Delft University of Technology
Master's Thesis in Embedded Systems

Handshake Recognition

Applied to Wireless Data Exchange in Smartbands

Aimee Ferouge

embedded software

TU Delft Delft University of Technology

Handshake Recognition

Applied to Wireless Data Exchange in Smartbands

Master's Thesis in Embedded Systems

Embedded Software Section
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Aimee Ferouge
A.A.Ferouge@student.tudelft.nl

9th December 2015

Document generated with LaTeX and TeXnicCenter.

Delft, 9th December 2015

Author
Aimee Ferouge (A.A.Ferouge@student.tudelft.nl)
Title
Handshake Recognition
Applied to Wireless Data Exchange in Smartbands
MSc presentation
November 26, 2015

Graduation Committee

Prof. dr. K. G. Langendoen	Delft University of Technology
M. A. Zuñiga Zamalloa	Delft University of Technology
dr. H. S. Hung	Delft University of Technology

Embedded Software Group
Department of Software Technology
Faculty EEMCS, Delft University of Technology
Delft, The Netherlands
www.es.ewi.tudelft.nl

Shake-On
Molengraaffsingel 12-14, 2629 JD
Delft
www.shake-on.com

Preface

This Master thesis is the final part of my Master of Science in Embedded Systems at Delft University of Technology. The work presented is performed at the Embedded Software Group led by Prof. dr. Koen Langendoen. It describes the research I conducted while developing a smart bracelet for Shake-On, a startup at the YES!Delft incubator.

My work combines Shake-On's need for a product with a more theoretical view. This report is written for the thesis committee and future developers of Shake-On. It will convey my ideas and provide recommendations with respect to further product development.

For questions or remarks, do not hesitate to contact me.

Aimee Ferouge
A.A.Ferouge@student.tudelft.nl
Delft, 9th December 2015

Acknowledgments

My daily supervisor for the past nine months has been Marco Zuñiga, to whom I would like to express my gratitude. Thanks to Shake-On for providing me with a both challenging and tangible research project. I would also thank Shake-On for the experience to work in a startup and the way you have completely taken me up in the team.

I would like to thank my fellow ES students from 'The 9th Floor' for their ideas and feedback. After nine months of work I would like to thank my roommates for volunteering in the experiments, understanding my absent-mindedness from time to time and for a fair amount of distraction. Finally many thanks to my family, for supporting me throughout my entire studies.

Abstract

The way people meet each other is usually face to face. Meanwhile, the way people maintain their contacts is mostly through social media. This results in a gap of translating a handshake into a digital connection. Shake-On is a start-up that has come up with an answer by means of a smart bracelet. Their aim is to wirelessly exchange contact details between users wearing the bracelet. This exchange is triggered by the most common human gesture people use when introducing themselves: the handshake.

This thesis will overcome two major challenges for Shake-On. First, no general pattern recognition method can be applied to detect handshakes. This is caused by the fact that handshakes gestures show large variations among individual persons.

Second, the system should be robust to multiple handshakes happening concurrently. This applies to the scenario of more than two people shaking hands while standing close to each other. Contact details should only be exchanged between people that are handshaking, which requires handshake matching. Again, large variations in 'handshaking style' make it a cumbersome task to identify matching handshakes.

This thesis proposes a two-fold solution to address the above-mentioned challenges. The first part includes ***handshake detection****, using new features for pattern recognition that are tailored to handshaking recognizing. The second part proposes a new method to perform* ***handshake matching*** *that overcomes the shortcomings of existing solutions.*

The work done in this thesis has led to the following results:

1. The developed **detection** method takes into account limited resources and is therefore suitable for implementation on a smart bracelet. Moreover, it shows similar performance as the state-of-the-art solutions, namely an accuracy of 95%. In contrast to existing solutions using 6 stochastic features, our solution uses 4 computationally lightweight features.

2. Being the first of its kind, the **matching** method proposes a novel technique that maps handshakes to an abstract binary format. This format is called *peakmaps*. Because it eliminates personal handshaking style, peakmaps result in a pairing accuracy of 80% compared to 24% using basic cross correlation.

Contents

List of Acronyms

MEMS	Microelectromechanical Sytems
SoC	System-on-Chip
HMM	Hidden Markov Model
SVM	Support Vector Machine
kNN	k-Nearest Neighbor
RF	Radio Frequency
DTW	Dynamic Time Warping
BLE	Bluetooth Low Energy

Chapter 1

Introduction

"It is possible to tell things by a handshake. I like the 'looking in the eye' syndrome. It conveys interest. I like the firm, though not bone crushing shake. The bone crusher is trying too hard to 'macho it.' The clammy or diffident handshake, fairly or unfairly, get me off to a bad start with a person."

— George H. W. Bush

1.1 Getting in Touch, Staying in Touch

Handshakes have been around for over 2,500 years, used by the ancient Greeks to reassure one another they held no weapon[1]. Nowadays, the gesture is common in the Western world when greeting people or meeting somebody new.

The most common way people meet each other is face to face, either by coincidence or on purpose. The preference for *getting in touch* with new people face to face is provided by conferences and networking events.

Meanwhile, the way people maintain in contact has changed with the introduction of social media. In a world where social media is rapidly evolving, many social platforms are arising for different types of purposes. Examples include Facebook for self-promotion or LinkedIn for business purposes[2]. Nowadays when it comes to *staying in touch*, the digital world is sufficient.

If we look at professional networking, this phenomenon is very well-known. Conferences and networking events are common practise for meeting new people. Many business cards are exchanged, only to be used when looking for the person on LinkedIn. For our scenario of interest, the process of meeting new people comprises two steps:

1. Getting in touch face to face
2. Staying in touch digitally

The question remains how to fill the gap between the encounter of new contacts in real-life and the emerging digital platforms for the maintenance of social networks.
In other words, how to translate a handshake to a digital connection?

1.2 Introducing the Shake-On Smart Bracelet

The issue stated above is addressed by Shake-On, a start-up at the YES!Delft incubator. Their idea is to create a smart wrist-band[1] that can be worn during events where people meet, e.g. conferences or networking events. Upon detecting a handshake between two people, the smartbands exchange contact information users by means of a smartphone application. This contact information contains only the basics, for example name, function and company. Users get an overview of recently collected handshakes and can decide whether to keep or discard the contacts. Upon mutual confirmation of both users, complete profiles are exchanged. These may include a phone number, email address or a link to their LinkedIn page.

The bracelet is responsible for the detection of handshakes and the wireless exchange of basic contact information.

[1]Smart wrist-band, (smart) bracelet and smartband are equivalent terms

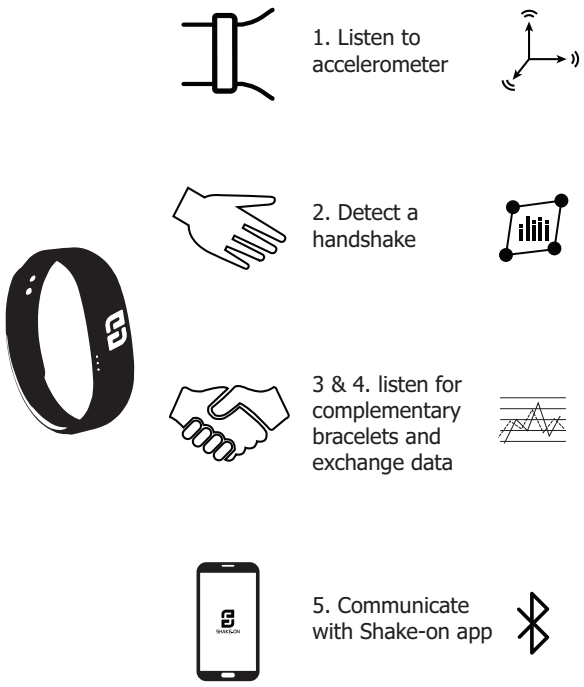


Figure 1.1: Actions performed by the bracelet when the user performs a handshake.

1.3 System Set Up and Requirements

Before summarizing the contents of this thesis, the functions of the bracelet are briefly explained. See also Fig. 1.1. Bracelets worn by each individual attendee follow this lifecycle:

1. **Idle mode:** bracelets keep track of the user's movements
2. **Detect handshake:** when the user performs a handshaking gesture, this is detected by the bracelet
3. **Communicate:** the bracelet broadcasts information about the new handshake and listens for incoming 'handshakes' from neighbouring bracelets
4. **Match handshake:** contact details will be exchanged between bracelets of two people shaking hands
5. **Notify:** The bracelet notifies the user of the new match by means of a smartphone application using Bluetooth Low Energy (BLE)

The first requirement is that the bracelet should be energy efficient and able to last at least three days without recharging. As large events usually take up a couple of days (or sometimes an entire week), the need for daily recharging of all bracelets should be avoided.

Power consumption is an important aspect that should always be taken into account thoughout the entire development.

Second, users should be offered real-time insight about who they have met during the event. Upon handshake detection by a bracelet, the identity (ID) of the matching bracelet should be passed on to a smartphone application[2]. This introduces the optional fifth step to the lifecycle.

1.4 Problem Statement

There are a few design constraints that originate from Shake-On's requirements. These introduce a couple of challenges, as they require algorithms and mechanisms that are not readily available.

First, the bracelet should be able to detect a traditional handshake gesture. The bracelet should not require a so-called invented gesture, which is a predefined gesture that is performed similarly by all users in order to increase detection performance. Instead, the user's natural handshake is used as a trigger to exchange contact details. This way, people don't have to act differently as a consequence of wearing the bracelet. The problem is that no handshake is the same, since they show large variability between individuals. Variable traits of a handshake include the duration of a handshake, the intensity of shaking and orientation of the grip.

Given the human factor in each handshake, there is no straightforward and general detection algorithm that can be applied.

Second, the handshake detection should be able to handle large crowds, i.e. more than two people shaking hands simultanuously in a small room. The matching step mentioned in the previous paragraph comprises selecting the right bracelet from a pool of candidates. When four attendees (pair A and pair B) are shaking hands, contact details should only be exchanged within pair A and pair B. The bracelet should compare candidate handshakes to its own decide whether they are its complementary or not. Sometimes one person will have a strong, powerful handshake and the other person will have a soft, timid handshake. For this reason, finding the right match can be cumbersome if the individual gesture will not show much resemblence.

[2]The smartphone application will be referred to as 'the app' in the remainder of this thesis.

Thus, the matching step introduces the challenge of selecting the matching bracelet from a pool of surrounding candidates, accounting for handshake uniqueness.

To summarize, the following challenges are introduced:

1. **Generality**: there is no straightforward solution for handshake detection, since every user has a unique 'handshaking style'
2. **Robust to concurrency**: in the case of multiple candidate matches, the bracelet should be able to select the right match

1.5 Contributions

The contributions of this thesis are two-fold. First, this thesis provides a comparative study of different technologies required to realize the bracelet Shake-On is aiming for, meeting all of its constraints. Second, algorithms are proposed to perform handshake detection and matching. Throughout the thesis, these contributions are described in the following manner:

- Chapter 2 explains the system architecture from which a requirements specification will be derived;
- Chapter 3 contains an analysis of related projects regarding wireless data exchange and gesture recognition;
- Chapter 4 proposes new features for pattern recognition that are tailored to handshaking gestures. This results in a detection accuracy of 95%, using fewer features than state-of-the-art solutions;
- Chapter 5 proposes a new method for handshake matching, called *peakmaps*. These will increase the pairing accuracy from 24% for basic cross correlation to 80%.

Finally, Chapter 6 draws conclusions and gives recommendations for future development.

Chapter 2

Design Requirements

Chapter 1 already touched upon the most important design constraints. This chapter will further elaborate on the requirements of the desired sytem, distinguishing the two main challenges:

- Handshake detection based on accelerometer data
- Handshake matching of two users

This chapter is structured as follows: Section 2.1 will describe the custom PCB that was developed for Shake-On, used as starting point for the bracelet. Section 2.2 explains why it is preferable to pre-classify gestures rather then broadcasting all sensor data. Section 2.3 lists the requirements for handshake detection and pinpoints the challenges that will be faced. Section 2.4 does the same for handshake matching, already giving a few insights why matching is difficult.

2.1 The Hardware

With respect to the hardware, a few design choices were already prescribed by Shake-On. First, the PCB that is used in the smartbands is custom-made and has been a starting point for this project. See Fig. 2.1. The board contains an nRF51 chip[1] , a printed antenna[2] and an accelerometer[3]. The nRF51 chip is built around a Cortex M0 CPU[4] , has an embedded radio

[1]Nordic Semiconductor nRF51822 System-on-Chip [3]

[2]Designed to use the 2.4 GHz band

[3]LIS3DH MEMS digital output motion sensor: ultra low-power high performance 3-axes "nano" accelerometer[4]

[4]32-bit ARM®-Cortex M0

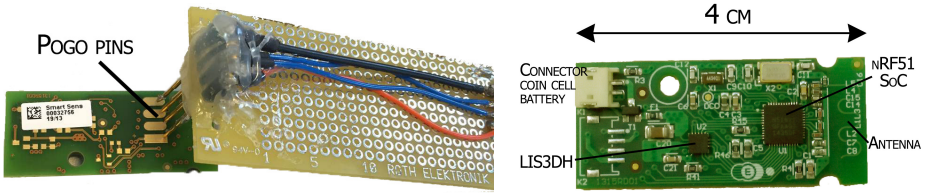


Figure 2.1: **The PCB and programmer from Salland Electronics.** Main components are the nRF SoC, antenna and accelerometer.

transceiver and supports so-called SoftDevices. These are pre-compiled, pre-linked binary files. The SoftDevice S110 is used on the nRF51 as it contains a Bluetooth®Smart protocol stack [5]. To program and debug the PCB, the manufacturer Salland Electronics has made a module (the *programmer*) with Pogo pins[5]. See Fig. 2.1.

