Footstrike pattern determination

Dewaldt Snyman
22706852

Report submitted in partial fulfilment of the requirements of the module Project (E) 448 for the degree Baccalaureus in Engineering in the Department of Electrical and Electronic Engineering at Stellenbosch University.

Supervisor: Dr W. A. Smit

October 2099

Acknowledgements

I would like to thank my dog, Muffin. I also would like to thank the inventor of the incubator; without him/her, I would not be here. Finally, I would like to thank Dr Herman Kamper for this amazing report template.

UNIVERSITEIT·STELLENBOSCH·UNIVERSITY
jou kennisvennoot • your knowledge partner

Plagiaatverklaring / *Plagiarism Declaration*

1. Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is.
   Plagiarism is the use of ideas, material and other intellectual property of another's work and to present is as my own.

2. Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is.
   I agree that plagiarism is a punishable offence because it constitutes theft.

3. Ek verstaan ook dat direkte vertalings plagiaat is.
   I also understand that direct translations are plagiarism.

4. Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelikse aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is.
   Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism

5. Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie.
   I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

22706852 Studentenommer / *Student number*	DN Snyman Handtekening / *Signature*
DN Snyman Voorletters en van / *Initials and surname*	October 4, 2022 Datum / *Date*

Abstract

English

The English abstract.

Contents

List of Figures

List of Tables

Nomenclature

Variables and functions

$p(x)$	Probability density function with respect to variable x.
$P(A)$	Probability of event A occurring.
ε	The Bayes error.
ε_u	The Bhattacharyya bound.
B	The Bhattacharyya distance.
s	An HMM state. A subscript is used to refer to a particular state, e.g. s_i refers to the i^{th} state of an HMM.
$\mathbf{S}$	A set of HMM states.
$\mathbf{F}$	A set of frames.
$\mathbf{o}_f$	Observation (feature) vector associated with frame f.
$\gamma_s(\mathbf{o}_f)$	A posteriori probability of the observation vector $\mathbf{o}_f$ being generated by HMM state s.
μ	Statistical mean vector.
Σ	Statistical covariance matrix.
$L(\mathbf{S})$	Log likelihood of the set of HMM states $\mathbf{S}$ generating the training set observation vectors assigned to the states in that set.
$\mathcal{N}(\mathbf{x}\|\mu, \Sigma)$	Multivariate Gaussian PDF with mean μ and covariance matrix Σ.
a_{ij}	The probability of a transition from HMM state s_i to state s_j.
N	Total number of frames or number of tokens, depending on the context.
D	Number of deletion errors.
I	Number of insertion errors.
S	Number of substitution errors.

Acronyms and abbreviations

AE	Afrikaans English
AID	accent identification
ASR	automatic speech recognition
AST	African Speech Technology
CE	Cape Flats English
DCD	dialect-context-dependent
DNN	deep neural network
G2P	grapheme-to-phoneme
GMM	Gaussian mixture model
HMM	hidden Markov model
HTK	Hidden Markov Model Toolkit
IE	Indian South African English
IPA	International Phonetic Alphabet
LM	language model
LMS	language model scaling factor
MFCC	Mel-frequency cepstral coefficient
MLLR	maximum likelihood linear regression
OOV	out-of-vocabulary
PD	pronunciation dictionary
PDF	probability density function
SAE	South African English
SAMPA	Speech Assessment Methods Phonetic Alphabet

Chapter 1

Introduction

In the modern day, jogging is one of the most popular physical activities. People all over the world partake in this physical activity. One would think that jogging is a simple exercise and there are no major injury or health concerns. However, there are some major concerns as jogging has a high injury rate. Over the past few years, there has been an increasing interest in this subject and many studies have been conducted on how these injuries can be analyzed. Strike patterns during running has been acknowledged as potential way of identifying injuries or risk of injury.

1.1. Problem Statement

There are not many commercial devices that can determine a user's foot strike patterns.

1.2. Objectives

The objective of this project is to use a prototype wearable device that captures the pressure applied to a grid of pressure sensing resistors. The prototype has been build by a previous student and is capable of capturing the data. The device has capabilities of transmitting the data wirelessly to a mobile phone. The main object is then to use the data captured and transmit it to a smartphone. An android application will then be designed and development that can process and display this data in a useful manner. The application can then be used by medical experts to assist in medical assessments like the Gait analysis.

1.3. Scope and Limitations

1.4. Overview of the report

1.5. Section heading

This is some section with two table in it: Table 1.1 and Table 1.2.

Table 1.1: Performance of the unconstrained segmental Bayesian model on TIDigits1 over iterations in which the reference set is refined.

