somehow protected from the weather and from accidental damage for it to be used freely outside. For this reason, a shockproof and waterproof protective case was acquired. The hardware components with all the wiring were hidden inside (Figure 31). The case has enough room to fit everything easily and to plug in a micro USB cable to charge the battery if needed. A Velcro strap was then stuck to the outside of the case to make attaching in to the user’s leg rather easy (Figures 32 ,33). A couple of tests were made to check if the Velcro strap holds the case tight enough by shaking the leg. The results were that it stayed in place.

Figure 33 Velcro strap attachment

Figure 31 Hardware placed inside case

Figure 32 Case attached to the leg

## 7.2 Analysis Algorithms Evaluation

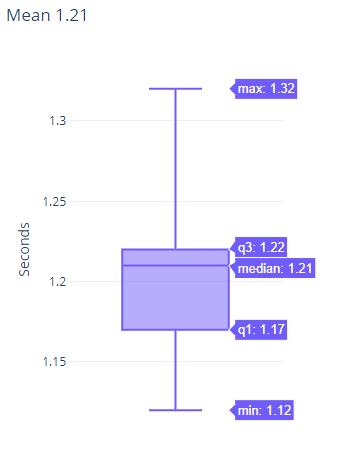
The evaluation of the algorithm was accomplished by making several tests. Their premise was to mount the device on myself, connect to it with the mobile application and conduct several test walks and runs while other people record the gait data manually by using stopwatches. The tests consisted of three 15 stride long (30 steps) walks and runs. After finishing with the tests, the manually gathered data was analysed and compared to results from the app. The results are represented giving several different parameters that are used when representing the data in a box plot. The explanations of the given parameters are as follows:

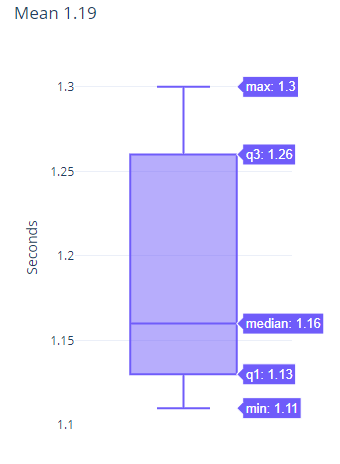
* Median – the mid-point of the data
* Minimum – the minimal recorded value
* Maximum – the maximal recorded value
* Lower quartile (q1) – the value bellow which fall 25% of all recorded values
* Upper quartile (q3) –the value bellow which fall 75% of all recorded values
* Mean – is the average value of all collected data

It is important to note that the manually recorded values are not 100% accurate as there exists high probability of human error. Also, the values received often are less than or around 1 second which is rather small, and the human reaction time can be off. The tests were conducted this way because the alternative is to rent a system that records accurate results but is expensive. Detailed results of the manually recorded data can be found in Appendix E.

The readings of the walk test showed that the algorithm correctly recorded the number of initial contacts in the first two tests 15/15 but missed a couple on the third and last test detecting only 12/15 strides. In both walking and running tests the left graph represents the manually recorded results while the right one shows the results produced from the algorithms.

**Stride parameter tests:**

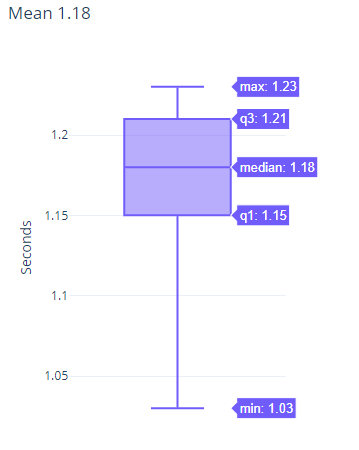
**Test 1**

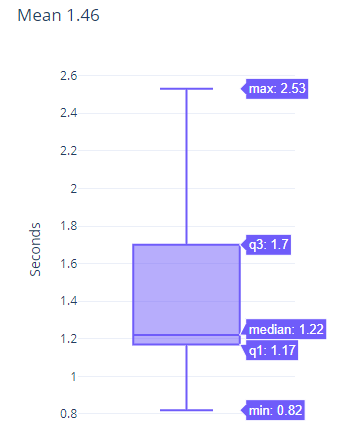


The evaluation of the first test shows good results. Overall all values are really close with the largest deviation being only +0.05s (4.3%) for the median and the mean value having a positive deviation of 0.02s (1.68%)  
**Test 2**



The second suite of results are slightly worse but still good. Here the largest deviations are in the minimum and maximum recorded values but because the app’s minimum value is lower while its maximum value is higher than the manually recorded one, they compensate each other. This makes the difference between the mean values as close as 0.03s (2.56%)

**Test 3**

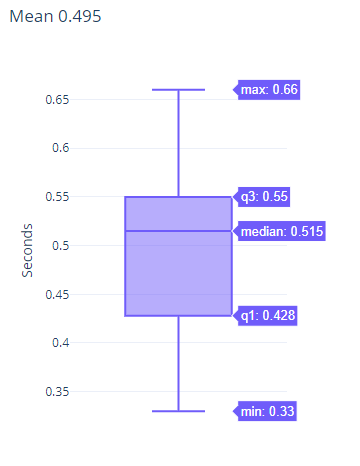
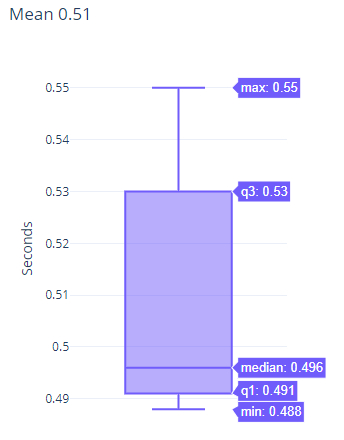


The last test showed the worst results. While the median and the lower quartile have a low deviation, all other parameters show a significant difference. The largest one being the difference between the two maximum values +1.30s (more than 100%). The reason behind that is that the algorithm only detected 12 out of 15 initial contacts. The way the algorithm calculates the time passed is by counting the time between two consecutive initial contacts and in this case when it misses to record some strides the time increases correspondingly. This also affects the mean value in the end increasing it as the overall time is likely to be the same, but it is divided by less strides thus making the deviation in this test case larger than the previous two +0.28s (23.7%).

In conclusion, the results from the initial detection and stride time estimation part of the algorithm can be considered as good. It can be observed that most of the times the algorithm overestimates the time values making the deviations mostly positive.

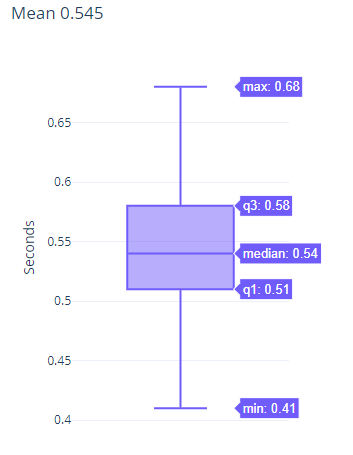
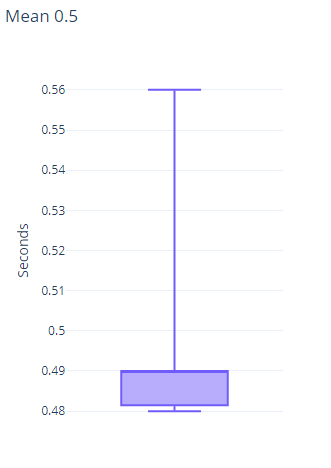
**Swing parameter tests:**

Test 1

****

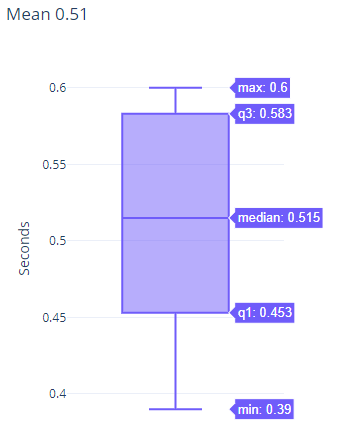
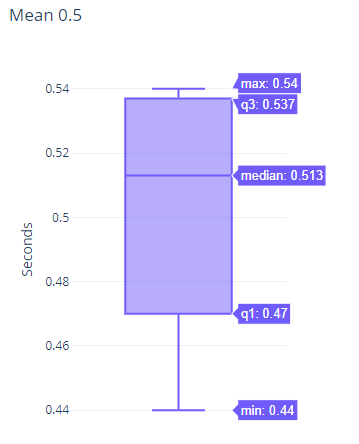
This test shows relatively low deviations in all parameters but the minimum and maximum values. Overall the results from this test are plausible with the mean differing only with +0.015s (3.2%)

**Test 2**

****

The same tendency as the stride parameter test occur here as well. The results show a larger deviation than the previous test with the median and mean having almost the same negative deviation of -0.05s (9.26%) and -0.045s (8.26%) deviation respectively.