2.2 Computation over Transmission

The smartbands continuously receive sensor data from the accelerometer, which can not all be classified as a potential handshake gesture. Knowing this, a filtering step is needed that precedes the transmission of handshake characteristics to other smartbands should. *Computation over Transmission* is a well-known design principle, stating that the energy cost of computation as compared to radio transmission is approximately 1000 calculations to 1 bit of data transfer[6]. Of course, this depends on the distance that the data must be transferred as transmission cost increases by the square of the distance. However, if it is possible to reduce the amount of data to be transmitted by increasing the number of computations this is preferable.

For this reason, a pre-classification algorithm should make sure only potential handshakes are broadcasted. This means on-chip handshake recognition done in two steps: classification of the sensor data (handshake detection) by the individual bracelets, followed by data broadcasting and comparison with other bracelets (handshake matching).

[5]Spring-loaded pins that are used to establish a temporary connection between to PCBs

2.3 Challenge 1: Handshake Detection

As explained in Section 2.2, handshake detection will comprise the classification of the sensor data. A sample can be either labeled 'Handshake' or 'No handshake'. Given the available hardware and the design specifications from Chapter 1, the following requirements apply to the handshake detection algorithm:

1. The algorithm should be able to run without any resources other than the ones offered by the components on the PCB
2. The algorithm should only use the accelerometer data as input
3. The algorithm should be able to detect natural handshake gestures (no need for invented gestures)

The *generality* property described in Chapter 1 makes the formulation of a detection algorithm challenging. It is impossible to train the system for all kinds of handshakes, so our solution needs to be general. On the other hand, the solution should be reliable; our algorithm should decrease the number of false negatives to avoid missing a handshakes. At the same time, the algorithm should minimize the number of false positives to save energy and avoid broadcasting gestures that are not handshakes.

To illustrate the need for a general yet reliable solution, Fig. 2.2 shows sensor data of two people shaking hands. Even though both gestures belong to the same handshake, they show very different traces. Despite this, they should be both recognized by the the detection algorithm.

2.4 Challenge 2: Handshake Matching

The second challenge comprises the comparison of handshake movements between two users wearing the bracelet. Both smartbands provide a sample characterizing their handshake movement, which is then checked for similarity. Given the available hardware and the design specifications from Chapter 1, the requirements for handshake detection also apply to the matching algorithm:

The algorithm should be able to run without any resources other than the ones offered by the components on the PCB

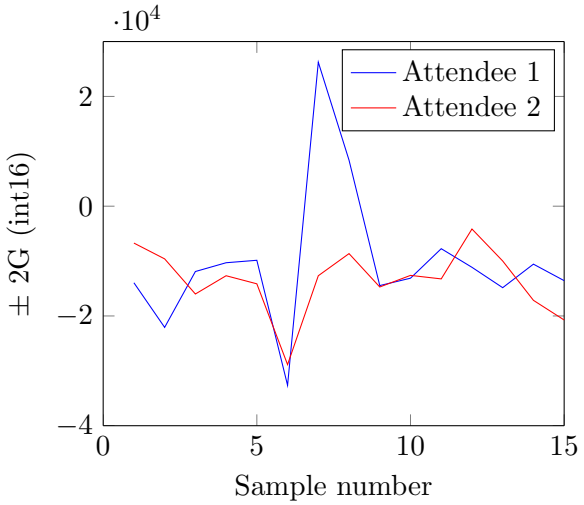


Figure 2.2: **Accelerometer data from two people shaking hands.** Even though both gestures belong to the same handshake, they show very different traces. For handshake detection, this introduces a need for generality. For handshake matching, it is hard to recognize such different traces as being the *same* handshake.

As we will see in Chapter 5, handshake matching is quite complex because traces that belong to the same handshake can be very different. An aspect that needs to be considered is that users might wear their bracelets in a slightly different manner, resulting in different sensor values. Another phenomenon that causes different sensor values is called *palm power*. This is the situation where one person tends to rotate the other person's hand downwards, forcing him in a submissive role. Finally, people vary a lot in the intensity of the handshake. When one person has a rather strong handshake, the other person's hand is more or less being shook. It may seem unlikely, but the two samples from Fig. 2.2 are from the same handshake. Because two individual handshake traces can show such large variability, it is difficult to see whether they belong to the same handshake or not.

Chapter 3

State-of-the-Art

This chapter will explore the research that has already been conducted on wireless data exchange triggered by gesture recognition. It will provide a comparative study on the available methods for gesture recognition explored by research projects that have had similar interests to ours. Not all algorithms mentioned here are candidate solutions for our bracelet, but discussing them helps to gain an understanding of the design choices made further down this thesis.

Section 3.1 and Section 3.2 will provide a rough overview on the conducted research in the field of gesture recognition and handshake recognition, respectively. It will also introduce some terminology and the evaluation criteria needed for discussing other research. Section 3.3, Section 3.4 and Section 3.5 will look into three existing solutions that have similar goals, but suffer from a few shortcomings. Section 3.6 completes the comparative study by analyzing three project that are focussed on gesture recognition in general. Section 3.7 provides an overview table, comparing the projects mentioned above and evaluating their applicability to our desired system.

3.1 Gesture Recognition

Shake-On is not the first to investigate the combination of accelerometer-based gesture recognition and wireless data exchange. Many research projects have been conducted with respect to such sensor networks in healthcare. In [7, 8], the aim of the research is fall detection for elderly people. More general activity monitoring using accelerometers involves detecting resting, walking and running in a non-invasive manner. The work done in [9, 10, 11, 12] focusses on monitoring such activities. Even more specific are [13, 14, 15]. Their work investigates the recognition and classification of

hand gestures, such as handwashing or writing letters with a Nintendo Wii mote.

3.2 Handshake Recognition

For over ten years there has been interest in accelerometer-based hand gesture recognition. Two incentives have been the introduction of smartphones having built-in accelerometers ([16, 17]) and the Nintendo Wiimote ([18, 13, 19, 20]).

This section will discuss some evaluation criteria that is of interest when looking for in-node gesture recognition. Also, some commonly used terminology is introduced. A detailed comparative study is conducted on the used features, classifiers, performance and system drawbacks.

3.2.1 Evaluation Criteria

Having defined the system requirements, existing solutions can be evaluated in a structured way. Since there are limited resources, system complexity and efficiency are important aspects in this comparative study. System complexity is related to the computational capabilities of the platform, in our case a simple nRF51 chip. System efficiency relates to the power consumption, constrained by the requirement of using a coin cell battery and three days without recharging. Recall from Chapter 2 that data transmission is much more costly in terms of energy than computations. This calls for a balanced trade-off between maximizing true positives and minimizing false positives. This can be measured using precision and recall, which considers two questions:

1. How many of the detected handshakes actually occured?
2. How many real handshakes have been detected?

Definition 1 *Precision is the fraction of retrieved data points that are relevant to the query.*

$$precision = \frac{|\ \{real\} \cap \{detected\}\ |}{|\ \{detected\}\ |} \tag{3.1}$$

Definition 2 *Recall is the fraction of data points relevant to the query that are successfully retrieved.*

$$recall = \frac{|\ \{real\} \cap \{detected\}\ |}{|\ \{real\}\ |} \tag{3.2}$$

3.2.2 Terminology

Before discussing related work, some jargon needs to be introduced. The following definitions are not straightforward to all readers, but will be used in the remainder of this thesis.

Definition 3 *Given a gesture sample $\mathcal{G}$ from the accelerometer,* ***classification*** *is the process of determining whether $\mathcal{G}$ can be considered a handshake or not.*

Definition 4 ***Classifiers*** *are classification algorithms that are used in the field of pattern recognition and data mining.*

Definition 5 *Given a gesture sample $\mathcal{G}$, a* ***feature vector*** *$\mathcal{G} = \{f_1, f_2, \ldots, f_n\}$ consists of a set of metrics that characterize $\mathcal{G}$. Examples that are often used in literature are stochastic statistics like the mean μ, standard deviation σ, (co)variance* $\mathrm{Cov}[X, Y]$ *and mean error ϵ.*

Well-known classifiers used in this thesis and related work are Hidden Markov Models (HMM), k-Neighest Neighbors (kNN), Support Vector Machines (SVM) and decision trees like C4.5.

3.3 Existing Solution 1: Smart-Its Friends by Holmquist

The Smart-Its Friends technique aims for easy establishments of connections between two artefacts[21]. Their objective is *'to develop a range of small, embedded devices as platforms for augmentation and interconnection of artefacts'*. During the matching phase, the Smart-It broadcasts its captured movement data and ID to other devices within a limited listening range (see Fig. 3.1). Surrounding devices compare the data to their own. Based on whether the devices are moved similarly, they either establish a dedicated connection or not. This connection will physically break when the devices move out of each others communication range, but will automatically reconnect when placed back in range.

Smart-Its Friends are stand-alone devices, meaning there is no base station (PC) involved. This is a property that meets the stand-alone requirement from Shake-On. Furthermore, there are no restrictions on the synchronized movement that is performed. Because of this, the bracelets are continuously broadcasting. Considering the *Computation over Transmission* design principle, this way of unconditional broadcasting is not energy efficient. All

(a) Matching phase. (b) Two connected devices.

Figure 3.1: **Two Smart Its devices interacting [21].** Upon synchronized movement, they establish a dedicated connection.

movement data is broadcasted over a 868 MHz ISM [1] band, leading to much traffic and continuous context-matching of Smart-Its that are close to each other. One of the authors has been contacted by email to shed some light on unclear matters. Unfortunately, there is no information about the performance of Smart-Its. Given the fact that Smart-Its uses a 2D accelerometer, it would have been interesting to know whether this provides enough information to perform gesture matching. Also, it remains unclear whether Smart-Its uses feature vectors (and what parameters they contain) and whether it is possible for multiple devices to connect to the same Smart-It.

3.4 Existing Solution 2: iBand by Kanis

The iBand is developed by the Human Connectedness group in Dublin for a study towards the influence of *techno-gestures* upon social networking [22]. Two people shaking hands cause their wristband to connect using infrared (IR) transceivers, after which data is exchanged. Like the Smart-It, the device uses a PIC microcontroller and a 2-D accelerometer. After the event, the user returns to the kiosk to upload all collected contacts at a base station. During an interview with Marije Kanis, one of the authors of [22], it appeared that the main focus of the iBand project was the social impact of the digital networking. Technical optimization of the device has not been an important aspect, which becomes clear when looking at the prototype (see Fig. 3.2). The coarse detection of handshaking movement and the need for an alignment of the bulky IR-module are two aspects that need to be avoided. Experiments on detection accuracy have not been conducted or are not public. Even after the interview with Marije Kanis and a short

[1] The industrial, scientific and medical (ISM) radio bands are reserved internationally for purposes other than telecommunications. ("ARTICLE 1 - Terms and Definitions", International Telecommunication Union, 19 October 2009)

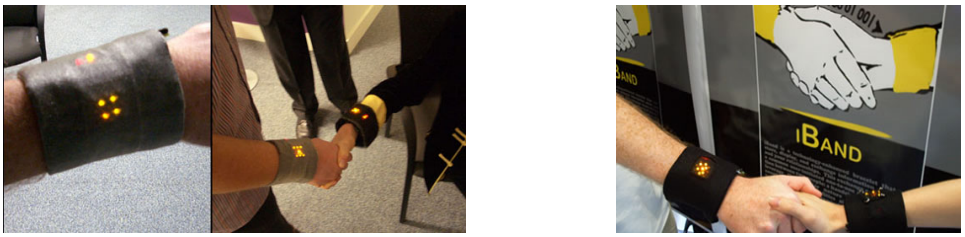


Figure 3.2: **The iBand prototype by [22].** Upon a handshake, the smartbands are connected by means of aligned infrared transceivers.

email conversation with one of the engineers, the system performance was not quantifiable.

3.5 Existing Solution 3: B-HandDS by Augimeri

The B-HandDS [23] can be considered the solution that comes closest to the system desired by Shake-On (see Fig. 3.3). The wristbands are in sleep mode until another device enters their proximity, using a beaconing mechanism. Then, Augimeri introduces a *cooperative handshake detection protocol* (CHDP) that requires both parties to confirm the handshake before they exchange data. Sensor data is first classified locally, after which only potential handshakes are advertised. Then, potential handshakes received from other smartbands are compared to its own and might lead to a connection. For both local classification and matching, a feature vector is composed and fed to an algorithm based on a J48 Decision Tree.[2]

The motivation for the use of the J48 decision tree is the fact that the algorithm is lightweight and computationally inexpensive, good for coarse-grained classification. This is a smart choice, since accuracy will be provided by the CDHP. The feature vector, however, is quite large and comprises parameters like zero crossing, the root mean square (RMS) of each sample, maximum amplitude, frame mean, standard deviation and total energy[3] Using these features has led to the best performance of the classifier, but they also require complex computations. This is not a problem in the project of Augimeri, since a PC base station is used for computation and communication. However, for our scenario of interest the algorithm should be as lightweight as possible and it would be interesting to see to what extent the

[2] A WEKA implementation of the C4.5 algorithm, developed by Ross Quinlan. WEKA is a software tool providing a collection of of machine learning algorithms for data mining tasks [24].

[3] the sum of the square of all values in the frame calculated across X, Y, Z accelerometer sensor channels. It can be considered as a measure of the global activity in a frame [23].

Figure 3.3: B-HandDS prototype by [23].

feature vector can be simplified. Also, the B-HandDS uses a beaconing mechanism to detect other nodes and wake up from an energy preserving mode (sleep mode). This kind of proximity detection is redundant for Shake-On, as the bracelets are either switched on when worn or switched off. Rather, there should be some kind of activity detection in order to wake up the device or put it to sleep. As for the performance, the J48 algorithm suffers the least from false positives compared to other decision trees. B-HandDS has an accuracy of 99%around 75% only considering *recall* and 75% when taking into account *precision*.

3.6 Common Gesture Recognition Techniques

Besides the three projects studied above, more interesting work has been done on gesture recognition. The techniques used are briefly discussed and evaluated on their applicability for our bracelet.

