

**FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO**

# **Assessing Accuracy of Low Cost Sensors in Sign Language Recognition**

**Daniel Lima Fernandes Vieira**



Mestrado em Engenharia de Software

Supervisor: João Pascoal Faria

Second Supervisor: Luís Filipe Teixeira

November 4, 2023



# **Assessing Accuracy of Low Cost Sensors in Sign Language Recognition**

**Daniel Lima Fernandes Vieira**

Mestrado em Engenharia de Software

November 4, 2023

# Resumo

De acordo com a Organização Mundial de Saúde, 1.5 bilhões de pessoas no mundo sofrem de algum nível de perda auditiva e 398.9 milhões possuem perda auditiva entre moderada e total. Para essas pessoas o principal meio de comunicação é a língua escrita ou algum tipo de língua gestual. Em Portugal, de acordo com a Associação Portuguesa dos Surdos, a Língua Gestual Portuguesa é falada por 30.000 pessoas com algum tipo de deficiência auditiva e pode chegar a 100.000 se considerarmos pessoas que podem se comunicar com línguas faladas, mas que por algum motivo aprenderam a língua gestual. Mesmo assim, se considerarmos a total população de Portugal, menos de 0,1% da população consegue se comunicar efetivamente através de uma língua gestual, o que gera uma enorme lacuna de integração para pessoas dependentes de uma língua gestual na sociedade.

Com a evolução tecnológica das últimas décadas inúmeros estudos vêm sendo realizados com o objetivo de diminuir essa lacuna e ajudar na inclusão dessa comunidade na sociedade. Esses estudos se aproveitam de inúmeros hardwares disponíveis para outros mercados e dos avanços na criação de componentes de hardware baratos, integráveis e com framework aberto (ex.: Arduino), e de software com a constante evolução dos algoritmos de aprendizado de máquina e redes neurais. Infelizmente, tão numerosas são os estudos, como são os tipos de hardware e software utilizados. Por ser uma área de estudos ainda muito despadronizada, novos estudos são fundamentais para validar novas técnicas e maneiras de abordar esse problema.

O objetivo principal desse estudo é determinar se sensores de baixo custo são capazes de gerar dados com qualidade suficiente para alimentar algoritmos de aprendizado de máquina com objetivo de realizar reconhecimento da Língua Gestual Portuguesa.

Para alcançar esse objetivo, e considerando o cenário onde existem diversas abordagens diferentes para esse problema, esse estudo abordou todas etapas necessárias para alcançar uma conclusão: realizamos a escolha técnica de quais sensores irão ser utilizados para gerar os dados do estudo; construímos um hardware capaz de gerar autonomamente dados durante a realização da língua gestual; realizamos a captura de dados de utilizadores da língua gestual com diferentes perfis (fluentes/não fluentes; homens/mulheres) durante sua execução, para um dicionário específico para esse estudo; aplicamos diferentes algoritmos de classificação para identificar a melhor relação custo/resultados, identificar se os dados possuem qualidade suficiente para esse tipo de estudo e se os algoritmos conseguem, em um primeiro e mais simples nível, classificar corretamente os movimentos para uma posterior tradução da Língua Gestual Portuguesa.

As análises preliminares realizadas nos dados gerados pelo hardware indicam que sensores de baixo custo conseguem gerar dados com qualidade suficiente para alimentar modelos de classificação e reconhecimento. Os dados possuem constância e baixo nível de ruído. Também foram identificados pontos que devem ser trabalhados em estudos futuros com objetivo de gerar resultados melhores e expandir a base de conhecimento adquirida: dificuldades na montagem do hardware; desafios na construção de um bom data set de treino para os algoritmos.

A execução dos algoritmos de classificação sob os dados capturados indica uma boa adequação

entre os dados e os objetivos do estudo. Os diferentes algoritmos aplicados foram capazes de classificar, em diferentes níveis de acurácia e tempos de processamento, os dados com sucesso. Foi possível atingir, com parâmetros específicos, uma acurácia de 82% na classificação de palavras e uma acurácia de 99% na classificação de letras do alfabeto, posicionando esse estudo no mesmo nível de outros trabalhos observados nessa mesma área.

Os resultados obtidos nesse estudo contribuem para o avanço dessa área, que é tão importante para inclusão dessa parcela da população dependente de línguas gestuais para comunicação no dia a dia.

# Abstract

According to the World Health Organization, 1.5 billion people worldwide suffer from some level of hearing loss, and 398.9 million have moderate to total hearing loss. For these people, the main means of communication is written language or some kind of sign language. In Portugal, according to the Associação Portuguesa dos Surdos, Portuguese Sign Language is spoken by 30.000 people with some type of hearing impairment, and it can reach 100.000 if we consider people who can communicate with spoken languages but who for some reason learned to use sign language. Even so, if we consider the total population of Portugal, less than 0.1% of the population is able to communicate effectively using a sign language, which creates a huge gap in the integration of people dependent on a sign language into society.