**Test 3**

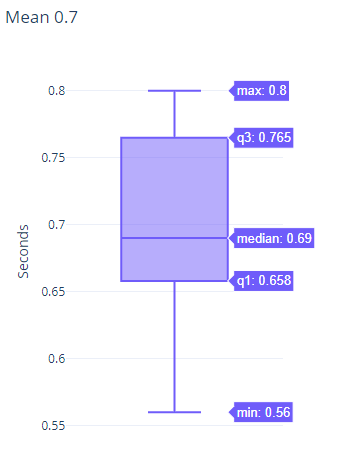
****

In contrast to the stride parameter testing the final swing test shows the best results. Overall all the values except the mean represent the lowest deviations of all previous tests but even the mean has a -0.01s deviation which is only -1.96%.

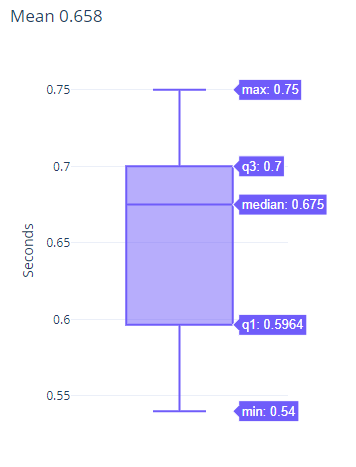
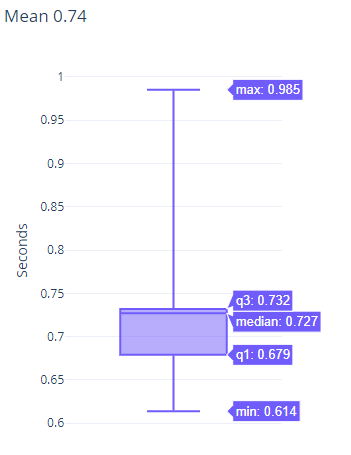
To conclude, the measurement of the swing parameter from the algorithm is good. The recurring thing happening in all three tests is that the algorithm’s minimum and maximum values are closer together in comparison with the values from the manual gathering which probably comes from the manual adjustment in the algorithm.

**Stance parameter tests:**

Test 1

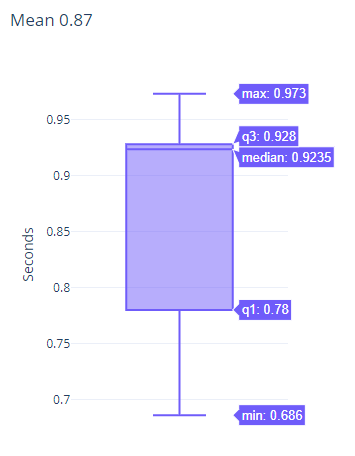
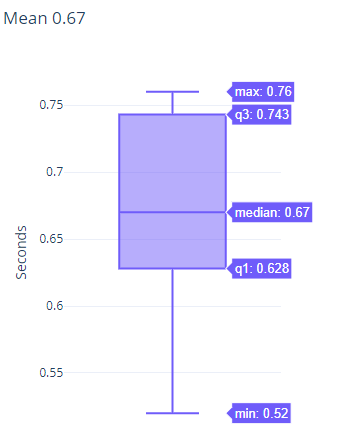
****

This test accomplished a rather good outcome. Except from the maximum and minimum deviation all other are lower than 0.036s. The biggest recorded deviation can be seen in the maximum value being 0.117s which is a 14.625% increase. The mean has a +0.03s (4.29%) difference.

****T**est 2**

This test shows worse results than the previous one with the deviation of the median and mean values being +0.052s and +0.082s which are 7.7% and 12.5% increase respectively. The maximum values also show a big difference of +0.235s

**Test 3**

****

The final test suite provides the worst results. Here the deviations are positive and quite large on all parameters. With the largest deviation being +0.2535s (37.83%) in the median parameter.

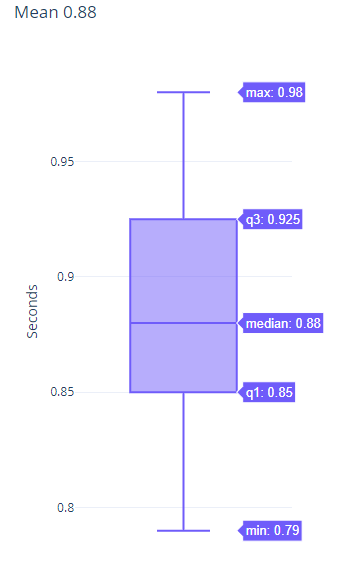
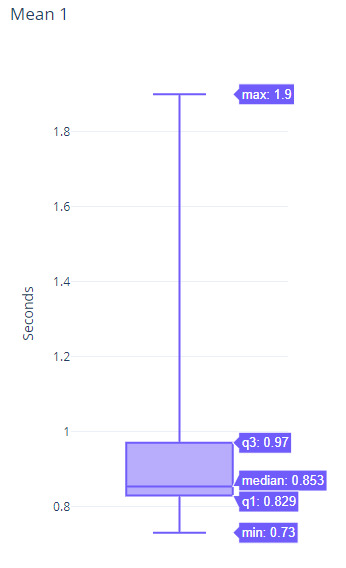
Finally, the algorithms estimation of the stance parameter can be rather poor at times showing that an improvement can be made. It can be observed that the algorithms almost always overestimate the parameters on all three tests.

The evaluation of the algorithm with regards to walking can be summed up as showing more good results than performing badly. If the performance of estimating the stride time and swing time of the algorithm can be classified as good, then the final part of finding the stance time needs improvement.

Next evaluation of running will be presented the same way the previous tests were. On the left the manually recorded results are shown and on the right the algorithm results. The tests recorded 13/15, 14/15 and 15/15 strides accordingly.

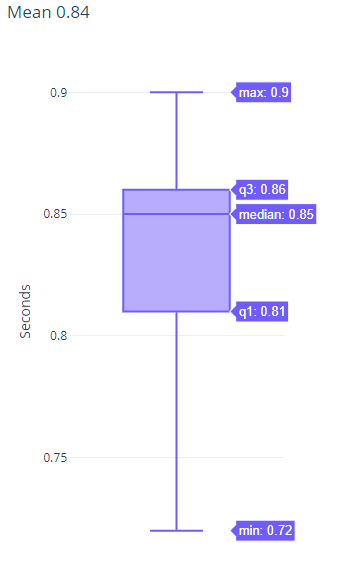
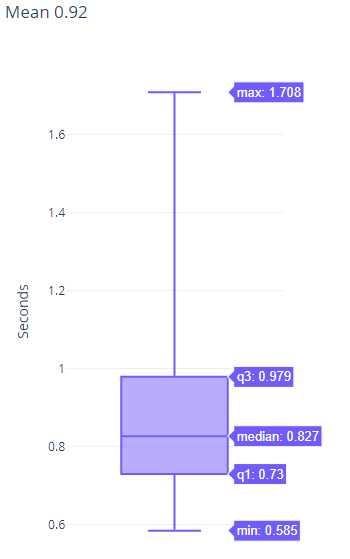
**Stride parameter:**

**Test 1**



The first tests show close values for the median, lower and upper quartiles and the minimum values. A big difference can be seen in the maximum recorded value. As explained in previous test suite the problem comes when the algorithm misses to record a step and the time passed between two strides becomes larger thus widening the gap between the two mean values which is +0.12s or 12%. But the quartile values and the median are rather grouped together which means that most of the values are similar to the manually recorded ones.

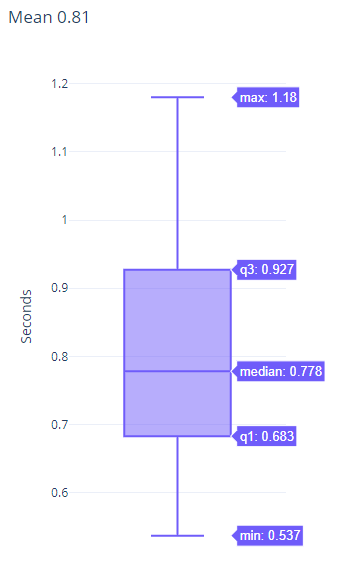
**Test 2**



This test missed only one step which is why here the gap between the mean values is smaller being +0.8s or smaller than 10%. The large difference between the two max values can be again observed. This time in almost all the other parameters the difference is larger, but the median recorded value is rather close being only 0.023s (2.7%).

**Test 3**

The last test suite shows the best outcome. The mean value has a +0.02s (2.5%) deviation. The maximum values are not as far apart as those in the previous tests. The median value is also close with +0.022s difference (2.75%). Otherwise the quartile values and the minimum and maximum values have a larger gap of at least 0.095s and at most ****0.32s.