In [14], Wu uses a Nintendo WiiMote to recognize gestures that are intended to give commands (e.g. turn up the volume). The proposed method is a Frame-based Descriptor and Multi-class Support Vector Machine (SVM), being able to distinguish 12 gestures. The feature vector has brought down raw sensor data to five parameters, which reduces signal variance and noise. The conducted experiments show that SVM outperfomes Dynamic Time Warping (DTW) and Hidden Markov Models (HMM), two classifier often used in pattern recognition. The experiments also show that SVM outperforms Naive Bayes and C4.5. However, SVM includes the need for discrete Fourier transform (DFT), only works for easy gestures (a simple 90 degree rotation of the WiiMote) and has an accuracy of 89,3% for user-independant recognition.

In conclusion, Wu shows the benefits of a feature vector-based classifica-

tion algorithm, however has a rather complex implementation for detecting simple gestures.

Another research on the WiiMote is done by Liu in [13], aiming for authentication and navigation based on personalized gesture recognition. The WiiMote sends its acceleration data through Bluetooth to a PC base station that runs the recognition algorithm. The proposed system can detect 8 gestures, using DTW with an accuracy of 75,4% for user-independant recognition (and 93.5% for user-dependent recogntion). For user-dependent recognition, the WiiMote is trained to recognize gestures from only one person, whereas user-independent should recognize gestures performed by different users that all have a different way of moving. With respect to user-dependent gesture recognition, DTW is suitable since it performs well with limited training.

In its recommendations, [13] mentions identification of common features in acceleration data to achieve a better user-independent recognition rate. This is interesting, since our scenario of interest involves user-independent recognition of a known gesture: a handshake.

The last relevant research is [15], which develops a wrist-worn sensor that recognizes the duration and techniques of hand washing. Using a PC running WEKA (also used in [23]), 15 different scrubbing motions are classified in order to study the hand washing behavior of medical staff. With an 8-parameter feature vector and a k-Nearest Neighbour algorithm (kNN), an accuracy of 88.5% is achieved. Interestingly, [15] suggests the decision tree used in [23] as an alternative classifier and it discards the use of Neural Networks because of their poor response time. Also, experiments regarding window length (or frame size) and sampling rates of the accelerometer can be quite useful in a later phase of development.

Project	HGO	Platform	Low Power	Classifier	Features	Precision	Recall
B-HandDS[23]	Yes	PC	Detection	J48	6	99%	75%
iBand[22]	Yes	PCB	IR alignment		N/A	*	*
Smart-Its[21]	Yes	PCB	No detection		RMS *	*	*
Gesture-3D[14]	No	PC		SVM	5	89.3%	*
uWave[13]	No	PC		DTW	N/A*	75.4%	90%
HCW[15]	No	PC		kNN	8	88.5%	*
Desired system							

Figure 3.4: **An overview of all techniques and methods discussed in this chapter.** HGO is short for Handshake Gestures Only. An asterisk (∗) means the content of the cell is unknown or uncertain. Green cells indicate the project meets the design requirements. Orange cells imply room for improvement regarding complexity. Red cells indicate the aspect is not considered or does not meet the design requirements.

3.7 Summary

Fig. 3.4 provides an overview of all techniques and methods mentioned in this chapter. Green cells indicate that the project meets our design requirements. The orange cells imply there might be room for improvement regarding the high number of parameters. Red cells mean the project aspect is not considered or that the design choice does not meet the design requirements. An asterisk (∗) means the content of the cell is unknown or uncertain.

We can conclude from this comparative study that there is no system or algorithm yet that satisfies all of the design requirements. B-HandDS provides a good starting point, proposing the lightweight J48 classifier for handshake detection. Since there are only two categories, namely 'handshake' and 'no handshake', there is no need for more complex classifiers like HMM, SVM or kNN. In order to meet the stand-alone requirement, a lot can be improved with respect to the used feature vectors. Commonly used features often require Fast Fourier Transform, which is no problem considering that all these projects use a PC base station.

The aim of our algorithm of interest is to find new features that can be obtained in a lightweight manner, taking into account the limited resources of a System-on-Chip (SoC). These can then be used for:

1. handshake detection, in combination with the J48 decision tree
2. handshake matching, mapping a handshake into a compressed format resulting in less data to transmit

The remainder of this thesis will described the work done in order to find such an algorithm.

Chapter 4

Handshake Detection

"Is that a dagger I see before me..?"

— Macbeth (W. Shakespeare)

This chapter will deal with challenge 1 from Chapter 2. It is important to distinguish handshake detection from handshake matching, which is discussed in Chapter 5. Thus, before proceeding we note that the questions to be answered have a different nature:

- **Handshake detection**: given one gesture, does it classify as handshaking? [1]
- **Handshake matching**: given more than two candidate handshakes, which one belongs to me?

This chapter will investigate handshake detection by the individual smartbands. Section 4.1 translates characteristics of a handshake into the qualitative properties of orientation, intensity and frequency. Section 4.2 translates these qualitative characteristics to quantitative features using accelerometer data. A hypothesis is formulated regarding a new method for handshake detection that requires only lightweight computations. Section 4.3 explains how the Filter and Wrapper method are used to obtain a minimal feature vector. This vector contains only those features that are crucial for handshake detection. Section 4.4 evaluates the feature vectors obtained by Section 4.3 by feeding them to a J48 decision tree.

[1] Or, using the Shakespeare quote with a slight twist: is that a handshake I see before me?

4.1 Handshakes: A Closer Look

The goal of this section is to find the features that define a handshake and a way to detect them using raw sensor data. The first part will discuss the nature of handshake gestures and formulate three key characteristics. Furthermore, other research projects are discussed regarding the feature vectors used and their applicability to handshake detection.

4.1.1 The Handshake Gesture

When talking about handshaking, the gesture that is addressed is clear. Two people grasp each other's right hand according to etiquette and perform a synchronised up-and-down movement. In a classical handshake, both hands have a thumb-up orientation and the movement takes between one and two seconds. Now, typical characteristics that classify a gesture as a handshake can be derived from the above qualitative description.

For the position of the hand, we can use the fact that the bracelet is strapped to the wrist and will follow all of its motions. When the hand is in a fingers-up orientation, it is very unlikely a handshake will be performed. The wrist *orientation* can therefore be of use in handshake detection. See Fig. 4.1.

Figure 4.1: **Three examples of wrist orientations, useful for detection of a potential handshake.** The second image shows the orientation of a conventional handshake.

The up-and-down shaking is a strong characteristic that can be used to distinguish between gestures. Whereas a handshake can be strong and excited, drinking a cup of coffee needs to be done slowly and in a controlled manner. Also, there is a lot of variation in the way people shake hands, which will be used to our advantage later on when performing handshake matching. Thus, the *intensity* of the movement is a property of interest.

Finally, a handshake is not a static. The rhythm of shaking can vary, with an average of approximately three up-and-down movements per second. Since handshakes are a cooperation of two people, the up-and-down motions of

both gestures are likely to be the same. This *frequency* is therefore a third interesting feature to investigate, resulting in the following 3-tuple to describe a handshake:

$$Handshake = \{orientation, intensity, frequency\} \tag{4.1}$$

4.1.2 Decision Problems vs. Classification Problems

As discussed in Chapter 3, other research projects also use feature vectors. They often included standard stochastic parameters like mean, standard deviation, (co)variance, correlation and mean error. There is a distinction, however, in the nature of the question addresed in the state-of-the-art projects. In general, research questions in the field of pattern recognition are either a:

- Decision problem: "Does this gesture resemble a handshake or not?"
- Classification problem: "Which of the predefined gestures in our library does this sample resemble?"

Classifying a dataset by the full set of standard stochastic statistics is a good method for tackling a search problem. For example, a speech recognition problem ('Who's voice is this?'). Stochastic statistics have proven to be effective speech recognition features. Therefore it is understandable that many academic works have reached out to them for gesture recognition as well.

However, in our case there is a single gesture of interest and needs to be *decided* whether the sample satisfies or not. Given the many ways handshakes can be performed, comparing them to predefined library-handshakes is too strict and will lead to many false negatives.
For the handshake detection algorithm, a feature set should be formulated that captures all handshakes but discards other gestures.

4.2 Sensor Data: Interpreting the Accelerometer

Having defined a qualitative handshake using the 3-tuple from the previous section, the raw sensor data can be studied. The accelerometer output contains a reading for each spatial axis, namely x, y and z. A 16-bit value represents the experienced gravity, with 0 representing no gravity. Gravity is measured by the standard free fall acceleration ($g_0 = 9.81 m/s^2$), which is equal to 1G.

Table 4.1: **Static acceleration on the three spatial axes x, y and z for six basic orientations.** $\pm g$ is equal to the gravitational acceleration of ≈ 9.81 m/s^2.

	x	y	z
Palm up	0	0	g
Palm down	0	0	$-g$
Fingers up	g	0	0
Fingers down	$-g$	0	0
Thumb up	0	$-g$	0
Thumb down	0	g	0

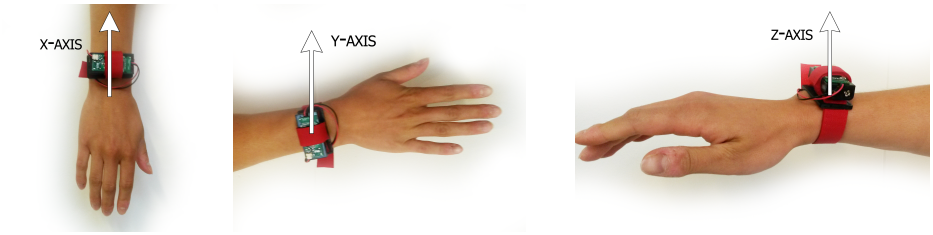


Figure 4.2: **Three static orientations of the smartband**. The second image depicts a handshaking orientation, showing negative gravity for the y-axis. This is useful information for formulation a detection algorithm.

4.2.1 Quantification of A Handshake: Orientation, Intensity and Frequency

By analyzing the LIS3DH accelerometer data, it is possible to distinguish basic movements. As mentioned by [25], a bracelet strapped to the wrist can give insight on static orientations of the hand. In Table 4.1 and Fig. 4.2, the concept of static acceleration is explained for each axis. The sensitivity is set to $\pm$ 2G, resulting in gravity being sensed as $\pm$ 16000. It can be seen that for each position, one axis is subject to gravity and the other two axes form a plane that does not experience gravity (meaning, the accelerometer reading is 0). This phenomenon can be used when detecting handshakes, since for the orientation we are interested in a thumb-up orientation. Putting this in numbers, we can derive the bracelet orientation by looking at the sensor value for each axis.

When it comes to intensity, the three axis also provide useful information. By looking at the maximum and minimum experienced gravity (representing the up-and-down movement), it is possible to quantify intensity. A strong handshake will result in large spikes, while a soft handshake will result in a more flattened signal, as shown in Fig. 4.3. Thus, handshake intensity can be quantified by looking at the range r of the signal. It is also possible

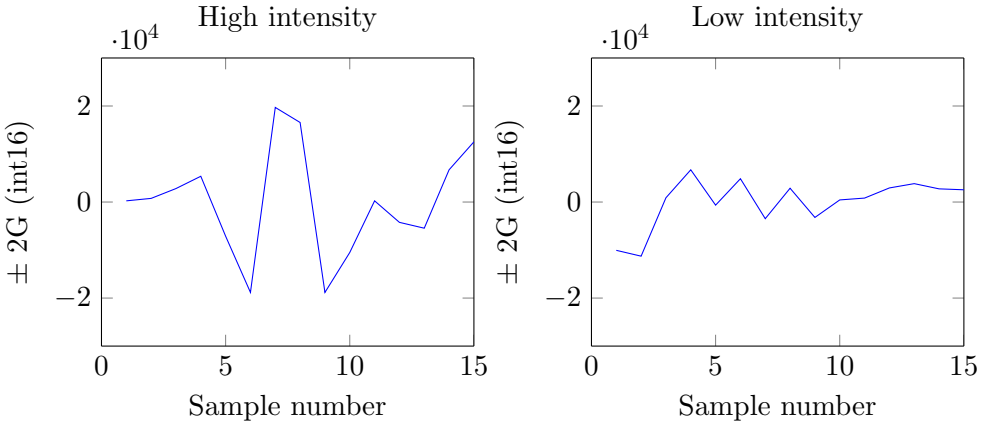


Figure 4.3: **Two handshaking gestures from which the intensity can be retrieved.** A larger amplitude or range (left) implies a higher intensity than a more attenuate gesture (right).

to observe the frequency feature, which is visualized in Fig. 4.4. The dots in the plot emphasize the upper and lower peaks. The frequency can be computed by counting the samples between the peaks (inter-peak distance) and multiplying by the sample rate.

4.2.2 Using Quantified Characteristics to Our Advantage

In conclusion, the readings of the accelerometer offer ways to identify the handshake tuple defined in Section 4.1. The total feature space comprises 9 features, namely orientation, intensity and frequency for each spatial axes x, y and z:

$$Handshake = \{orientation_{x,y,z}, intensity_{x,y,z}, frequency_{x,y,z}\} \quad (4.2)$$

An optimal feature set can be obtained by discarding features that are irrelevant for handshake detection. For example, we already saw how analyzing experienced gravity helps us identify a handshake orientation. The remainder of this chapter will evaluate the following hypotheses:

Hypothesis 1 *Handshake detection can be performed by using the orientation, intensity and frequency as features for pattern recognition.*

Hypothesis 2 *Using this new feature set, handshake detection can be performed with 95% predictive accuracy (precision-recall) and a smaller feature vector than existing solutions.*

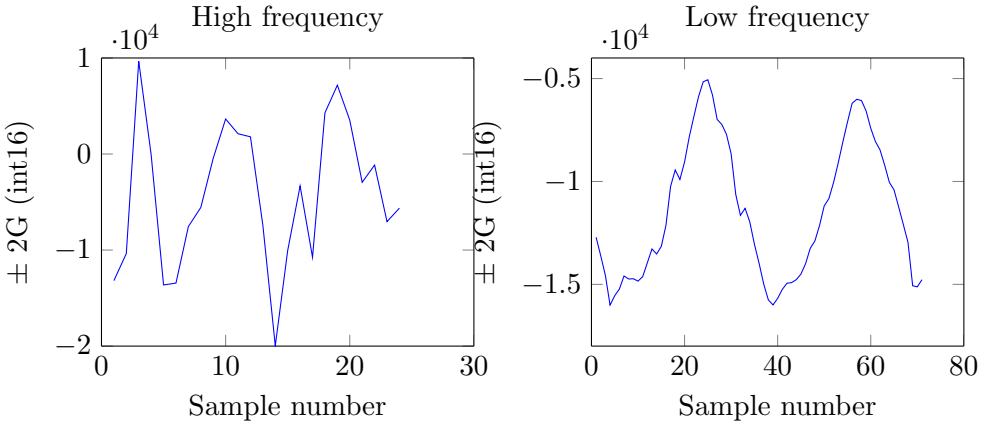


Figure 4.4: **Two handshaking gestures from which the frequency can be retrieved.** Many spikes (left) implies faster handshaking than only a few spikes (right).