Metric	1	2	3	4	5
WER (%)	35.4	23.5	21.5	21.2	22.9
Average cluster purity (%)	86.5	89.7	89.2	88.5	86.6
Word boundary F-score (%)	70.6	72.2	71.8	70.9	69.4
Clusters covering 90% of data	20	13	13	13	13

Table 1.2: A table with an example of using multiple columns.

	Accuracy (%)		
Model	Intermediate	Output	Bitrate
Baseline	27.5	26.4	116
VQ-VAE	26.0	22.1	190
CatVAE	28.7	24.3	215

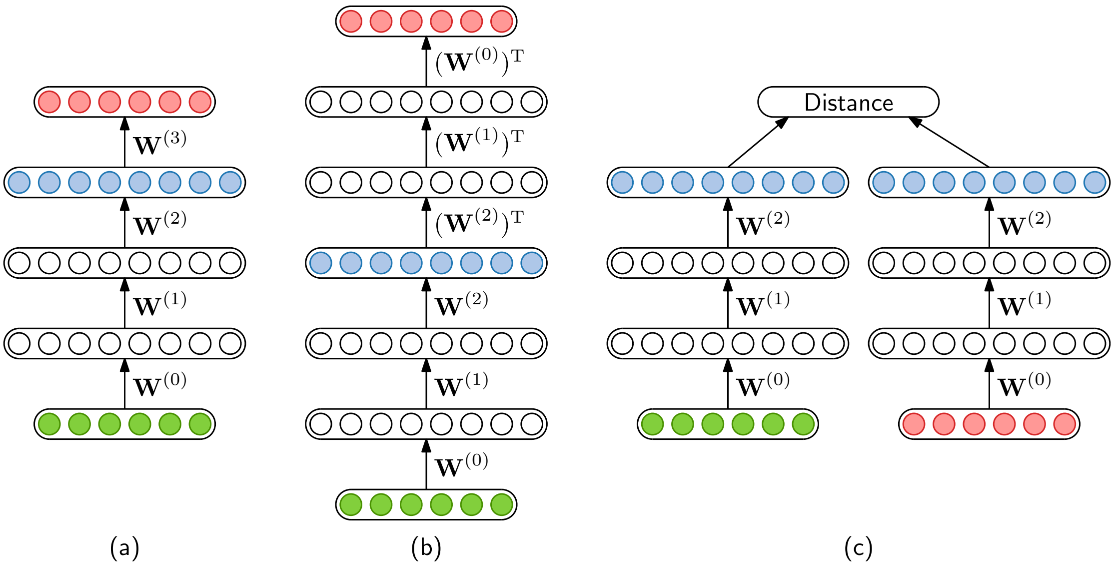

Figure 1.1: (a) The cAE as used in this chapter. The encoding layer (blue) is chosen based on performance on a development set. (b) The cAE with symmetrical tied weights. The encoding from the middle layer (blue) is always used. (c) The siamese DNN. The cosine distance between aligned frames (green and red) is either minimized or maximized depending on whether the frames belong to the same (discovered) word or not. A cAE can be seen as a type of DNN [1].

This is a new page, showing what the page headings looks like, and showing how to refer to a figure like Figure 1.1.

The following is an example of an equation:

$$P(\mathbf{z}|\boldsymbol{\alpha}) = \int_{\boldsymbol{\pi}} P(\mathbf{z}|\boldsymbol{\pi})\, p(\boldsymbol{\pi}|\boldsymbol{\alpha})\, \mathrm{d}\boldsymbol{\pi} = \int_{\boldsymbol{\pi}} \prod_{k=1}^{K} \pi_k^{N_k} \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1}\, \mathrm{d}\boldsymbol{\pi} \tag{1.1}$$

which you can subsequently refer to as (1.1) or Equation 1.1. But make sure to consistently use the one or the other (and not mix the two ways of referring to equations).

Chapter 2

Literature Review

2.1. Foot strike patterns and Gait analysis

2.2. Bluetooth Low Energy BLE

2.2.1. How Does BLE Work?

When using Bluetooth Low Energy it is important to know the roles of each device. In all BLE application there are two roles which are the **central** and the **peripheral** devices. The peripheral device will be the device that broadcasts or advertises information and the central device will be scanning for information. A good visual representation of how BLE works is to think of an advertising board where the peripheral device keeps pinning new info onto the board and the central device scans the board and uses the information available. These two devices have their own unique addresses. The peripheral will be advertising information to any near devices while the central will for any device or devices that are advertising information. When the central device finds the advertised information a connection attempt is made. Once a connection is established, the central device can start read and writing information from and to the peripheral device.