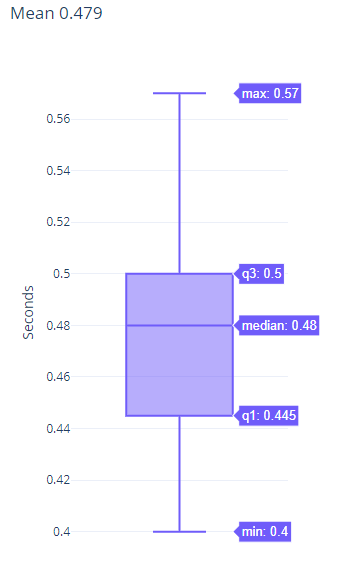
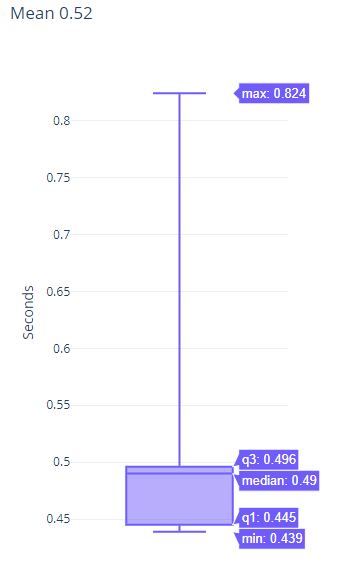
****

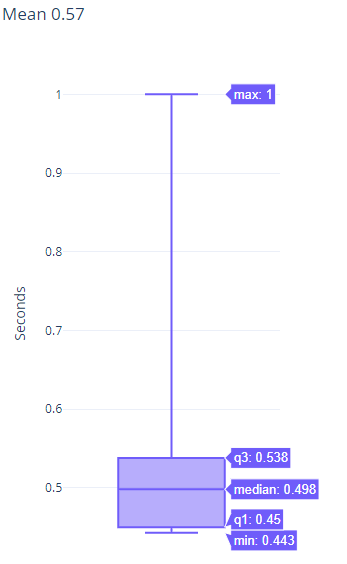
In conclusion, the algorithm’s stride detection while running shows promising results but again struggles if it fails to detect some strides. Overall the estimated results mostly overestimate the maximum value and underestimates the minimum one which balances out the mean value which also is always higher the manually recorded one.

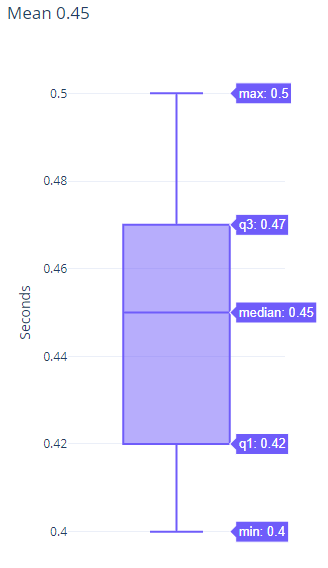
**Swing parameter test:**

**Test 1**

The first test shows plausible results. The lower quartile values are the same and the minimum, median and upper quartile values are really close to each other which means that overall the recorded values are close. The only big gap is in the maximum value which is 0.254s (44.56%). This is the reason the mean value has a +0.041s (8.56%) and not closer.

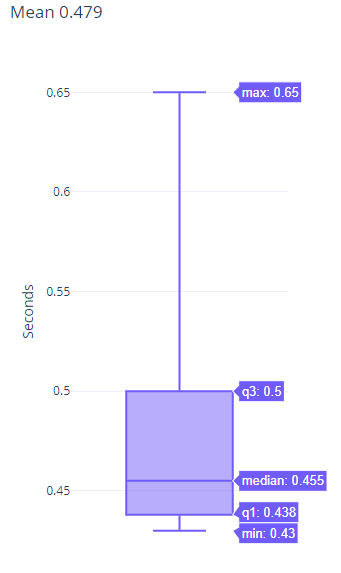
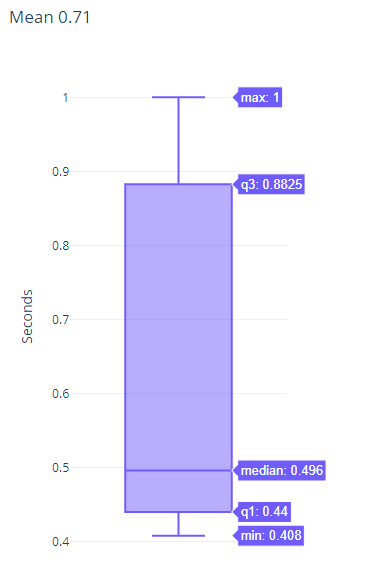
****

**Test 2**



The second test shows similar results but worse with all parameters except the maximum one which is +0.5s (100%) being grouped together. Here the upper quartile shows a larger difference of +0.068s (14.47%) which means that there are several values recorded from the algorithm that are larger. This makes the gap of the mean value 0.12s (26.67%).

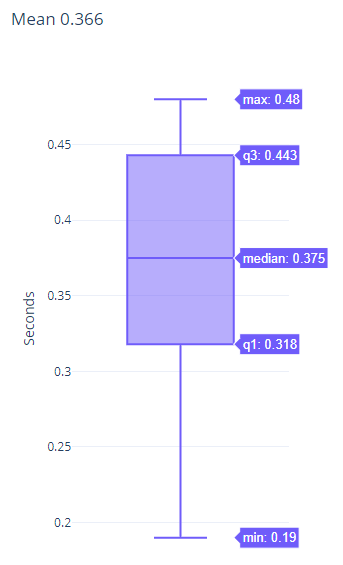
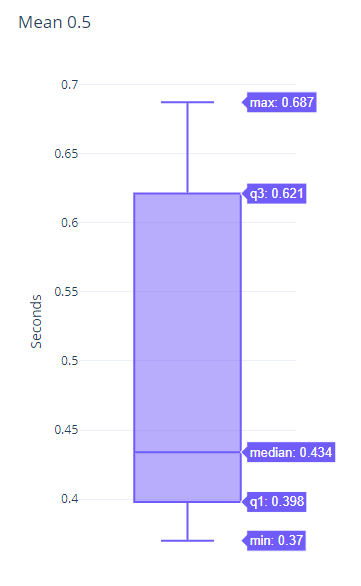
**Test 3**

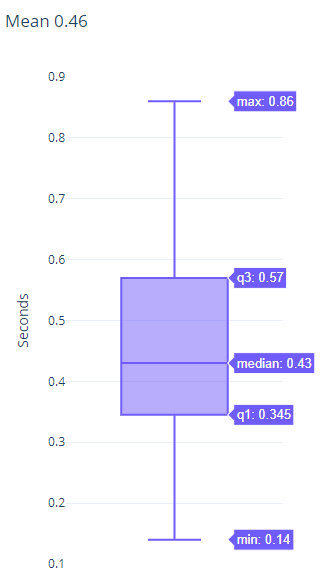
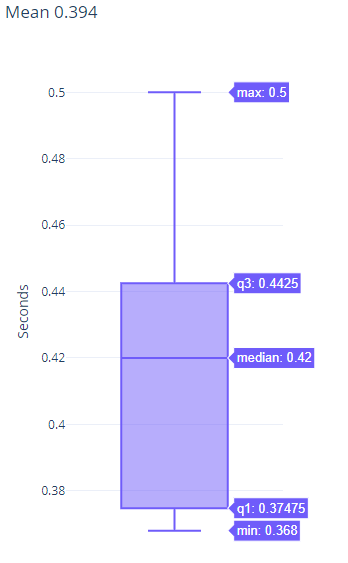


This test suite continues the trend. It is worse than the previous ones. The median, lower quartile and minimum values are close to each other while the upper quartile and maximum values have large gaps meaning here again there were several recorded values close to the maximum one. The mean value in this test is +0.231s (48.23%)

**Stance parameter tests:**

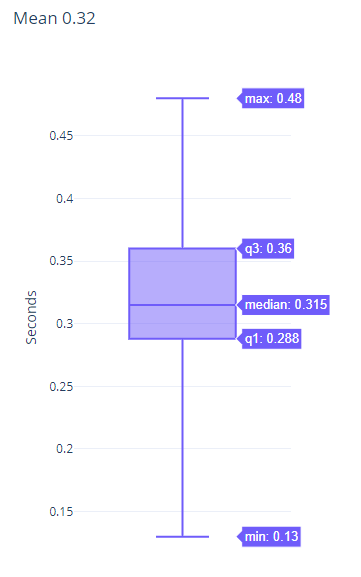
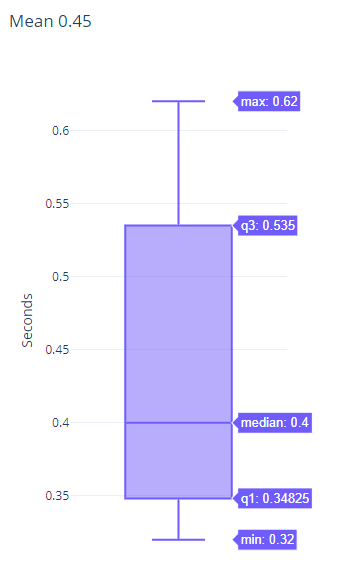
**Test 1**The first test shows rather large differences on all parameters. The minimum and maximum values are both larger that the manually recorded ones thus not compensating for each other. This makes the mean difference +0.134s (36.61%)

****

**Test 2**

The second test suite performs better than the fist one. The median value has only 0.01s (2.38%) deviation. Here the maximum and minimum values compensate for each other making the gap of the mean value smaller 0.066s (16.75%) which is still quite large.