4.3 Method

When using pattern recognition for classifying predefined gestures, it is necessary to gather training data. Feature extraction is performed on the training data to translate qualitative handshake characteristics to numerical features. Recall that each reading contains sensor values for three axes, resulting in 9 features, namely orientation, intensity and frequency for the x, y and z plane. Using sensitivity analysis, the impact of each feature on the classification accuracy is determined. Finally, a minimal set of features is identified that provides an accuracy of 95%.

4.3.1 Training Data Acquisition

In order to gather training data, the bracelet was worn by a test group. This group consisted of my roommates and me. Each participant performed a handshake with the other five participants. This was repeated three times, with the setup mimicking three different scenarios: standing, walking and sitting. These scenarios were based on typical situations that people are likely to meet at a conference or fair. The sample rate of the sensor was set at 10 Hz as done in [23] and the sensitivity set at ± 2G. The sensor data was transmitted wirelessly to the PC terminal using proprietary radio communication by means of an in-house developed protocol. Each run included the five handshakes and other random movements that were performed in between. So, for six participants performing five runs for three rounds the dataset contains 90 handshakes. Fig. 4.5 shows the prototype bracelet and

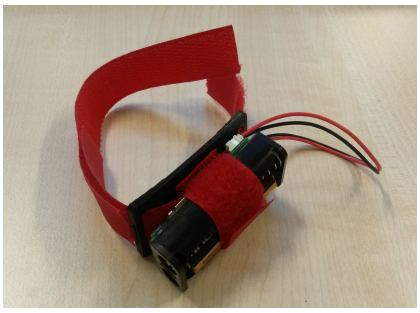


Figure 4.5: **Training data acquisition with the first prototype and roommates serving as participants.** The bracelet is strapped around the right wrist using Velcro. Sensor readings are sent to the PC.

a photo while collecting training data.

4.3.2 Feature Extraction

The first step in the analysis of the training data is to distinguish all handshakes from other random movements. This is done using visual inspection of the raw data plots, followed by manual annotation of the handshakes. Each annotated handshake has a duration of 1.5 seconds, equal to 15 sensor readings. This value is chosen since it appears to be the typical duration of a handshake when studying the training data. Of course, the dataset consists of many other movements that are not handshakes. This means, features need to be computed for the entire dataset in order to classify it. Since features are computed using frames of 15 samples, a *sliding window* is used that moves along the received sensor data. After every reading, the window is shifted to the right by one sample[2]. This way, every sensor reading will lead to a new feature vector that can be used for classification.

Definition 6 *A rectangular window function $w(n)$ is a mathematical function that is constant inside and zero-valued outside interval [0, N-1].*

$$w(n) = \begin{cases} 1, & \textit{if } n \in [0, N-1] \\ 0, & \textit{otherwise} \end{cases} \tag{4.3}$$

[2]The shift size is initially set to be one, but can be varied in a later experimental phase.

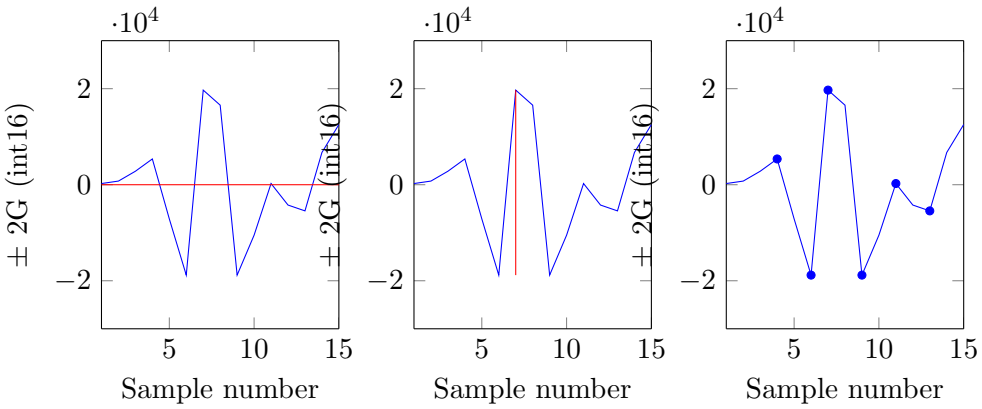


Figure 4.6: **Plots depicting how the features are computed.** The mean μ is found by calculating the average value of the frame. The range r is found by calculating the amplitude, or the difference between the maximum and minimum value. The peaks are found by computing the local extrema.

Using MATLAB, the three important handshake features from the *Handshake* tuple are computed. For each axis, the features can be translated from a qualitative to a quantative description as follows:

- *Orientation*: the average window mean μ
- *Intensity*: the maximum range r
- *Frequency*: local extrema, being the upper and lower peaks

Fig. 4.6 shows how the mean μ, range r and peaks (red) are computed from the raw sensor data (blue).

More importantly, Fig. 4.7 shows all recorded gestures performed by the participants. For each sample, the x, y and z-component are depicted as a blue, green and red dot, respectively. Thus, each sample is represented by three dots. Handshakes are marked as black dots. Due to the wide variability of the handshakes, the black dots are scattered quite a lot. To eliminate outliers, the 5% outermost black dots are not taken into account. Based on a 95%-rule, dashed boxes mark the values required to satisfy a handshake. As an example, three handshakes are emphasized to show how for each axis the dot is within the dashed box. These are the circles, triangles and squares.

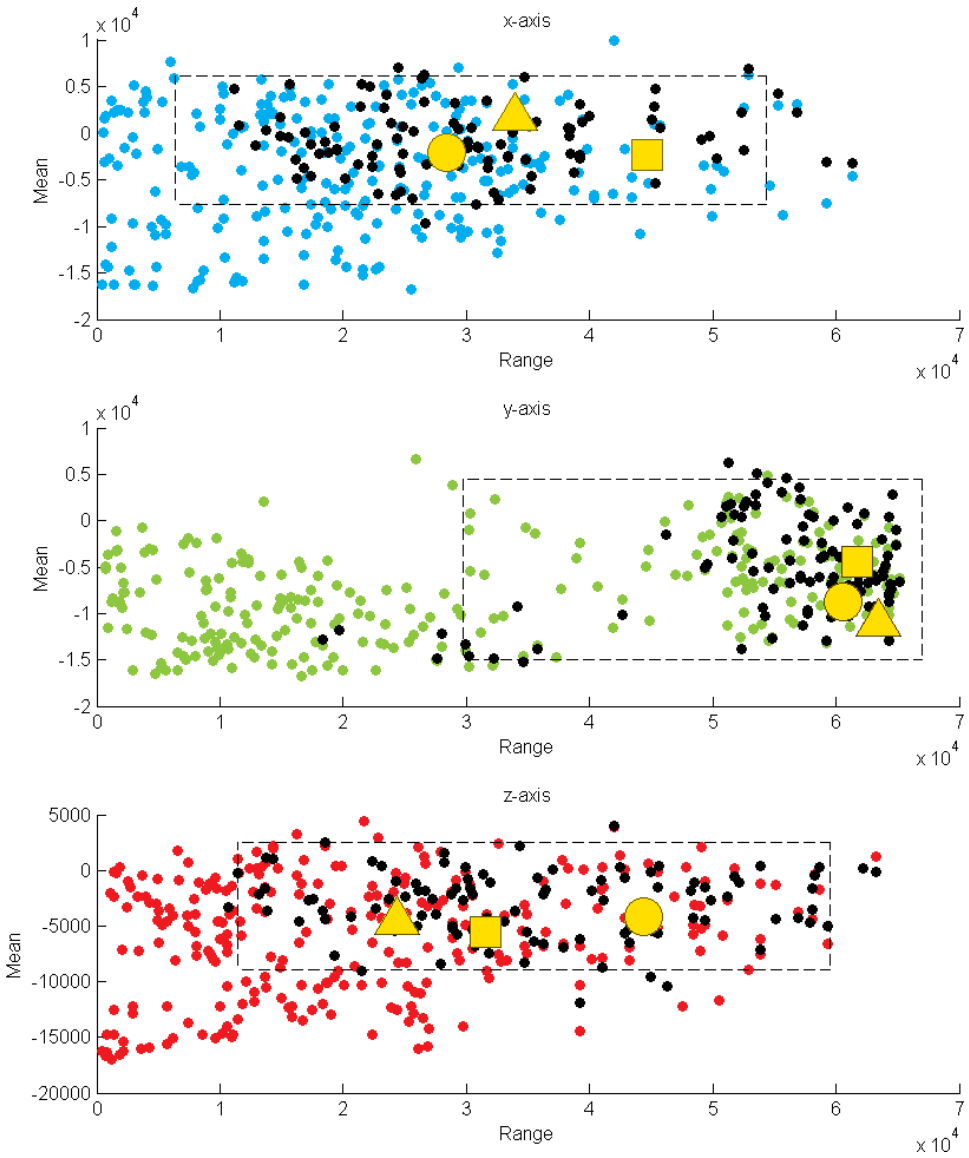


Figure 4.7: **Gesture samples represented as dots for the x, y an z-axes.** The dashed box indicates the area that satisfy handshaking behaviour. For a gesture to be classified as a handshake, the dots should be within the box for each axis. This is shown for three examples, being the circles, triangles and squares.

Table 4.2: **Snippet from training data that is used as input for WEKA.** The feature vector comprises of the window mean μ, range r, and peaks. The last column holds the label YES or NO, dependending on whether the window contains a handshake or not.

x_μ	x_r	x_d	y_μ	y_r	y_d	z_μ	z_r	z_d	handshake?
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
5568	14912	2.57	-5755	47488	2.37	-10167	25152	4.50	no
57258	14912	2.28	-4996	47488	2.50	-95018	25152	4	no
53034	16512	2.25	-4070	47488	2.66	-81450	25152	4.25	yes
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

4.3.3 Feature Selection

In classification, some features of a vector might play a more important role than others. The question is which of the nine features are of interest for detecting a handshake and which can be discarded. In order to find the minimum set of features necessary to gain a 95% accuracy, the data mining tool WEKA from [23, 15] is used.

First the training data is processed in MATLAB, computing all 9 features of F for each run using the sliding window concept. Recall that handshakes are annotated beforehand as they can be easily distinguished by visual inspection of the training data. Now, an `.arff` file can be composed, which is the data format for WEKA. See Table 4.2. Now there are two approaches offered by WEKA for feature selection, namely the *Filter method* and the *Wrapper method*.

The Filter Method

The filter method uses a subset evaluator and a ranker to assign a weight to each feature in the dataset. Based on the ranking, features from the lower ranks can be omitted one by one. After each removal the predictive accuracy (precision and recall) can be estimated with the preferred classifier, in our case the J48 decision tree. This way, it is possible to examine and plot so-called *learning curves*. Typically, a learning curve will show better performance as more low ranked features are omitted. Because the information they hold is insignificant, they only add noise to the classifier. However, omitting low-ranked features works up to a point when the feature subset becomes too small and the classifier will start suffering from underfitting . Underfitting occurs when the learning hypothesis considers too few features, resulting in a too coarse-grained classification. Thus, finding the global minimum of the learning curve will lead to the best feature subset.

Table 4.3: **Feature ranked according to the Filter method.** We use 10-fold cross-validation, the *InfoGainAttributeEval* evaluator and a ranker (199 instances).

Rank	Feature
1	y_r
2	z_μ
3	z_r
4	x_μ
5	x_r
6	y_μ
7	z_d
8	y_d
9	x_d

The Wrapper Method

The wrapper method uses a subset evaluator that will identify *all possible subsets* from your feature vector. This means, every possible combination of features will be evaluated for their classification performance. Next a classifier is defined, in our case the J48 decision tree. Both for the Filter and Wrapper method, classification can be performed using either n-fold cross-validation[3] or percentage split[4]. In our test 10-fold cross validation is chosen over percentage split. This is because our dataset contains 90 samples and for percentage split a larger dataset is preferred.

The result comprises a ranking of the features based on their appearance in each of the 10 folds. Very often, not all features are equally significant. The next step involves omitting the lowest-ranked features and have WEKA compute the precision and recall on the obtained $F_{wrapper}$.

4.4 Results

First, the filter method is used. The chosen subset evaluator is *InfoGainAttributeEval*, accompanied by a ranker. Using 10-fold cross-validation, the features are ranked as shown in Table 4.3. By omitting the lowest ranked feature one by one, the optimal subset of features is found by looking a the predictive accuracy. In Table 4.4, each newly obtained feature vector is

[3]Used to limit overfitting. Cross-validation involves partitioning the dataset in a *validation* subset and a *testing* subset. To reduce variability, multiple rounds (or *folds*) are performed using different partitions, and the validation results are averaged over the rounds.

[4]With percentage split, a part of your dataset is used to train the system and the rest for testing.

Table 4.4: Predictive accuracy for different F_{filter}, using 10-fold cross-validation and a J48 decision tree classifier.

x_μ	y_μ	z_μ	x_r	y_r	z_r	x_d	y_d	z_d	**Precision (%)**	**Recall (%)**
■	■	■	■	■	■	■	■	■	97.6	97.5
■	■	■	■	■	■				97.6	97.5
■		■	■	■	■				94.7	94.5
■		■		■	■				96.1	96.0
		■		■	■				85.3	85.4
		■		■					82.9	82.9

evaluated for meeting the constraint of 95% accuracy in both precision and recall.