2.2.2. Services, Characteristics and Descriptor

The information transfer system for BLE can be seen in the following hierarchical order. A service is a collection of characteristics and each characteristic has a descriptor that describes the characteristic. See the basic illustration below 2.1.

Service		
Characteristic 1	Characteristic 2	Characteristic 3
Descriptor	Descriptor	Descriptor

Figure 2.1: Basic hierarchy of BLE

Each **service** has its own unique identifying code called a UUID. This is to allow one peripheral device to have multiple services. This code can be 128-bit long for each service. A service can also be seen as a group of capabilities. For example, a smartwatch that can measure hart rate, temperature and track your GPS location. These three capabilities can be grouped under one service and can be called the activity service. This method of grouping information allows the central device to better understand and use the information that the peripheral is advertising.

The capabilities mentioned in the above example are better known as **characteristics**. Each characteristic has its own unique identifying code called a UUID. This is to allow one service to have multiple characteristics. A characteristic can be seen as a single capability. For example, the characteristic of measuring heart rate. This characteristic can be seen as a single capability of the activity service.

Each characteristic can have a **descriptor** that describes the characteristic. A descriptor can be seen as a single piece of information about the characteristic. For example, the descriptor of the characteristic of measuring heart rate can be the range of the heart rate measurement. This descriptor can be seen as a single piece of information about the characteristic of measuring heart rate.

2.3. Open GLES for android

OpenGL is an open source, graphics library for high performance 2D and 3D graphics rendering. OpenGL—ES is flavor of OpenGL specifically intended for embedded and mobile devices. OpenGL—ES is a cross-plaftorm high performance graphics API that can be used by Android devices. Android supports both the framework API and the Native Development Kit (NDK) of OpenGL.

Chapter 3

System Design

3.1. System overview

The prototype foot sensor device was designed and built by Jared Adams who is the predecessor of this skripsie topic. The LiFePO4 battery supplies power to all the components of the device. These components are an Arduino Nano 33 Ble, an ADS115 ADC module and the IEE foot sensor. The battery has a nominal voltage of 3.2VDC and is charged by a battery charge management controller namely, the MCP73123.

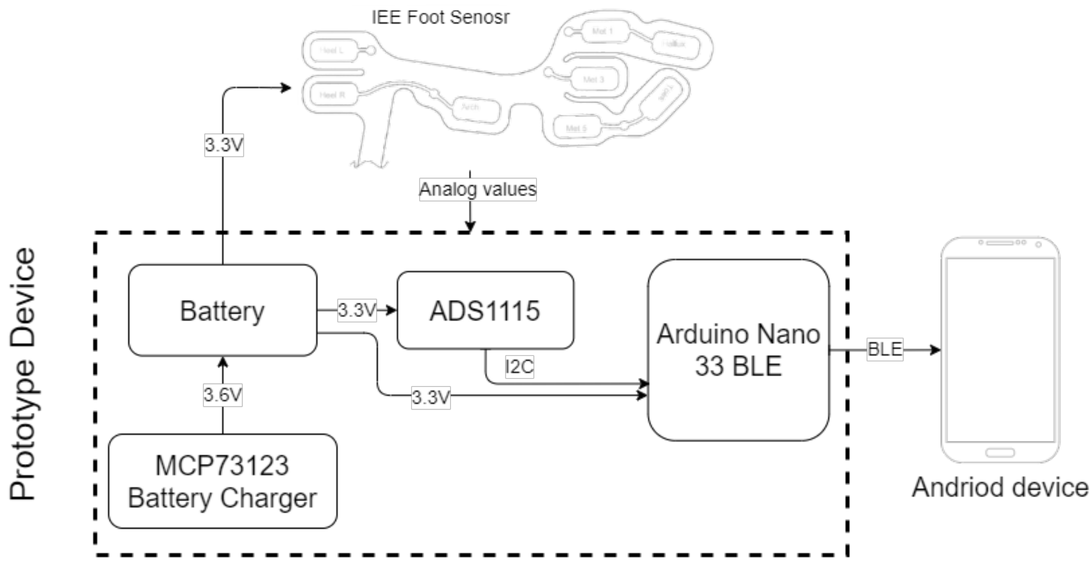

Figure 3.1: System Diagram

3.2. Device components

3.2.1. Arduino Nano 33 BLE

The Nordic nRF52840 SoC processor features a configurable 12-bit, 200 ksps, 8channel ADC. The ADC has a configurable internal analog reference voltage.

Analog inputs have programmable internal pull-up resistors.

The NINA-B306 is a Bluetooth 5 low energy module that allows for low power consumption when sending data over Bluetooth wireless standard.