**Test 3**

****

The final test doesn’t show plausible results again. Every parameter has a rather large deviation. The median value has a + 0.085s (26.98%) difference and the mean has a 0.13s (40.625%) gap.  
In conclusion, the values received from the stance parameter tests show that a lot of improvement needs to be made to achieve plausible results. Most of the times the results shown are larger than the manually recorded ones.

To sum up, the running tests perform worse than the walking ones. The only parameter that shows plausible results is the stride one which means that a lot of improvement must be made with the stride and swing estimation algorithm.

## 7.3 Price Comparison

The price of the hardware components come up to 46 pounds.

* ESP32 HUZZAH Feather Board – 19 pounds
* SparkFun IMU – MPU 9250 – 14 pounds
* 500 mAh 3.7 LiPo battery – 7 pounds
* Soldering pins – 3 pounds
* Protective case – 3 pounds

A comparison with the systems mentioned in the related work section would be ideal but they do not disclose their prices without a specific inquiry. For this reason, an assessment is made by comparing this system with the prices for conventional gait analysis in a laboratory. The 3D gait analysis available on the Run Lab [30] propose three different categories. Novice (Technique Screening) with an 80-pound price, Competitor (Performance/Injury) with a price of 125 pounds and Pro (Performance/Injury) with a price tag of 180 pounds. The Foot Function centre [31] has a pricing for Clinical Gait Analysis Assessment of 90 pounds per hour.

The accuracy of the parameters received from the system and the lab analysis are not even close, but this portable wearable system has a low price considering it can be reused many times.

**Chapter 8**

# Summary and Future Work

## 8.1 Summary

The final system produced allows the user to quickly and easily mount the device on the leg and go outside for run or walk. As the hardware is placed in shockproof and waterproof container the weather is not problem and there is a good protection from different accidents. On overall the project has been successful in achieving the set objectives. While the algorithm has room for much improvement it provides a very good proof of concept and a starting point for further development.

## 8.2 Future Work Opportunities

This section describes further development opportunities. It includes ideas considered in the initial gathering of the requirements and research and while developing the project.

An easier opportunity to enhance the system is to completely duplicate the hardware setup and use it to gather and analyse information from the other leg thus presenting more information in the results. The only additional thing that needs to be made code wise for the system to work is to setup the Bluetooth connection in terms of creating Services and Characteristics with their own UUIDs and plugin the received data into the detection algorithms.

Another idea would be to enhance the currently existing algorithms to produce more accurate results as well as to introduce new ways to calculate other spatio-temporal parameters such as step time, step length and speed etc.

A more challenging and long-term development opportunity is to research how to make a gait analysis based on plantar pressure and incorporate pressure sensor in the system. The end product will resemble the Smart-Insole device mentioned in the related work section earlier in the report.

Design wise improvements can be made regarding the transition between pages. They can be made more fluent and more pleasing for the eye. Also, a battery reading can be incorporated to show how much battery does the mounted device currently have and to inform the user when it needs charging

## 8.3 Conclusions

In conclusion, the fact that all the planned features outlined in the requirements are for the most parts met is satisfying. The only disappointing thing can be the accuracy of the algorithm. Nonetheless, the research undertaken, and the number of new things learned will be very beneficial for me in the future. Moreover, the developed system looks promising and there is a lot of opportunities for future work in different directions.

# Appendix A – References

[1] Bonnefoy-Mazure, A. and Armand, S. (2015). Normal gait. p.200

[2] Pistacchi M., Gioulis M., Sanson F., Giovannini E., Filippi G., Rosetto F., Marsala S. Z. (2017) Gait analysis and clinical correlations in early Parkinson’s disease

[3] Figure of gait cycle available at: <https://www.researchgate.net/publication/230763961_Gait_Symmetry_in_Children_with_Autism/figures?lo=1>

[4] Perry J., Burnfield J. (2010). Gait analysis: Normal and Pathological Function

[5] Essa S., Gait analysis, Available at: <https://www.slideshare.net/shimaa2022/gait-analysis-15743497>

[6] Kintec (2016) The 4-point run analysis. Available at: <https://www.kintec.net/blog/the-run-centre-4-point-run-analysis/>

[7] Richards J., Chohan A., Erande R., Biomechanics, Chapter 15. Available at: <https://musculoskeletalkey.com/biomechanics-2/>

[8] Tekscan. (2019). F-scan system, last accessed March 21,2019. Available: <https://www.tekscan.com/products-solutions/systems/f-scan-system?tab=configuration>

[9] BTS Bioengineering. (2019). G-walk. last accessed March 21,2019. Available: <https://www.btsbioengineering.com/products/g-walk-inertial-motion-system/#1484563444724-22eeaa09-fb7f>

[10] Lin, Feng & Wang, Aosen & Zhuang, Yan & Tomita, Machiko & Xu, Wenyao. (2016). Smart Insole: A Wearable Sensor Device for Unobtrusive Gait Monitoring in Daily Life. IEEE Transactions on Industrial Informatics

[11] Bluetooth Low Energy for Smart Application. Available at:

<https://www.slideshare.net/adunnantakaew/wearable-device-bluetooth-low-energy-ble-connect-with-android>

[12] Ray B., (2015). Bluetooth vs Bluetooth Low Energy: What’s the Difference. Available: <https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy>

[13] (2016) Bluetooth Classic vs Bluetooth Low Energy: Which one is right for you?

<https://www.plugintoiot.com/bluetoothclassic-vs-bluetoothlowenergy/>

[14] Bluetooth GATT found on the Bluetooth official website: https://www.bluetooth.com/specifications/gatt/

[15] Getting Started with ESP32 Bluetooth Low Energy (BLE) on Arduino IDE. Available at:

<https://randomnerdtutorials.com/esp32-bluetooth-low-energy-ble-arduino-ide/>

[16] Kalnoskas A. (2015). A beginner’s guide to microcontrollers. Available:

<https://www.microcontrollertips.com/a-beginners-guide-to-microcontrollers-faq/>  
[17] (2019) IoT Hardware Guide. Available: <https://www.postscapes.com/internet-of-things-hardware/>

[18] Official website of the ESP32 chip. Available at: <https://www.espressif.com/en/products/hardware/esp32/overview>

[19] Passara V. M. N., Cuccovillo A., Vaiani L., Carlo M., Campanella C. E. (2017) Gyroscope Technology and Applications: A Review in the Industrial Perspective, Sensors Open Access Journals, vol.17 issue 10

[20] Ahmad N., Ghazilla R. A. R., Khairi N.M. (2013) Reviews on Various Inertial Measurement Unit (IMU) Sensor Applications, International Journal of Signal Processing Sustems, vol.1, No 2.

[21] Towsend K. (2015) Introducing the Adafruit Bluefruit LE UART Friend. Available: <https://learn.adafruit.com/introducing-the-adafruit-bluefruit-le-uart-friend/current-measurements>

[22] Jscrambler. (2017). 12 Frameworks for Mobile Hybrid apps, last accessed March 10, 2019. Available: <https://blog.jscrambler.com/10-frameworks-for-mobile-hybrid-apps/>

[23] Bootstrap official website: <http://www.getbootsrap.com/>

[24] Plotly official website: <https://plot.ly/>

[25] Sfuptownmaker. I2C tutorial. Available: <https://learn.sparkfun.com/tutorials/i2c/all>

[26] Eager D., Pendrill A., Reistad N. (2016) Beyond velocity and acceleration: jerk, snap and higher derivatives. European Journal of Physics, vol 37, number 6, European Journal of Physics.

[27] Visi, Federico; Georgiou, Theodoros; Holland, Simon; Pinzone, Ornella; Donaldson, Glenis and Tetley, Josie (2017). Assessing the Accuracy of an Algorithm for the Estimation of Spatial Gait Parameters Using Inertial Measurement Units: Application to Healthy Subject and Hemiparetic Stroke Survivor. In: 4th International Conference on Movement Computing (MOCO ’17), 28-30 Jun 2017, London, UK, ACM.