Next the wrapper method is used. The subset evaluator is set to *ClassifierSubsetEval* for the J48 decision tree and the search method to Best Fit. Using 10-fold cross-validation, the features are ranked based on their frequency of appearance. See also Table 4.5. Again, by omitting the lower features one by one, the optimal subset of features is found by looking at the predictive accuracy. The constraint of a precision and recall of 95% leads to Table 4.6.

In conclusion, the minimal feature vector that satisfies the accuracy requirement of 95% precision and recall contains 4 features and can be either F_{filter} or $F_{wrapper}$ based on the Filter method and Wrapper method, respectively.

$$F_{filter} = \{x_\mu, z_\mu, y_r, z_r\} \tag{4.4}$$

$$F_{wrapper} = \{x_\mu, x_r, y_\mu, y_r\} \tag{4.5}$$

Since both feature vectors have the same size and performance, we have to pick either one of the methods. A disadvantage of the filter method is that the weight put by the ranker algorithms is different than those used by classification algorithms. For this reason, the filter method is often used for large feature vectors that need to be reduced drastically. For example, going from a thousend features down to fifty. A typical application to use the filter method is data mining, which includes finding coarse patterns in very large datasets.

A typical application to use the wrapper method is machine learning tests. This is the goal of the algorithm since the bracelet should learn to detect a handshake. Thus, for our case of interest the wrapper method is preferred. See Appendix A for the resulting decision tree.

Table 4.5: **Feature ranked according to the Wrapper method.** We use 10-fold cross-validation, the *ClassifierSubsetEval* evaluator and a J48 classifier (199 instances).

#folds (max 10)	**Feature**
10	x_μ
10	y_μ
9	y_r
6	x_r
6	z_r
2	x_d
2	z_μ
1	y_d
1	z_d

Table 4.6: Predictive accuracy for different $F_{wrapper}$ using 10-fold cross-validation and a J48 decision tree classifier.

x_μ	y_μ	z_μ	x_r	y_r	z_r	x_d	y_d	z_d	**Precision (%)**	**Recall (%)**
■	■	■	■	■	■	■	■	■	97.6	97.5
■	■	■	■	■	■	■			97.1	97.0
■	■		■	■	■				95.7	95.5
■	■				■	■			94.6	94.5
■	■		■	■					95.6	95.5
■	■				■				94.7	94.5

4.5 Summary

This chapter has explored a new type of features to perform handshake detection. The fact that we know the gesture of interest is used to our advantage by looking at the characteristics of a handshake. These are identified and translated into quantifiable characteristics:

- Wrist orientation, found by computing the frame mean μ
- Handshake intensity, found by computing the frame amplitude (or range r)
- Shaking frequency, found by computing the local extrema (or peaks)

Using the Filter and Wrapper method in the data mining tool WEKA, two lightweight feature vectors are composed. Both contain 4 features and have gain a predictive accuracy of 95% precision and recall. Recall from Chapter 3 that existing solutions that achieve such high accuracy require a PC to run complex classifiers with large stochastic features.

The two hypotheses formulated at the beginning of the chapter are supported by the above results. Handshake detection can be done in a less complex manner than existing solutions, without compromising on performance.

Chapter 5

Handshake Matching

The goal of this chapter is to verify whether the qualitative handshakes characteristics defined earlier can be used for comparing and matching handshakes. Recall the questions from Chapter 4 and how their nature is different:

- **Handshake detection**: given one gesture, does it classify as handshaking?
- **Handshake matching**: given more than two candidate handshakes, which one belongs to me?

In the literature, cross correlation and dynamic time warping (DTW) are often used for gesture recognition. DTW is an algorithm for measuring similarity between two temporal sequences which may vary in time or speed. For gesture recognition this is useful, since the recorded sample could be matching the reference gesture in the predefined library, only shifted in time or skewed. However, handshaking is a synchronized movement that does not take varying speed into account.

Cross correlation is a measure of similarity between two series as functions of the lag of one relative to the other. It will be tested and evaluated for its applicability on handshake matching. As we shall see, there are a few limitations when applying simple cross correlation. therefore, a new method will be proposed that aims to overcome the shortcomings of cross correlation.

The findings in the previous chapter give reason to again formulate a hypothesis regarding handshake matching.

Hypothesis 3 *Handshake matching can be performed by examining the orientation, intensity and frequency of the handshake gesture and can be done using only lightweight computations.*

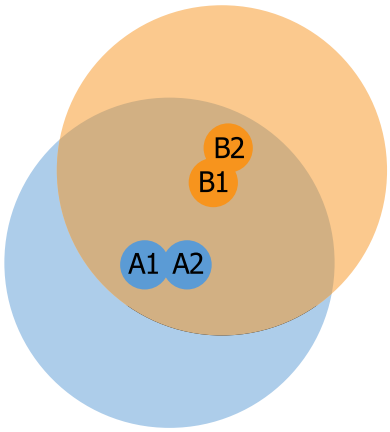


Figure 5.1: **Two pairs of attendees (A and B) shaking hands simultanuously within each others radio range.** The smartbands should only exchange data with their match, so A1↔A2 and B1↔B2.

Section 5.1 introduces the problem of how multiple bracelets can be within each other's listening range. Section 5.2 describes how training data is collected for developing and testing the matching algorithm. Section 5.3 explains why cross correlation on the raw sensor data suffers from the *palm power* effect. Also, it proposes the use of Pareto frontiers that eliminates the orientation characteristic. Section 5.4 explains why cross correlation on the raw sensor data suffers from the *weak shaker* effect. Also, it proposes the use of a new method called *peakmaps* that only consider the frequency characteristic. Section 5.5 explains how peakmaps work and why they obtain a three times better performance basic cross correlation.

5.1 Four is a Crowd

Pairing two bracelets that have both detected a handshake does not seem like a complex task. They will turn on their radio, listen for incoming packets and connect. However, there is a probabiity that multiple pairs of attendees shake hands at similar time and are located within each other's listening range. See Fig. 5.1. To account for this scenario, a matching algorithm is needed that can pair the right bracelets with each other. For this thesis, all proposed matching algorithms are tested considering a pool of four bracelets, so two pairs of attendees shaking hands simultanuously. Since handshaking is a synchronized movement, it is to be expected that mismatching handshakes will show less correlation than handshakes that belong to the same gesture. See also Fig. 5.2.

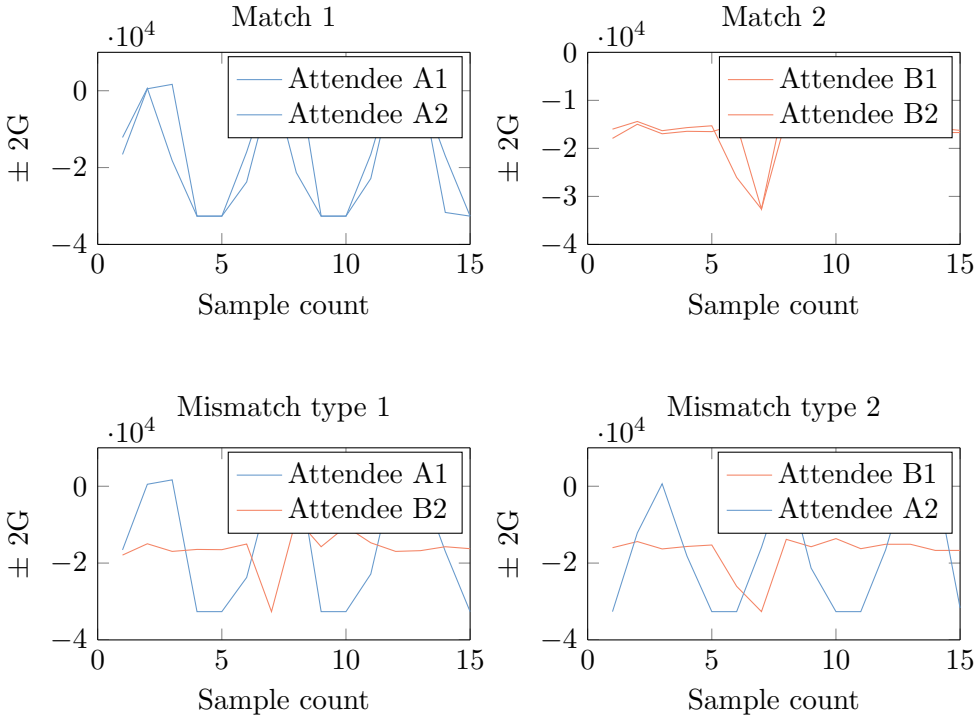


Figure 5.2: **Four handshake examples, cross-matched to illustrate similarity between matching samples.** Note: these plots are generated for explanation purposes. In real life, the bracelets worn by the attendees show less perfect correspondance.

5.2 Training Data Acquisition

In order to gather training data, two handshaking participants wear a bracelet of which the data is recorded synchronously.

The training data is acquired by 8 participants, forming pairs that perform 4 handshakes. When testing the matching algorithm, the set up will consider 2 pairs that shake hands simultanuously. This means test are done on a group consisting of 4 participants, which is repeated for 20 different group configurations. In total, 22 groups performing 4 handshakes yields a dataset of 80 test cases.

Table 5.1: **Setup for training data acquisition.** Eight participants are being cross-matched in order to obtain 80 test cases.

Parameter	Used in dataset
#participants	8
#handshakes performed by a pair	4
#groups of 4 participants	20
#test cases	80

5.3 Cross Correlation on Raw Sensor Data

Previous studies have applied cross correlation to classify gestures captures by accelerometer-based devices. [18, 15, 26] apply cross correlation for a gesture comparison, yielding a performance of 70%-90% accuracy. For this reason, cross correlation is used as a starting point for our matching algorithm.

5.3.1 A Short Introduction to Cross Correlation

Cross correlation[1] is a measure for similarity for signals that are shifted in time. This is useful, since one bracelet might detect the handshake slightly earlier than the other one. The signals are aligned by finding the lag that corresponds with the highest correlation[2].

Definition 7 *In signal processing,* ***cross correlation*** *is a measure for similarity of two series u[n] and v[n] as a function of the lag* $i \in [0, k]$ *of one relative to the other.*

$$(u * v)[n] = \sum_{m=-\infty}^{\infty} (u * [m]v)[m + n] \tag{5.1}$$

Positive correlation exists when, given an increase in $u[n]$, $v[n]$ *also increases and vice versa.*

Negative correlation exists when, given a decrease in $u[n]$, $v[n]$ *also decreases and vice versa.*

[1] Also known as Sliding Dot Product of Sliding Inner-product

[2] The lag is limited at ±500 ms, which is considered the maximum difference in detection.

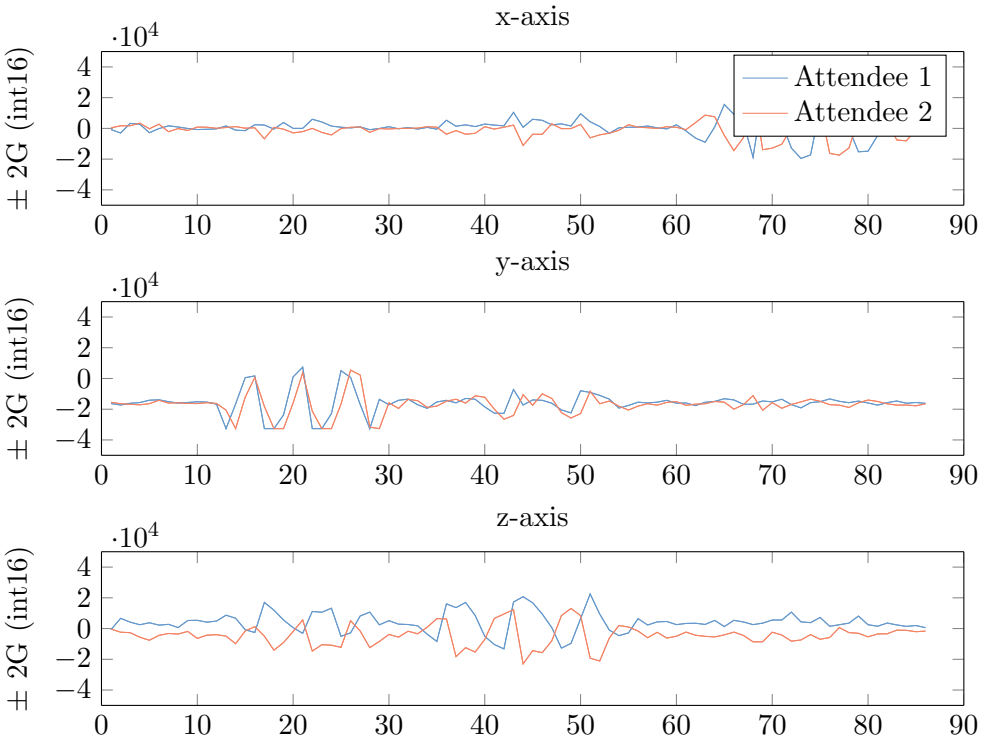


Figure 5.3: **The y-axes of the bracelets show positive correlation, the x-axes and z-axes show negative correlation.** This is caused by the mirrored orientation of the bracelets (see Fig. 5.4). Note: these plots are generated for explanation purposes. In real life, the bracelets worn by the attendees show less perfect correspondance.

5.3.2 Algorithm 1: Simple Cross Correlation

It is reasonable to think the two people shaking hands show identical sensor values. Therefore it is important to introduce the *mirror effect*, caused by the fact that the two people are facing each other. Fig. 5.4 and Fig. 5.3 show how two people facing each other results in inversed sensor values for the x- and z-axis.