Absolute supply voltage ratings of

3.2.2. LiFePO4 Battery

A lithium iron phosphate battery is a type of lithium iron battery that uses lithium iron phosphate as its cathode material. The LiFePO4 battery has a maximum charge cut-off of 3.65V and a minimum charge cut-off of 2.5V. LiFePO4 batteries importantly feature a flat discharge curve with a nominal output voltage of 3.2V, as seen in Figure 3.4. This provides a stable supply voltage, VBAT, for the system and a stable reference voltage, Vref, for the ADCs.

3.2.3. IEE Smart Footwear Sensor

The IEE Smart Footwear Sensor (Section 2.2.1) consists of 8 separate high-dynamic FSR cells which each measure a pressure range of 100 mbar up to 7 bar. The foot sensor has 11 pins1 - 1 supply voltage pin, 2 ground pins and 8 output pins. The structure of the foot sensor is essentially a resistive divider network with a fixed resistor, Rfix, made from conductive ink with a value in range of 2 kΩ ¡ Rfix ¡ 4 kΩ . The foot sensor is supplied with VBAT and the 8 outputs are connected to the Arduino and DS1115 modules. Each individual sensor has a typical activation response time of 2-3ms and can measure a wide pressure range which make it more than sufficient for the desired application.

3.3. App components

Chapter 4

Detailed Design

4.1. Software

4.1.1. Arduino Code

Setup

When using Arduino devices there will be a setup() method which will run once only when the device starts. The Serial communications and baud-rate are specified with the Serial.begin() function. Thereafter the Arduino pin modes are configured to activate the internal pull up as this is needed for measuring the analog values from the IEE foot sensor. The pin mode can be configured using the pinMode(). This function takes two arguments which is the pin number and the mode. To use the ADC readings send by the ADS1115 via I2C, An object of the Adafruit_ADS1115 class was created and called ads. Now within the setup() method ads.begin method can be triggered to do all the necessary I2C setup for the ADS1115.

Next, the BLE setup commences. Arduino has a good library namely, ArduinoBLE.h, which provides all the needed function to setup BLE for the Arduino device. See code snippet 4.1

```
  // BLE setup
  BLE.setLocalName("Arduino Nano 33 BLE (Peripheral)");
  BLE.setAdvertisedService(gaitService);
  gaitService.addCharacteristic(gaitCharacteristic1);
  gaitService.addCharacteristic(gaitCharacteristic2);
  BLE.addService(gaitService);
  BLE.advertise();
  if (!BLE.begin()) {
    Serial.println("BLE error-Ble could not start");
    while (1);
  }
```

Listing 4.1: BLE Setup

The ”gaitService”, ”gaitCharacteristic1” and ”gaitCharacteristic2” object, used in code snippet 4.1, are created outside the setup() function. The ArduinoBLE library provides the

BLEService and BLECharacteristic classes to create these objects. The BLEService class only requires the service UUID as an argument to create an object. The BLECharacteristic class requires the characteristic UUID, the properties of the characteristic and the size of the value that the characteristic will represent. The properties can be specified by doing or operation with predefined constants provided by the ArduinoBLE library.

ADC

Unfortunately the Arduino NANO 33 only has 6 ADC pins and therefor an external ADC module, the ADS1115, was used to send the other 2 ADC values across I2C. The Adafruit library has a class with a method called readADCSingleEnded(). This method is used to get the ADC reading from the ADS1115. This method requires the pin number of the ADS1115 as a property. The ADS1115 has a 16 bit resolution and therefor it is required to use the map() function to scale the 16 bit ADC value down to 12 bit. The remaining 6 ADC readings can be retrieved using the analogRead() function and passing the pin number as an argument.

All the ADC operations that need to happen are within the getReadings() function. This function stores 8 analog readings in 8 floats which is then copied across two byte arrays each having a length of 16 bytes. The is to minimize the use of characteristics to only two characteristics.

BLE

In the main loop() function the Arduino will continuesly wait for a connection from a central device. An if statement has a BLEDevice object as condition and the condition will be true as soon as an connection is established. Within the if statement is a while loop which will loop while the central device is connected. Inside the while loop the getReadings() function is called to get all the ADC readins then these readings are written to each characteristic with the writeValue method from the BLECharacteristic class objects that were created. These characteristics are now advertising data to the central device

4.1.2. Android application code

4.1.3. Overview of user interface

4.2. Hardware

Chapter 5

Summary and Conclusion

Bibliography

[1] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 1, pp. 30–42, 2012.

Appendix A

Project Planning Schedule

This is an appendix.

Appendix B

Outcomes Compliance

This is another appendix.