[28] Website from utddallas stating the average walking cadence: <https://www.utdallas.edu/atec/midori/Handouts/walkingGraphs.htm/>

[29] Official website of nRF connect for mobile:   
<https://www.nordicsemi.com/Software-and-Tools/Development-Tools/nRF-Connect-for-mobile/>

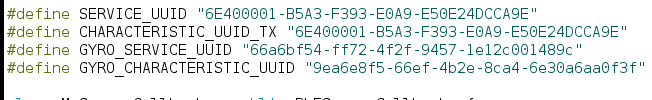
[30] The Run Lab official website: <https://www.therunlab.co.uk/3d-gait-analysis/prices-2/>

[31] The foot function centre official website: <https://www.footfunctioncentre.com/clinical-gait-analysis.html>

# Appendix B – Maintenance Guide

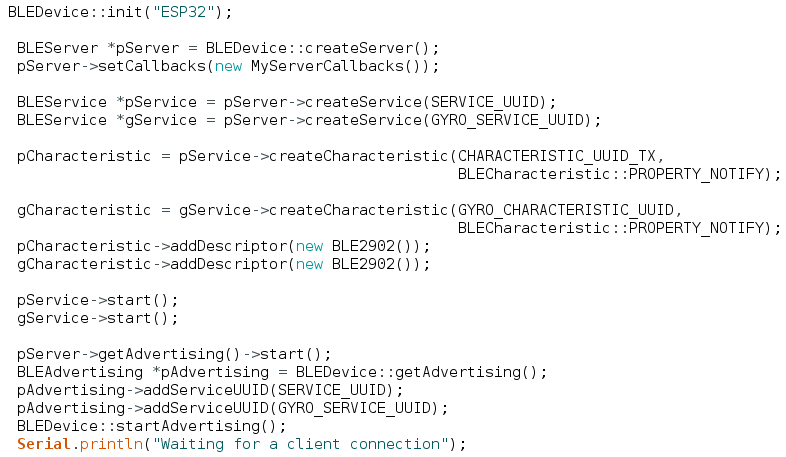
The use of the Maintenance Guide is to outline how changes to the system can be made with regards to using other parameters from the sensor.

Step 1

In the Arduino sketch define new Service and Characteristic UUIDs and create a new variable referring the Service

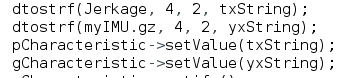
You can use a random UUID generator: <https://www.uuidgenerator.net/> or search for a defined Service and characteristic in the Bluetooth official website.

Step 2

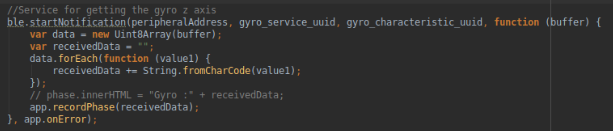
Initialize the Service and Characteristic in the setup method the following way

Step 3

Convert the wanted value from float to a char array and set it to the characteristic the following way



Step 4

In the mobile application if the newly created Characteristic has the property notify you can subscribe to receiving updates the following way 

You will have to substitute the second and the third parameter with the corresponding UUIDs. The first four lines are needed to read the value. Replace the app.recordPhase function with the function to which you want to pass the received information.

If the newly created Characteristic has other properties please refer to the documentation of the ble plugin to find out the appropriate function you need.  
Link:  
<https://github.com/don/cordova-plugin-ble-central>

# Appendix C – Technical Characteristics

Adafruit HUZZAH32 – ESP32 Feather Board:

* 240 MHz dual core Tensilica LX6 microcontroller with 600 DMIPS
* Integrated 520 KB SRAM
* Integrated 802.11b/g/n HT40 Wi-Fi transceiver, baseband, stack and LWIP
* Integrated dual mode Bluetooth (classic and BLE)
* 4 MByte flash include in the WROOM32 module
* On-board PCB antenna
* Ultra-low noise analog amplifier
* Hall sensor
* 10x capacitive touch interface
* 32 kHz crystal oscillator
* 3 x UARTs (only two are configured by default in the Feather Arduino IDE support, one UART is used for bootloading/debug)
* 3 x SPI (only one is configured by default in the Feather Arduino IDE support)
* 2 x I2C (only one is configured by default in the Feather Arduino IDE support)
* 12 x ADC input channels
* 2 x I2S Audio
* 2 x DAC
* PWM/timer input/output available on every GPIO pin
* OpenOCD debug interface with 32 kB TRAX buffer
* SDIO master/slave 50 MHz
* SD-card interface support
* Dimensions: 51.0mm x 22.7mm x7.3mm / 2.0” x 0.9” x 0.3”
* Weight: 6.8g/0.2oz

The full datasheet containing tutorials of the microcontroller board is available at: <https://cdn-learn.adafruit.com/downloads/pdf/adafruit-huzzah32-esp32-feather.pdf>

All of the above and more information can be found on: <https://www.adafruit.com/product/3405>

SparkFun IMU breakout - MPU – 9250:

* Digital-output X-, Y-, and Z-axis angular rate sensors (gyroscopes) with a user-programmable full-scale range of ±250, ±500, ±1,000 and ±2,000°/sec and integrated 16-bit ADCs
* Digital-output triple-axis accelerometer with a programmable full-scale range of ±2g, ±4g, ±8g and ±16g and integrated 16-bit ADCs
* 3-axis silicon monolithic Hall-effect magnetic sensor with magnetic concentrator
* Digitally programmable low-pass Gyroscope filter
* Gyroscope operating current: 3.2mA
* Accelerometer normal operating current: 450µA
* Magnetometer normal operating current: 280µA at 8Hz repetition rate
* VDD supply voltage range of 2.4 – 3.6V
* Small board design
* Detachable mounting holes

The sensor’s extensive datasheet is available on: <https://cdn.sparkfun.com/assets/learn_tutorials/5/5/0/MPU9250REV1.0.pdf>

Register map for the sensor: <https://cdn.sparkfun.com/assets/learn_tutorials/5/5/0/MPU-9250-Register-Map.pdf>

All the above and more information about the sensor can be found on: <https://www.sparkfun.com/products/13762>

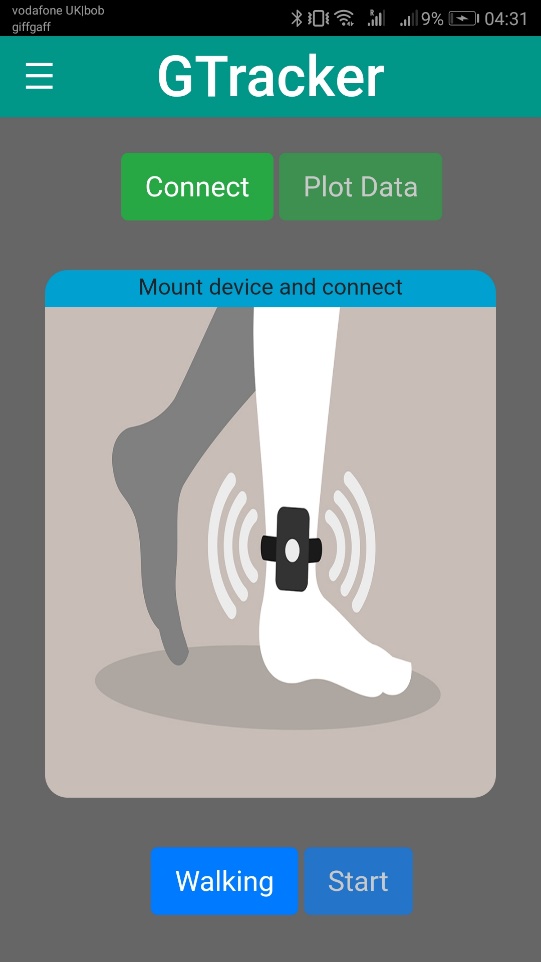
500 mAh LiPo Battery Pack:

* 3.7 nominal voltage
* 100m JST terminated lead
* Comes 30% pre-charged for longer life storage
* 4.2V charging cut-off and 2.4 emergency discharge cut-off
* Dimensions: 37mm x 30.5mm x 5.3mm

The above and more information can be found on the site of the distributor: <https://shop.pimoroni.com/products/lipo-battery-pack?variant=20429082055>