In our testcase with 4 attendees shaking hands simultanuously, the matching algorithm should be structured have a basic structure in the form of:

$$\mathcal{M} = f(c1, c2, c3) \tag{5.2}$$

where $c1$, $c2$ and $c3$ are potential matches (or *candidates*) and $f(c1, c2, c3)$ the matching algorithm selecting the matching handshake. In this section, the algorithm is based on cross correlation on the raw sensor data.

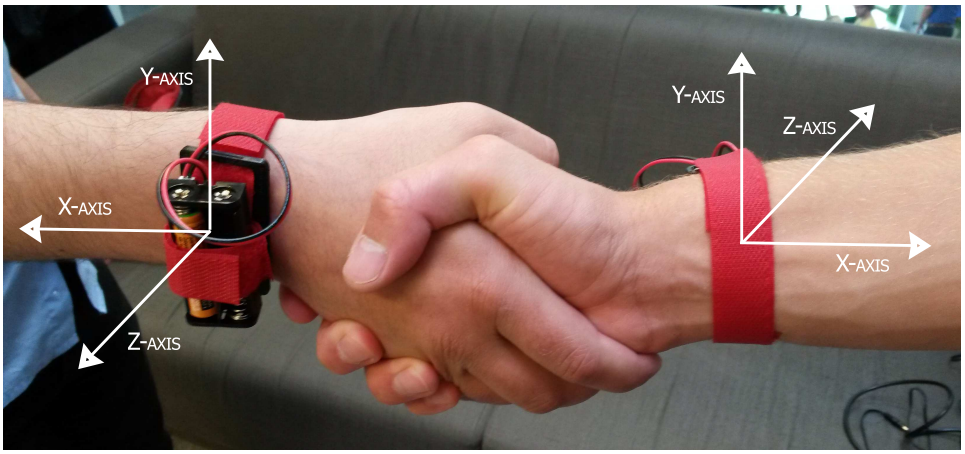


Figure 5.4: **Mirrored x-axis and z-axis as a result of two people facing each other**. See Fig. 5.3 for the corresponding sensor output.

Matching algorithm 1 involves the following steps, to be executed by each bracelet:

1. Collect sensor data from the other 3 candidates
2. For each candidate compute the total maximum cross correlation $M_{max} = (u * v)_{max,x} + (u * v)_{max,y} + (u * v)_{max,z}$. Note: to account for the mirrored x and z-axes, the data for these axes is inverted for one of the bracelets.
3. Construct 3 possible combinations: match (A1A2‖B1B2), mismatch type I (A1B1‖A2B2) and mismatch type II (A1B2‖A2B1)
4. Determine the matching coefficient $\mathcal{M}$ for each combination using the summation $\mathcal{M}_{combination} = M_{pair1} + M_{pair2}$
5. Select the right candidate based on the highest matching coefficient using the equation below.

$$match = \max(\mathcal{M}_{match}, \mathcal{M}_{mismatch1}, \mathcal{M}_{mismatch2}) \tag{5.3}$$

Fig. 5.5 shows how the maximum correlation measure M_{max} is retrieved. Table 5.2 provides an example of Algorithm 1, using the data from Fig. 5.2. It holds the values of M_{max} for all combinations. Next, matching coefficient $\mathcal{M}$ is determined for each of the three possible combinations by adding the M values of the two pairs involved. This leads to Table 5.3, showing that the right combination ($match$) yields the highest matching coefficient $\mathcal{M}$.

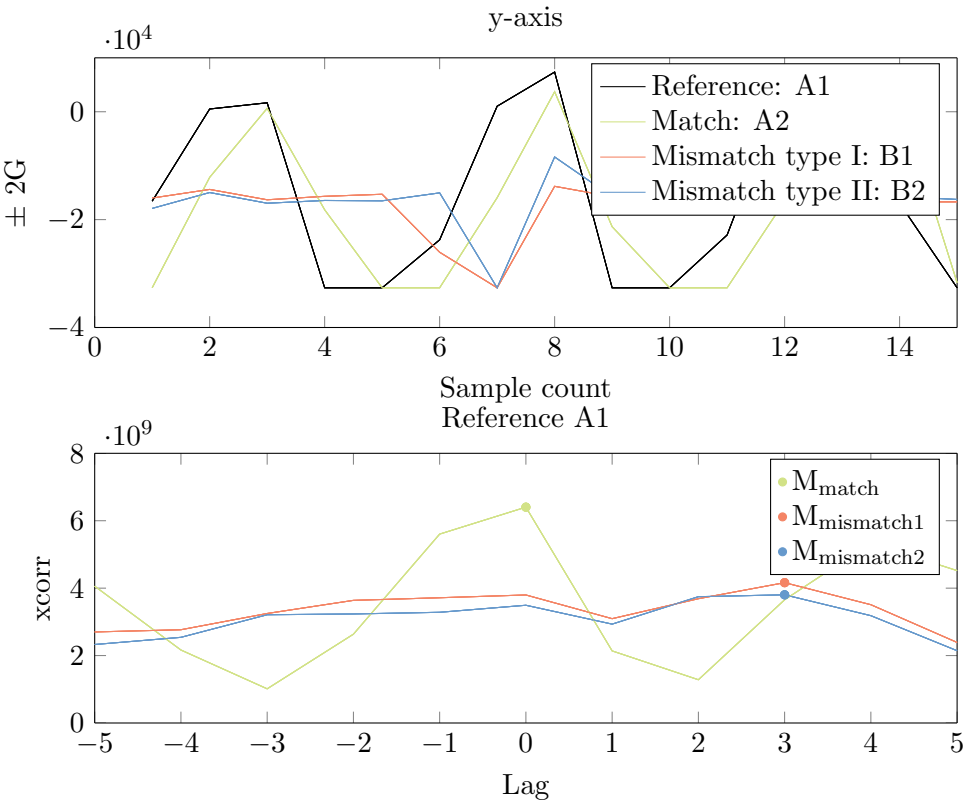


Figure 5.5: **Raw sensor data (upper plot) and cross correlation measure using a lag$\in [-5, 5]$ (lower plot).** The maximum correlation measures M_{max} (marked with a dot) will be used for computing the matching coefficient $\mathcal{M}$.

Table 5.2: Maximum correlation measures M_{max} computed for the example from Fig. 5.2

Reference	**A1**	**A2**	**B1**	**B2**
A1		6.4	4.2	3.8
A2			4.4	4.8
B1				4.5

Table 5.3: **Matching coefficient $\mathcal{M}$ computed for the handshakes from Fig. 5.2.** For the correlation measures, the values from Table 5.2 are used.

Combination	M_{pair1}	M_{pair2}	$\mathcal{M}$
Match (A1A2‖B1B2)	6.4	4.5	10.9
Mismatch type I (A1B1‖A2B2)	4.2	3.8	8.0
Mismatch type II (A1B2‖A2B1)	3.8	4.8	8.6

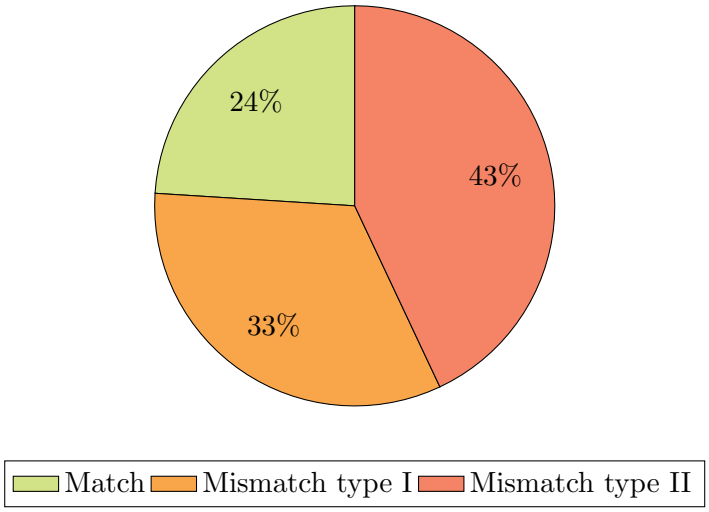


Figure 5.6: **Performance of Algorithm 1.** This is a poor performance as it is even worse than random guessing, which has a 33% probability.

5.3.3 Results Algorithm 1

Algorithm 1 is tested on all 88 test cases from the training dataset. Fig. 5.6 shows the outcome, correctly matching the bracelets only 22% of the time. This result is even worse than random guessing, asking for a critical analysis on Algorithm 1. Studying the sensor data and the maximum correlation measures M_{max} for the x-axis only, it seems to show no difference for matching and mismatching bracelets. This implies that the movements along the x-axis, which corresponds to moving back and forth, are similar for all attendees and insignificant for matching.

Observation 1 *The sensor data for the x-axis is similar for all candidates, adding noise rather than useful information*

5.3.4 Algorithm 2: Analyzing the y, z-plane

Considering the observation made in the previous section, Algorithm 1 can be adjusted to reduce noise by eliminating x-axis data. This means that for Algorithm 2 the maximum correlation measure M_{max} is obtained as follows:

$$M_{max} = (u * v)_{max,y} + (u * v)_{max,z} \tag{5.4}$$

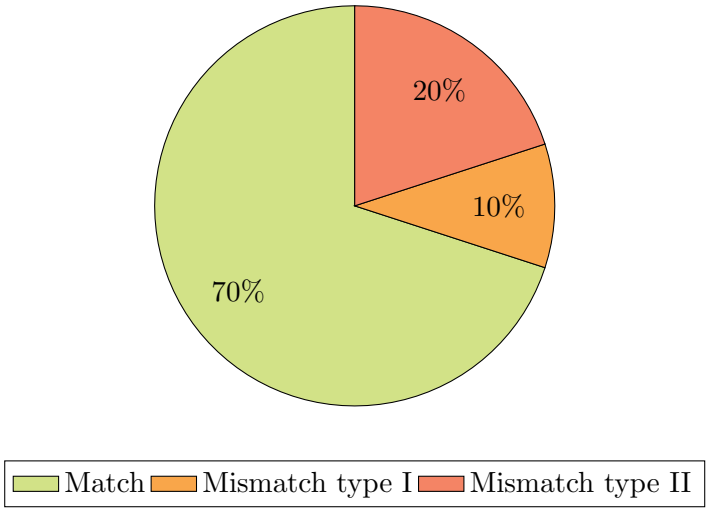


Figure 5.7: **Performance of Algorithm 2.** Taking out the x-axis yields a much better performance compared to Algorithm 1.

5.3.5 Results Algorithm 2

Again, Matching Algorithm 2 is tested on all 88 test cases from the training dataset. Fig. 5.7 shows the outcome, correctly matching the bracelets 58% of the time. Despite the fact that this result is still rather poor, it is quite an improvement compared to Algorithm 1.

Analyzing Algorithmm 2, it is important to consider the fact that we are trying to tackle a 2-dimensional problem (namely the y,z-plane) by studying 1-dimensional signals (for the y and z-axis). Correlation is computed for the y and z-axis separately and then added together. This approach does not take into account the possibility that these two axes might give us more information when they are combined. The handshaking movements are spread over the y and z-axis depending on whether the hand is on top or on the bottom. This is known as *palm power*, described by Pease and Pease[27]. It is explained to be a strategy to obtain either a *dominant* or *submissive* role in a handshake. Handshakes play a major role in body language and have often been the topic in behavioral science. Later in this chapter more handshake behavior will be introduced. For now, the following observation can be made.

Observation 2 *Handshake characteristics are spread over the y and z-axis due to the* ***palm power effect****. Studing each axis separately might result in relevant information getting lost.*

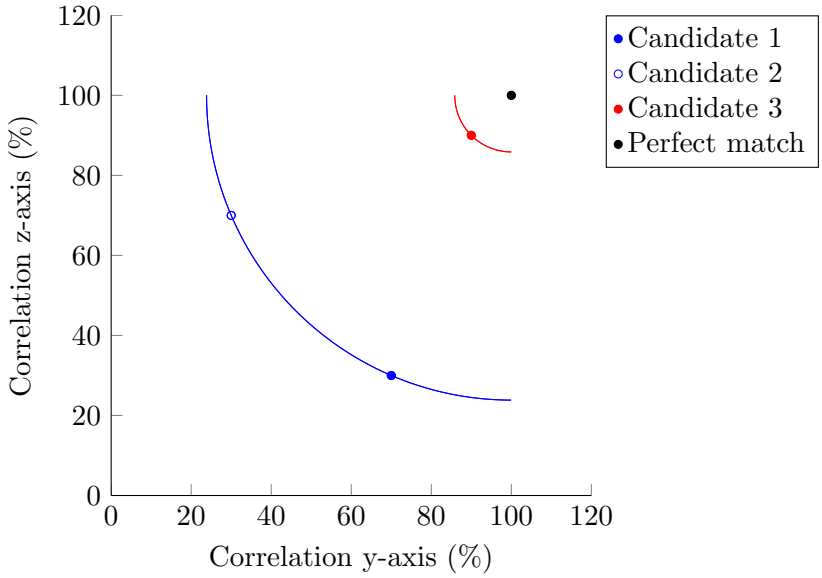


Figure 5.8: **Three candidates plotted as Pareto frontiers.** Since Candidate 3 has the smallest distance to a perfect match (black dot), this is the optimal solution. Note that Candidate 1 and 2 are on the same Pareto frontier. This means they have the same *Pareto efficiency* and are therefore equally matching the reference signal.

5.4 Introducing the Pareto Frontier

The insight from Algorithm 2 provides reason to consider a new way of computing M_{max}. We will no longer study the y and z-axis individually, which means the notion of *orientation* is discarded. Instead, we study both axes simultanuously using Pareto frontiers.

5.4.1 A Short Introduction to Pareto Frontiers

Pareto frontiers are easily explained using one of the handshakes as an example. For two handshakes that matching perfectly, the correlation for both the y and z-axis would be 100%. Now consider three candidates showing a 50%-50%, a 60%-40% and a 40%-60%correlation for the y an z-axis, respectively. Which candidate is a better match? Fig. 5.8 shows how a Pareto frontier can help, since it depicts the distance from both candidates to the perfect solution: a 100%-100% score.