# Appendix D – User Guide

When the application is started the following screen will show up



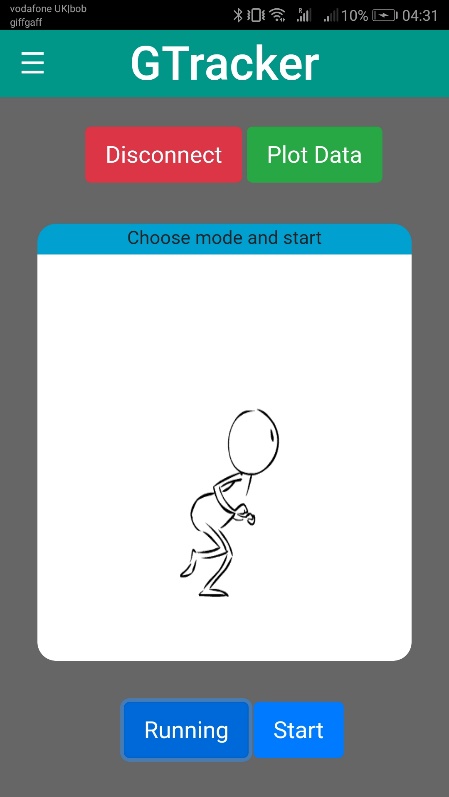
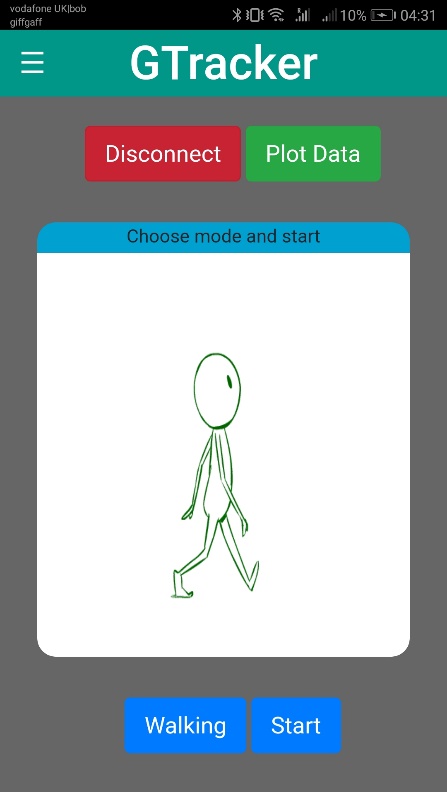
Follow the instructions and mount the device on the outer side of the right leg with the small arcs upwards.

The next step is to press connect and wait for the connection to happen

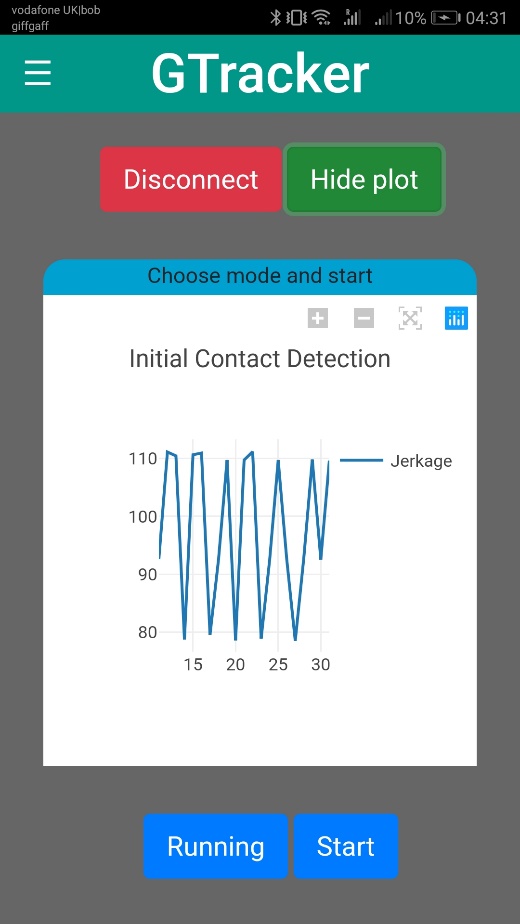


You will see a message stating that a connection is trying to be established. After which a popup will come up stating a connection is successful.

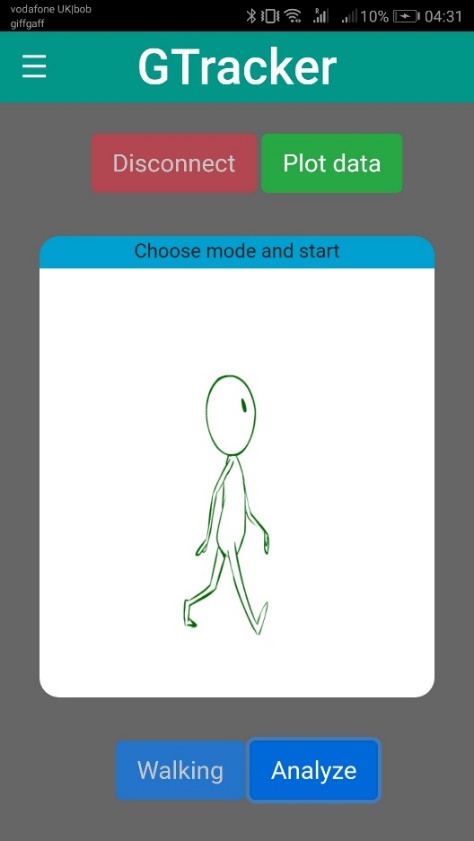
Choose if you will be walking or running. A gif will help you establish what is the current chosen type

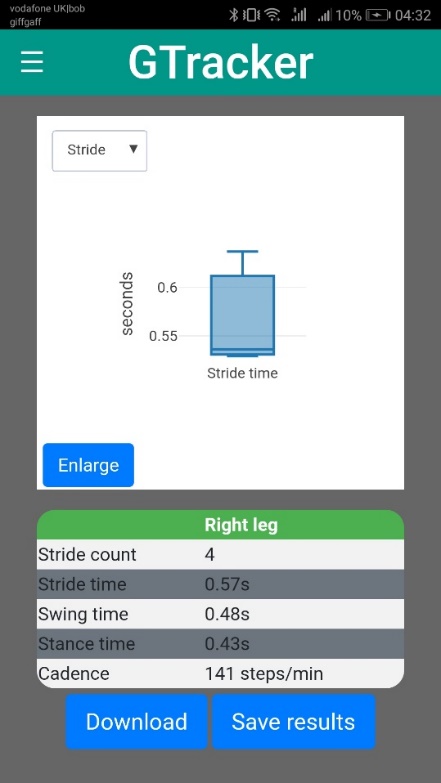


There is an opportunity presented to view the incoming data plotted live

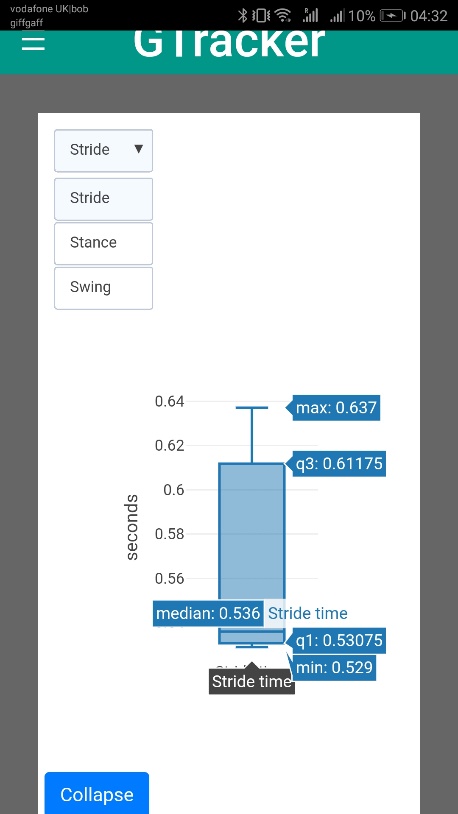
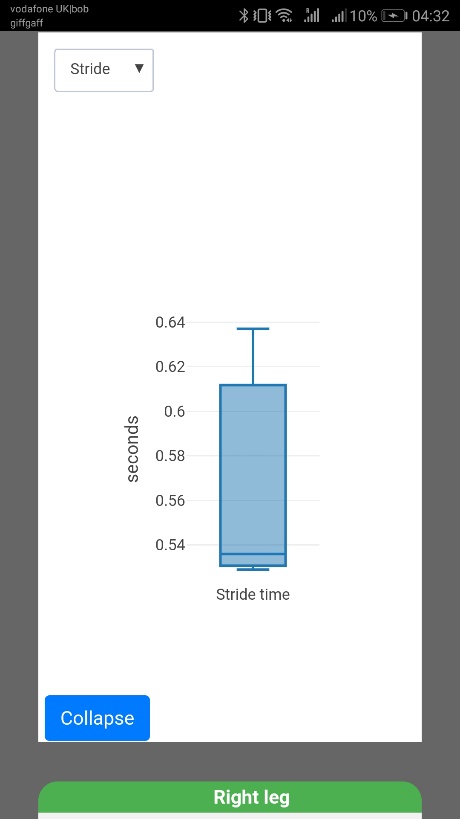


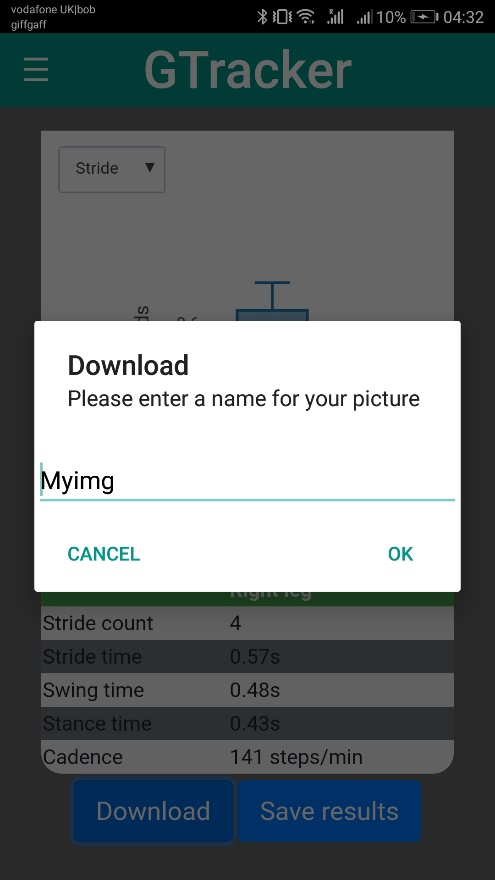
After you have chosen either a walk or run click start do you run/walk and click analyze

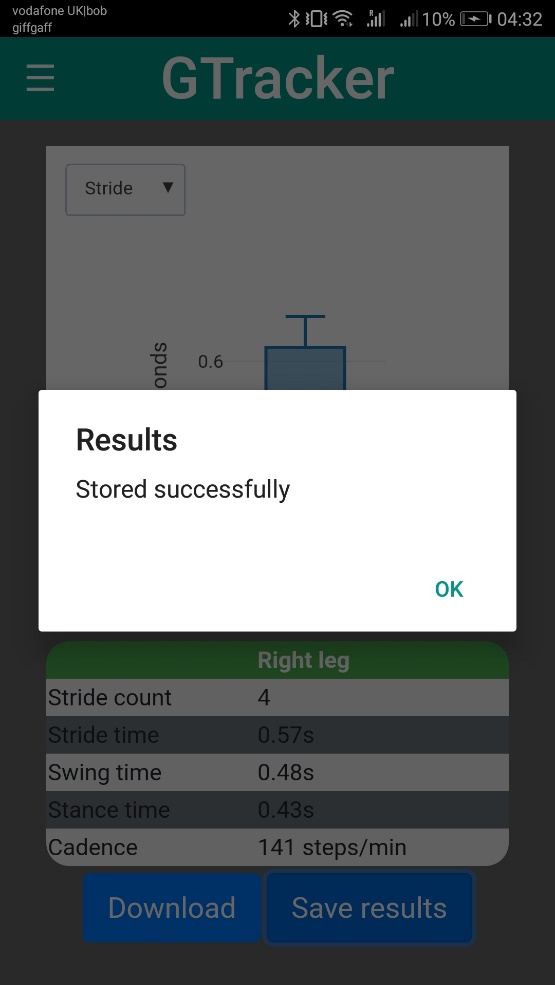


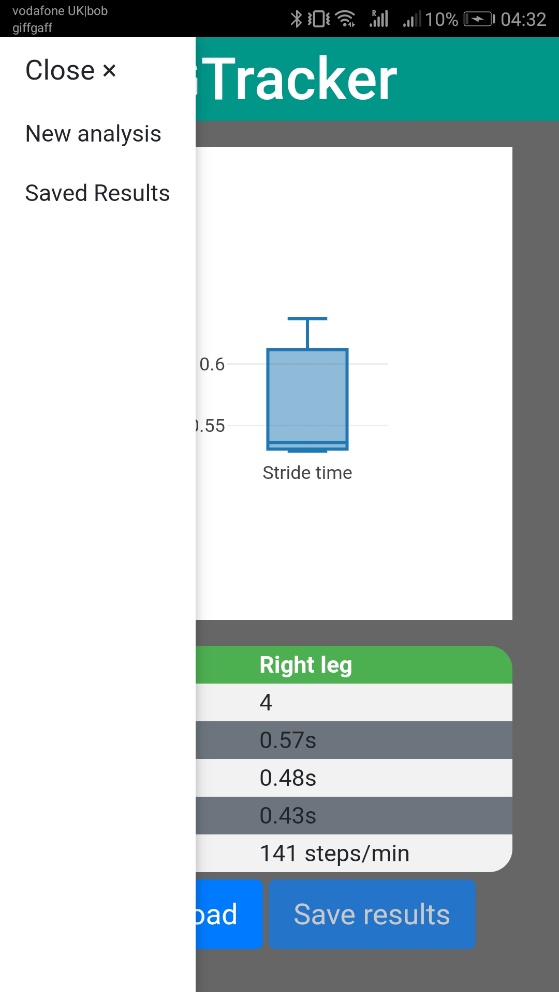
Another page will be displayed showing the analysis

You can enlarge the displayed plot and click on it in for the data to be displayed. You can   
also choose which parameter to be displayed from the dropdown menu.

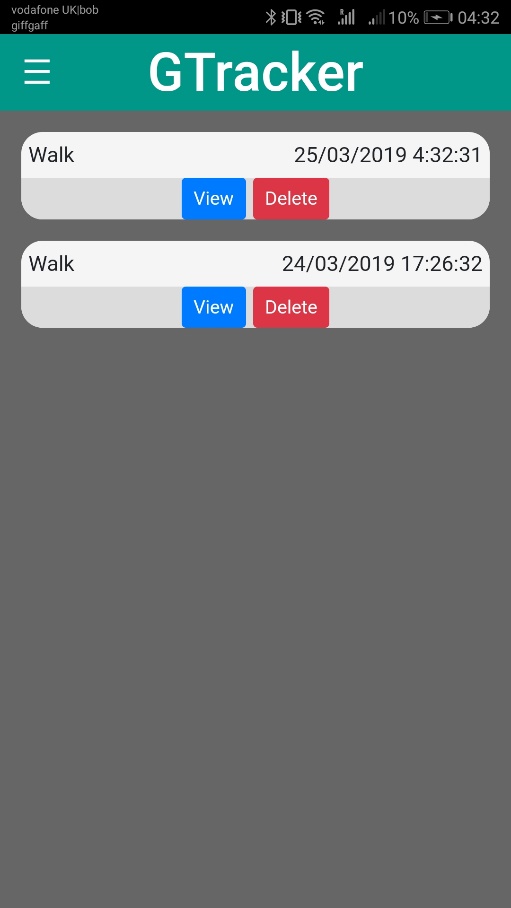


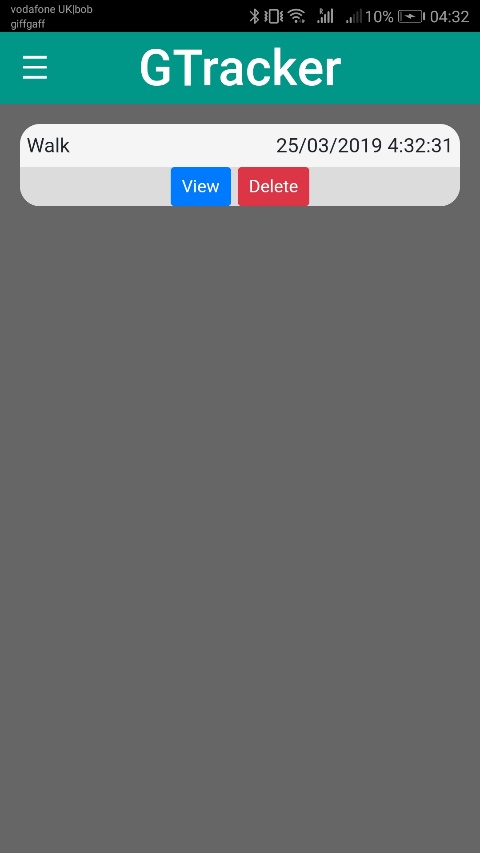
There is an option to download the plot in your Download folder in the Gallery. The popup will prompt you to enter the desired name of the picture.