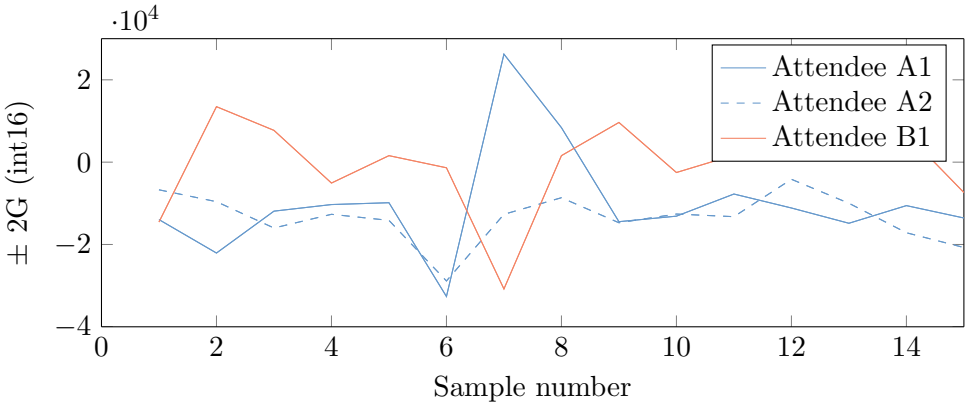


Figure 5.9: **Illustration of the *weak shaker* effect.** Both attendee A1 (blue) and attendee B1 (red) select attendee A2 (dashed) to be their match, since this shake matches everybody.

Definition 8 *A Pareto frontier is a set of resource allocations (e.g. correlation percentage) that are Pareto efficient, meaning they are an equally eligible solution.*

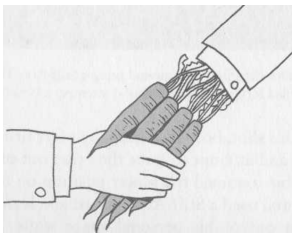
So instead of summing the correlation measures for the individual axes, it is worthwhile to consider the Pareto efficiency. This will be done for our next algorithm.

5.5 Cross Correlation on Peakmaps

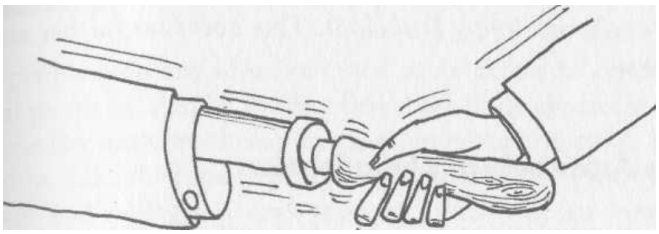
Another interesting observation can be made when analyzing the sensor data. Handshakes with a rather low amplitude (or range r, or *intensity*) are more likely to be selected as match. Handshake with low intensity are referred to in [27] as *The Dutch Treat*.[3]. The opposite is the so-called *Pump Handler*, meaning a very enthusiastic handshake. This distinction is important and will be addressed to as the *weak shaker effect*. Fig. 5.9 shows sensor data for three participants. Participant A1 and Participant B1 are both active shakers (see Fig. 5.10b) and will select Participant A2 (a weak shaker, see Fig. 5.10a) to be their match because the signal is neutral.

Observation 3 *People are either **active** or **passive** handshakers. Two active shakers are often correctly matched, while passive shakers tend to decrease the matching performance because they match to everybody.*

[3]This handshake style has origins in the Netherlands, where a person can be accused of 'Geeft een hand als een bosje wortels' or 'Giving handshake like a bunch of carrots'.



(a) *The Dutch Treat*



(b) A*The Pump Handler.*

Figure 5.10: **Handshaking styles described by [27].** Passive handshakes are referred to as *The Dutch Treat*. Active handshakes are referred to as *The Pump Handler*.

Because the *weak shaker effect* involves intensity, it might be worthwhile to take this element out. Combined with the decision to discard orientation, only the frequency characteristic remains.

Observation 4 *Handshakes should be mapped to a format where only the frequency matters, completely eliminating handshaking style.*

5.5.1 From Raw Sensor Data to Peakmaps

In order to map raw data to a format that only considers frequency, absolute sensor values should be omitted. Using local extrema, so-called *peakmaps* are created that register either no extremum, a maximum or a minimum. This is done in Fig. 5.11, transforming raw sensor data to a bar plot with $peakrange \in \{-1, 0, 1\}$.

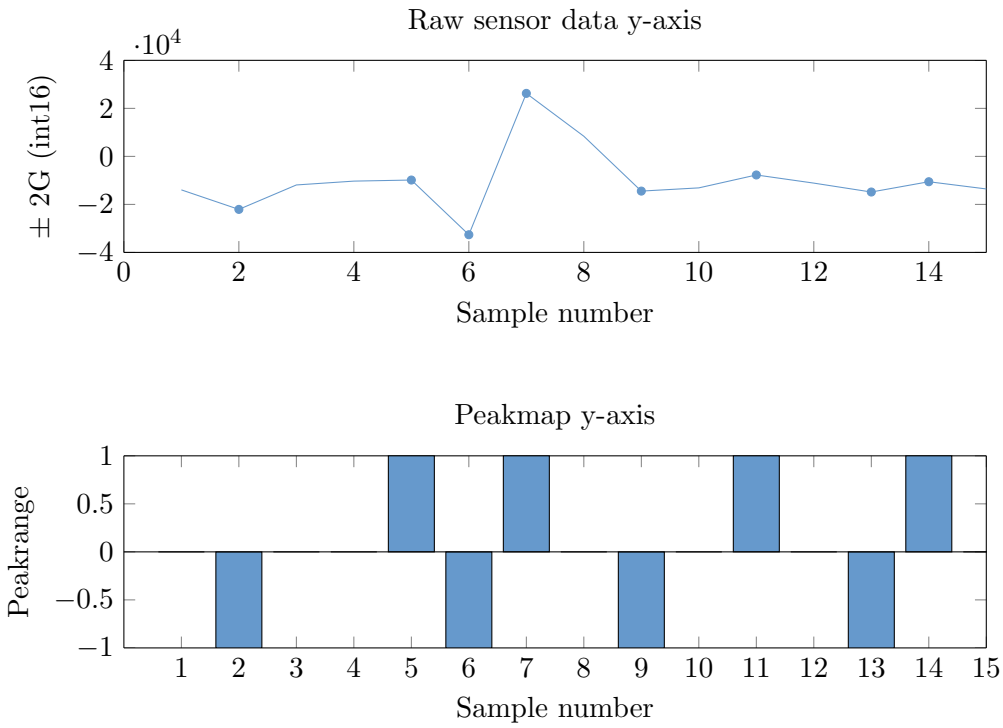


Figure 5.11: **The procedure of *peakmapping* a handshake.** The amplitude of the peak is no longer relevant now that both orientation and intensity have been discarded

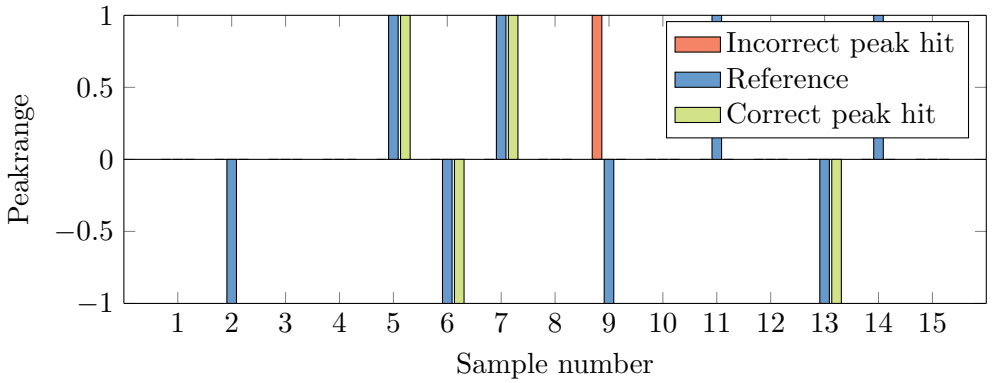


Figure 5.12: **A plot explaining the notion of peak hits and misses**. A peak from the reference signal is either missed (only blue peaks), correctly hit (green peaks) or incorrectly hit (red).

5.5.2 Peak Hits and Misses

This new approach applies cross correlation on peakmaps, which is rather coarse-grained but provides an easy way know whether peaks are *hit* or *missed*. Considering a reference bracelet, peaks that are missed by the candidate add nothing to the correlation since $1 \cdot 0 = 0$. Peaks that are mimicked by a candidate add positive to the correlation measure: $1 \cdot 1 = 1$. Peaks that mimicked but mirrored are worse than a peak miss, as they show an opposite movement. For this reason they add negative to the correlation measure and will be called *negative* peak hits: $1 \cdot -1 = -1$.

Definition 9 ***Correct peak hits*** *occur when the candidate mimicks a peak from the reference bracelet, with corresponding orientation. They result in correlation measure M being increased by 1.*

Definition 10 ***Incorrect peak hits*** *occur when the candidate mimicks a peak from the reference bracelet, but with opposite orientation. They result in correlation measure M being decreased by 1.*

Definition 11 ***Peak misses*** *occur when the candidate does not mimick a peak from the reference bracelet. They result in correlation measure M staying the same.*

Fig. 5.12 provides an example. The blue peaks are the reference, which are either missed (entirely blue), correctly hit (green) or incorrectly hit (red). Since peak misses do not change the correlation measure, they go unnoticed. This means that a mismatch with many peaks is likely to yield a higher correlation measure than the actual match.

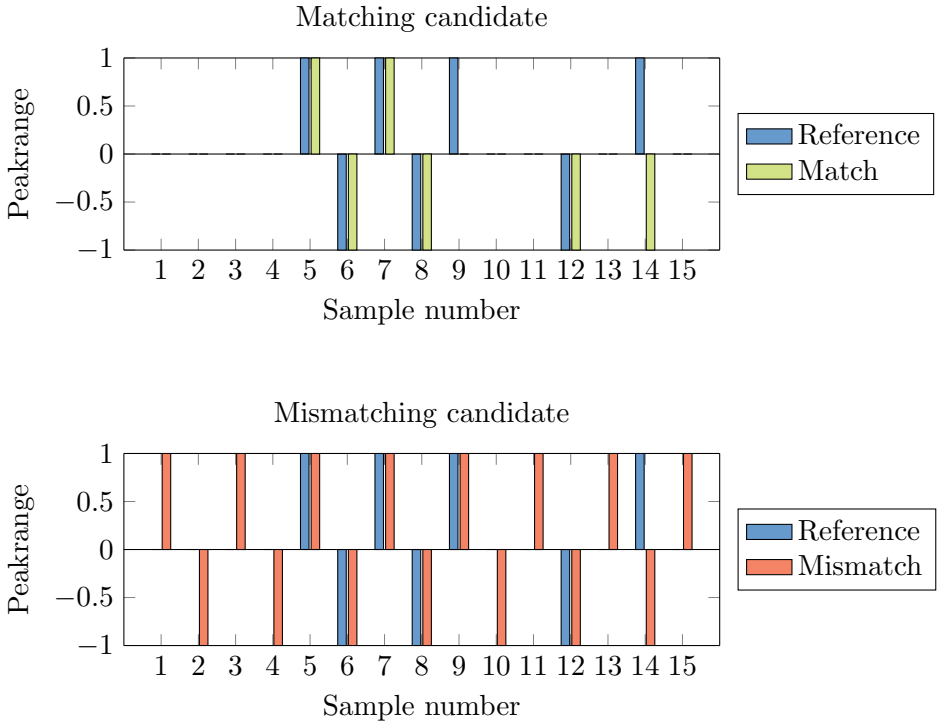


Figure 5.13: **An example of unmodified peakmaps failing due to minor fluctuations.** The matching peakmap (green) has five correct hits and one incorrect hit, resulting in a correlation measure of $M = 4$. The mismatching peakmap (red) has six correct hits and one incorrect hit, resulting in a correlation measure of $M = 6$.
Since the red peakmap has a higher value for M, the algorithm will think this is the match.

Fig. 5.13 provides an example: the mismatching candidate correctly hits more peaks than the matching candidate, but also has many peaks that are not mimicked by the reference. Since these are not considered by basic correlation, the wrong candidate is selected.

Observation 5 *Peakmaps should omit minor fluctuations, since they are likely to influence the correlation measure in an unpredictable way.*

5.5.3 Peak Threshold using top-k

It is interesting to investigate why some peakmaps show many peaks. Especially since handshake typically involves two or three up-and-down movements. Taking another look at the raw data from Fig. 5.11, we can easily distinguish the handshake. It is also manually annotated in Fig. 5.14, result-

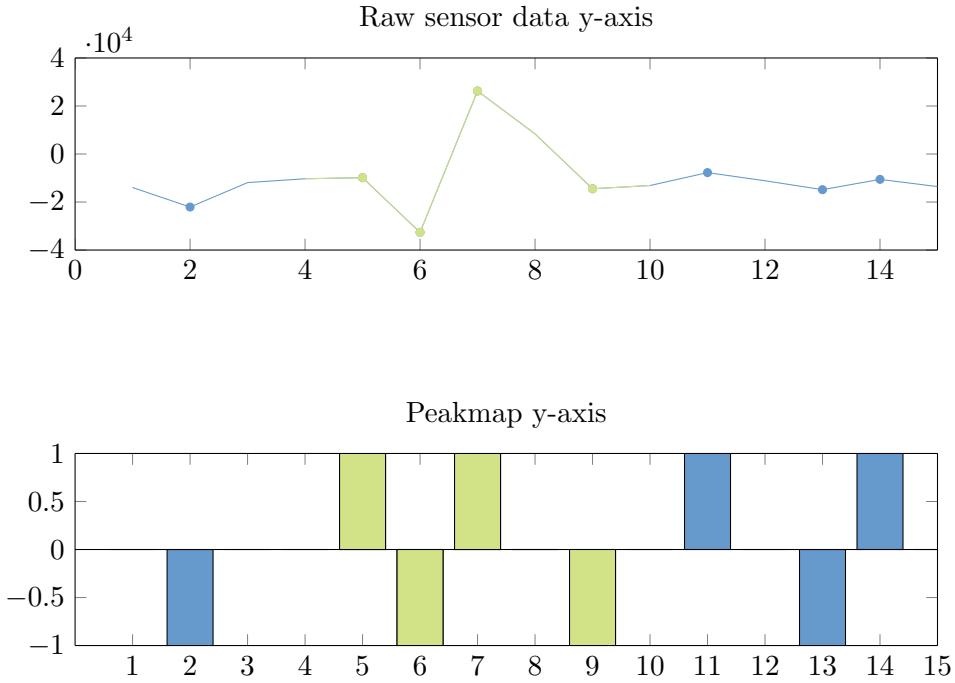


Figure 5.14: **An example of a handshake on which a top-2 threshold is applied.** For the upper and lower peaks, only the two most characteristc are *peakmapped*.

ing in the green peaks. Analyzing the raw data, it appears that the insignificant extrema (blue) are caused by minor vibrations. Since frequency is the only feature being considered, these minor vibrations weaken the handshake uniqueness property.