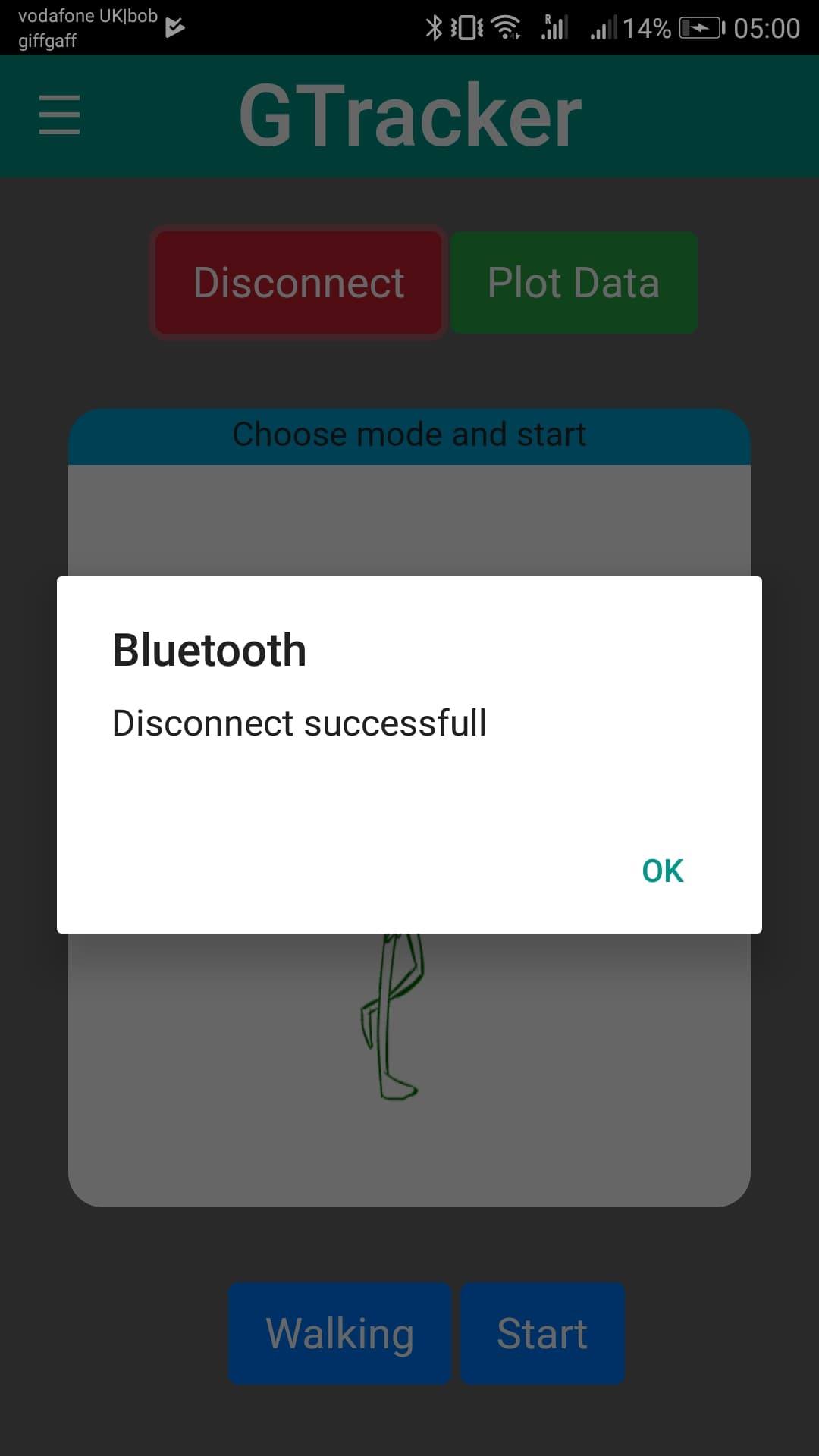
The other thing you can do is to store the results on the phone

From the navigation menu located on the top left you can either do a new analysis or view the saved results

The stored results page shows the results in chronological order



You can choose either to delete an entry or view it which will redirect you to the analysis page.

One more thing you can do is disconnect manually when connected.

# Appendix E – Detailed Evaluation Results

Manually recorded test results are represented in the following tables

Stride/Walk

|  |  |  |
| --- | --- | --- |
| Test1 | Test 2 | Test 3 |
| 1.30 | 1.19 | 1.18 |
| 1.15 | 1.36 | 1.23 |
| 1.13 | 1.01 | 1.03 |
| 1.26 | 1.24 | 1.21 |
| 1.16 | 1.13 | 1.18 |
| 1.13 | 1.15 | 1.20 |
| 1.18 | 1.23 | 1.11 |
| 1.18 | 1.26 | 1.18 |
| 1.14 | 1.14 | 1.18 |
| 1.26 | 1.23 | 1.19 |
| 1.13 | 1.26 | 1.19 |
| 1.25 | 1.03 | 1.15 |
| 1.11 | 1.18 | 1.13 |
| 1.29 | 1.19 | 1.28 |
| 1.13 | 0.9 | 1.26 |

Swing/Walk

|  |  |  |
| --- | --- | --- |
| Test 1 | Test 2 | Test 3 |
| 0.63 | 0.68 | 0.59 |
| 0.45 | 0.55 | 0.59 |
| 0.33 | 0.53 | 0.43 |
| 0.5 | 0.55 | 0.51 |
| 0.42 | 0.45 | 0.57 |
| 0.58 | 0.41 | 0.39 |
| 0.55 | 0.53 | 0.6 |
| 0.53 | 0.61 | 0.5 |
| 0.43 | 0.56 | 0.52 |
| 0.55 | 0.53 | 0.46 |
| 0.52 | 0.57 | 0.46 |
| 0.51 | 0.48 | 0.52 |
| 0.41 | 0.52 | 0.43 |
| 0.53 | 0.67 | 0.58 |

Stance/Walk

|  |  |  |
| --- | --- | --- |
| Test 1 | Test 2 | Test 3 |
| 0.8 | 0.54 | 0.52 |
| 0.68 | 0.75 | 0.74 |
| 0.76 | 0.74 | 0.67 |
| 0.78 | 0.61 | 0.62 |
| 0.7 | 0.68 | 0.75 |
| 0.56 | 0.69 | 0.69 |
| 0.67 | 0.7 | 0.56 |
| 0.72 | 0.55 | 0.63 |
| 0.66 | 0.67 | 0.65 |
| 0.68 | 0.67 | 0.76 |
| 0.65 | 0.71 | 0.637 |
| 0.75 | 0.65 | 0.72 |
| 0.78 | 0.7 | 0.76 |
| 0.62 | 0.56 | 0.65 |

Stride/Run

|  |  |  |
| --- | --- | --- |
| Test 1 | Test 2 | Test 3 |
| 0.91 | 0.72 | 0.78 |
| 0.88 | 0.79 | 0.8 |
| 0.85 | 0.85 | 0.74 |
| 0.85 | 0.85 | 0.79 |
| 0.85 | 0.9 | 0.81 |
| 0.92 | 0.89 | 0.76 |
| 0.93 | 0.81 | 0.77 |
| 0.98 | 0.81 | 0.8 |
| 0.85 | 0.84 | 0.83 |
| 0.8 | 0.87 | 0.8 |
| 0.85 | 0.85 | 0.86 |
| 0.91 | 0.86 | 0.78 |
| 0.79 | 0.86 | 0.8 |
| 0.97 | 0.85 | 0.86 |

Swing/Run

|  |  |  |
| --- | --- | --- |
| Test 1 | Test 2 | Test 3 |
| 0.53 | 0.45 | 0.43 |
| 0.45 | 0.4 | 0.51 |
| 0.48 | 0.45 | 0.48 |
| 0.48 | 0.46 | 0.45 |
| 0.46 | 0.42 | 0.44 |
| 0.43 | 0.48 | 0.55 |
| 0.43 | 0.45 | 0.65 |
| 0.4 | 0.41 | 0.48 |
| 0.5 | 0.5 | 0.45 |
| 0.55 | 0.45 | 0.46 |
| 0.45 | 0.42 | 0.45 |
| 0.48 | 0.47 | 0.5 |
| 0.57 | 0.48 | 0.43 |
| 0.5 | 0.47 | 0.43 |

Stance/Run

|  |  |  |
| --- | --- | --- |
| Test 1 | Test 2 | Test 3 |
| 0.19 | 0.41 | 0.23 |
| 0.33 | 0.42 | 0.29 |
| 0.33 | 0.41 | 0.32 |
| 0.38 | 0.38 | 0.36 |
| 0.38 | 0.35 | 0.33 |
| 0.46 | 0.43 | 0.31 |
| 0.41 | 0.37 | 0.13 |
| 0.48 | 0.42 | 0.28 |
| 0.45 | 0.45 | 0.41 |
| 0.36 | 0.4 | 0.3 |
| 0.28 | 0.36 | 0.3 |
| 0.44 | 0.39 | 0.36 |
| 0.37 | 0.37 | 0.36 |
| 0.26 | 0.36 | 0.48 |