The last modification to the peakmaps is that only the top-2 upper and lower peaks are allowed in the peakmap.

Observation 6 *Peakmaps should not contain all local extrema, as these add noise. Only the two most characteristic upper and lower peaks should be represent in the peakmap.*

5.5.4 Results Algorithm 3: Peakmaps

Algorithm 3 incorporates all of the observations made in Section 5.5. *Orientation* is discarded by means of Pareto frontiers and *intensity* is discarded using peakmaps. Minor peaks are filtered out using top-k filtering. Matching performance of Matching Algorithm 3 leads to a performance of 80%.

These results are surprising in the sense that only the *frequency* character-

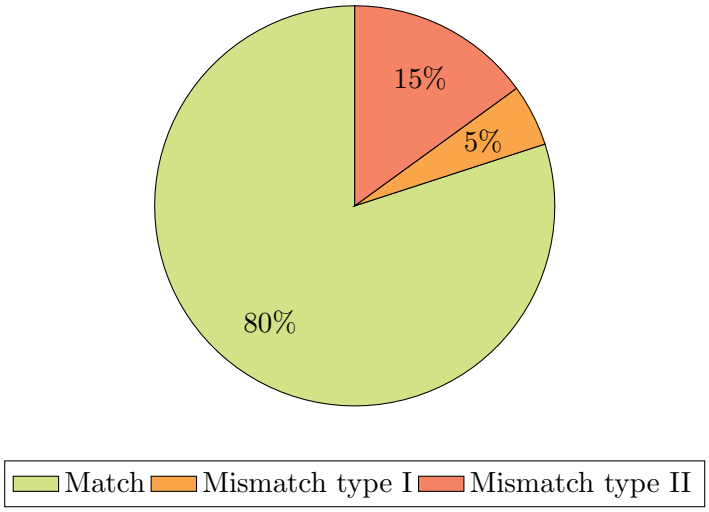


Figure 5.15: **Performance of Algorithm 3.** Correctly pairing the bracelets for 80% of the time, peakmaps outperforms cross correlation on raw sensor data.

istic seems to matter. *Orientation* and *intensity* are properties that make each individual gesture unique. By this point, we have learned that this does not apply to a handshakes, which combines the gesture of two individuals.

5.6 Summary

This chapter has explored a new approach to perform handshake matching. Using cross correlation on raw sensor values as starting point, three steps have led to a solution that has a matching performance of 80%. We have seen that individual traces show very much variability, even between two people handshaking. This has led to an algorithm that only considers synchronized up-and-down movement, omitting *orientation* and *intensity*.

All three steps are reported in this chapter in order to provide insight in the flow of thought. The hypothesis formulated at the beginning of the chapter is supported by the above result. Handshake matching can be done with reasonable performance, without the need for raw sensor data.

Chapter 6

Conclusions and Recommendations

The purpose of the work done is this thesis was to derive methods that could detect two people shaking hands, using only an accelerometer. The two major challenges that were faced comprise 1) user-independent handshake detection and 2) correct matching in the scenario of multiple concurrent handshakes. This chapter will report the results and give recommendations to Shake-On for attaining a final implementation of the bracelet.

6.1 Conclusions

For handshake detection, a tool is developed that computes *orientation*, *intensity* and *frequency* characteristics for a given gesture. It suffices to examine the orientation μ and intensity r of the gesture. The optimal feature vector F, obtained using the Wrapper method, contains four features and has a predictive accuracy of 95%. Compared to existing solutions, this method shows equal performance while requiring fewer features and more lightweight computations.

For handshake matching, an algorithm is derived and evaluated for its ability to select the complementary bracelet from a pool of candidates. We have seen that it suffices to consider only the handshaking frequency. Using basic cross correlation as starting point, a new algorithm is proposed and evaluated. Mapping raw sensor data to *peakmaps* eliminates the *palm power effect* and the *weak shaker effect*. For a group of four bracelets, the peakmap algorithm is able to select its match for 80% of the time. Compared to the performance of basic cross correlation (only 24%), this is more than a factor 3 improvement.

6.2 Recommendations

First of all, the current PCB has a few shortcomings that need to be overcome in a future revision. Not only is it still a bit too large in size, basic components like a reset buttons, a couple of LEDs and a holder for the coin cell battery are preferable.

Regarding the sensing components, this thesis has shown that handshakes can be detected using only an accelerometer. However, problems arise when the bracelet is worn upside down or when people apply palm power[1]. This is because an accelerometer can only measure experienced acceleration, but is not aware its orientation. A solution to this would be to use a gyroscope together with the accelerometer, which can tackle this problem when combined. This enables recognizing movements regardless of how the bracelet is worn.

At the moment, a prototype exists that can detect handshakes and pass on the credentials of the other user, matched by a handshake, to the phone. Future work includes implementing the matching algorithm in order to handle multiple broadcasting bracelets. Once this is done, a large field test would be worthwhile to test both functionality and usability.

The MATLAB tool developed in this chapter computes *orientation*, *intensity* and *frequency* characteristics for given training data. As explained in Chapter 4, the fact that we are explicitly looking for a handshaking gesture is used to our advantage. However, it would be interesting to investigate the performance of the tool with respect to other other gestures, e.g. voting (right arm raised). For Shake-On, this might be very interesting with respect to gesture recognition tailored to individual clients.

Future work includes:

1. Redesign of the PCB, considering the addition of a gyroscope for orientation-independant handshake recognition
2. Formulation of a simple communication protocol that has no need for an initiating node (peripheral-device configuration)
3. A large-scale field test, testing the robustness of the prototype
4. Experiments with respect to the detection of other gestures that might be interesting during conferences, e.g. voting
5. Power optimization to maximize the battery life of the bracelets, for example by creating an energy preserving mode (sleep mode)

[1] A form of powerplay in handshaking, where one person forces the other person to be in a palm-down position which implies silent authority. Explained in Chapter 6.

Appendix A

Handshake Detection Decision Trees

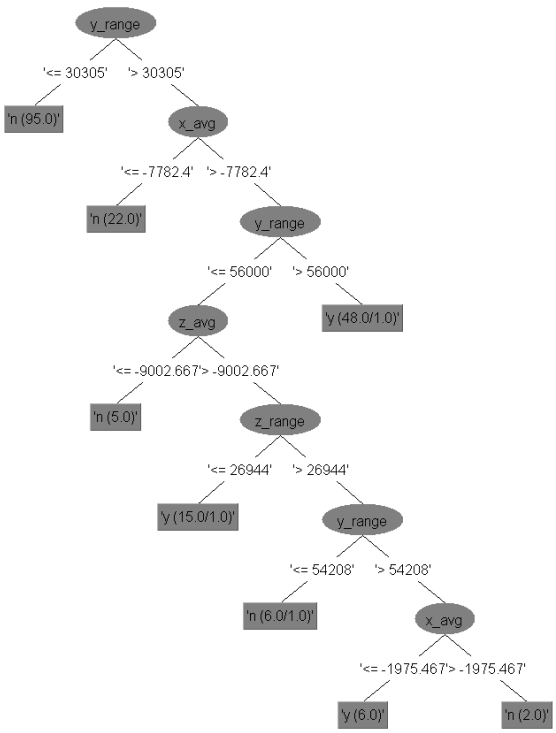


Figure A.1: J48 decision tree for feature vector $F_{wrapper}$ obtained in Chapter 4, generated by WEKA.

Bibliography

[1] P. Steve. (2015) History of the handshake. [Online]. Available: http://www.bumpshake.com/history-of-the-handshake/

[2] J. H. Kietzmann, K. Hermkens, I. P. McCarthy, and B. S. Silvestre, "Social media? get serious! understanding the functional building blocks of social media," *Business horizons*, vol. 54, no. 3, pp. 241–251, 2011.

[3] *nRF51822 Product Specification v3.1.*

[4] *LIS3DH - ultra low-power high performance 3-axes.*

[5] *S110 nRF51822 SoftDevice Specification v1.3.*

[6] D. Slezak, T. Vasilakos, M. Li, and K. Sakurai, *Communication and Networking: International Conference, FGCN/ACN 2009, Held as Part of the Future Generation Information Technology Conference, FGIT 2009, Jeju Island, Korea, December 10-12, 2009. Proceedings*, ser. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2009. [Online]. Available: https://books.google.nl/books?id=UR1uCQAAQBAJ

[7] W.-Y. Chung, Y.-D. Lee, and S.-J. Jung, "A wireless sensor network compatible wearable u-healthcare monitoring system using integrated ecg, accelerometer and spo 2," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*. IEEE, 2008, pp. 1529–1532.

[8] A. Purwar, D. U. Jeong, and W. Y. Chung, "Activity monitoring from real-time triaxial accelerometer data using sensor network," in *Control, Automation and Systems, 2007. ICCAS'07. International Conference on*. IEEE, 2007, pp. 2402–2406.

[9] R. Martinez-Alvarez, D. Rodriguez-Silva, S. Costas-Rodriguez, and F. González-Castaño, "Low cost remote effort monitoring with wearable accelerometers," in *Consumer Communications and Networking Conference*, vol. 13, 2009, pp. 1–2.

[10] Y.-D. Lee and W.-Y. Chung, "Wireless sensor network based wearable smart shirt for ubiquitous health and activity monitoring," *Sensors and Actuators B: Chemical*, vol. 140, no. 2, pp. 390–395, 2009.

[11] A. Kalpaxis, "Wireless temporal-spatial human mobility analysis using real-time three dimensional acceleration data," in *Computing in the Global Information Technology, 2007. ICCGI 2007. International Multi-Conference on.* IEEE, 2007, pp. 1–1.

[12] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim, "A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 14, no. 5, pp. 1166–1172, 2010.

[13] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan, "uwave: Accelerometer-based personalized gesture recognition and its applications," *Pervasive and Mobile Computing*, vol. 5, no. 6, pp. 657–675, 2009.

[14] J. Wu, G. Pan, D. Zhang, G. Qi, and S. Li, "Gesture recognition with a 3-d accelerometer," in *Ubiquitous intelligence and computing.* Springer, 2009, pp. 25–38.

[15] Unknown, "Hand hygiene duration and technique recognition using wrist-worn sensors."

[16] S.-J. Cho, E. Choi, W.-C. Bang, J. Yang, J. Sohn, D. Y. Kim, Y.-B. Lee, and S. Kim, "Two-stage recognition of raw acceleration signals for 3-d gesture-understanding cell phones," in *Tenth International Workshop on Frontiers in Handwriting Recognition.* Suvisoft, 2006.

[17] G. Niezen and G. P. Hancke, "Gesture recognition as ubiquitous input for mobile phones," in *International Workshop on Devices that Alter Perception (DAP 2008), conjunction with Ubicomp.* Citeseer, 2008.

[18] J. Wu, G. Pan, D. Zhang, G. Qi, and S. Li, "Gesture recognition with a 3-d accelerometer," in *Ubiquitous intelligence and computing.* Springer, 2009, pp. 25–38.

[19] T. Schlömer, B. Poppinga, N. Henze, and S. Boll, "Gesture recognition with a wii controller," in *Proceedings of the 2nd international conference on Tangible and embedded interaction.* ACM, 2008, pp. 11–14.

[20] S. Kratz and M. Rohs, "A $3 gesture recognizer: simple gesture recognition for devices equipped with 3d acceleration sensors," in *Proceedings of the 15th international conference on Intelligent user interfaces.* ACM, 2010, pp. 341–344.

[21] L. E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H.-W. Gellersen, "Smart-its friends: A technique for users to easily establish connections between smart artefacts," in *Ubicomp 2001: Ubiquitous Computing*. Springer, 2001, pp. 116–122.

[22] M. Kanis, N. Winters, S. Agamanolis, A. Gavin, and C. Cullinan, "Toward wearable social networking with iband," in *CHI'05 extended abstracts on Human factors in computing systems*. ACM, 2005, pp. 1521–1524.

[23] A. Augimeri, G. Fortino, M. R. Rege, V. Handziski, and A. Wolisz, "A cooperative approach for handshake detection based on body sensor networks," in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*. IEEE, 2010, pp. 281–288.

[24] E. Frank, M. Hall, P. Reutemann, and L. Trigg. Weka 3: Data mining software in java. [Online]. Available: cs.waikato.ac.nz

[25] D. Biswas, D. Corda, G. Baldus, A. Cranny, K. Maharatna, J. Achner, J. Klemke, M. Jöbges, and S. Ortmann, "Recognition of elementary arm movements using orientation of a tri-axial accelerometer located near the wrist," *Physiological measurement*, vol. 35, no. 9, p. 1751, 2014.

[26] M. Raptis, D. Kirovski, and H. Hoppe, "Real-time classification of dance gestures from skeleton animation," in *Proceedings of the 2011 ACM SIGGRAPH/Eurographics symposium on computer animation*. ACM, 2011, pp. 147–156.

[27] B. Pease and A. Pease, *The definitive book of body language*. Bantam, 2008.