With the technological evolution of recent decades, numerous studies have been carried out with the aim of reducing this gap and helping to include this community in society. These studies take advantage of the numerous pieces of hardware available and advances in the creation of cheap and integrable hardware components with an open framework (e.g. Arduino) and software with the constant evolution of machine learning algorithms and neural networks. Unfortunately, so numerous are the studies, as are the types of hardware and software used. Because it is an area of study that is still very unstandardized, new studies are fundamental to validate new techniques and ways to approach this problem.

The main objective of this study is to determine whether low-cost sensors are capable of generating data of sufficient quality to feed machine learning algorithms in order to perform recognition of Portuguese Sign Language.

To achieve this goal, and considering the scenario where there are several different approaches to this problem, this study addressed all the necessary steps to reach a conclusion: we made the technical choice of which sensors will be used to generate the study data; we built a hardware capable of autonomously generating data during the realization of the signals; we captured data from sign language users with different profiles (fluent/non-fluent; men/women) during the execution of signs, for a specific dictionary design to this study; we applied different classification algorithms to identify the best cost/result ratio, identify whether the data are of sufficient quality for this type of study and whether the algorithms are able, at a first and simplest level, to correctly classify the movements for subsequent Portuguese Sign Language translation.

Preliminary analyses carried out on the data generated by the hardware indicate that low-cost sensors can generate data with sufficient quality to feed classification and recognition models. The data has consistency and a low noise level. Points were also identified that should be worked on in future studies aiming to generate better results and expand the acquired knowledge base: difficulties in assembling the hardware; challenges in building a good training data set for the algorithms.

The execution of the classification algorithms on the captured data indicates a good fit between the data and the study objectives. The different algorithms applied were able to successfully classify the data at different levels of accuracy and processing times. It was possible to achieve,

with specific parameters, an accuracy of 82% in the classification of words and an accuracy of 99% in the classification of letters of the alphabet, positioning this study at the same level as other works observed in this same area.

The results obtained in this study contribute to the advancement of this area, which is so important for the inclusion of this portion of the population dependent on sign languages for day-to-day communication.

# Acknowledgements

The realization of this thesis consists of an old dream, an idea that came up when I was studying a degree in Systems Analysis in Brazil. I always had with me the idea that the main purpose of all technological evolution carried out by humanity in the last centuries is to help other people.

To Ana Cristina Ramada Paiva for being an excellent first contact of mine with the FEUP and for accepting that I bring an idea from outside FEUP to carry out this thesis. I also would like to thank my advisor João Pascoal Faria for believing in my idea and for encouraging me since our first contact.

To Armando Gabriel Teixeira Baltazar, coordinator of Associação dos Surdos do Porto, for welcoming me so well into the community of Portuguese Sign Language users, for explaining to me details of the deaf community that I did not know and for helping me capture the data, so essential for my thesis.

To my spouse Camila Lobão and my son Heitor Vieira for all their patience and help throughout the period I was studying for my master's degree.

Daniel Lima Fernandes Vieira

*“Technology is best when it brings people together”*

Matt Mullenweg

# Contents

|          |                                                                              |           |
|----------|------------------------------------------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                                                          | <b>1</b>  |
| 1.1      | Context . . . . .                                                            | 1         |
| 1.2      | Portuguese Sign Language . . . . .                                           | 2         |
| 1.3      | Problem . . . . .                                                            | 2         |
| 1.4      | Objectives and Contributions . . . . .                                       | 4         |
| 1.5      | Methodology . . . . .                                                        | 4         |
| 1.6      | Document Structure . . . . .                                                 | 5         |
| <b>2</b> | <b>Background and State of the Art</b>                                       | <b>6</b>  |
| 2.1      | Introduction . . . . .                                                       | 6         |
| 2.2      | Sign Language Recognition Methods - Hardware . . . . .                       | 7         |
| 2.2.1    | Computer Vision . . . . .                                                    | 7         |
| 2.2.2    | Commercial Gloves . . . . .                                                  | 8         |
| 2.2.3    | Blood Flow Sensors . . . . .                                                 | 8         |
| 2.2.4    | Stretch Sensors . . . . .                                                    | 9         |
| 2.2.5    | Inertial Sensors . . . . .                                                   | 9         |
| 2.2.6    | Summary . . . . .                                                            | 10        |
| 2.3      | Classification Models and Algorithms for Sign Language Recognition . . . . . | 11        |
| 2.4      | Performance Metrics for Sign Language Recognition . . . . .                  | 12        |
| 2.5      | Summary . . . . .                                                            | 13        |
| <b>3</b> | <b>Hardware Assembly</b>                                                     | <b>14</b> |
| 3.1      | Platform . . . . .                                                           | 14        |
| 3.2      | Microcontroller . . . . .                                                    | 15        |
| 3.3      | Sensors . . . . .                                                            | 16        |
| 3.4      | Integration . . . . .                                                        | 18        |
| 3.5      | Communication . . . . .                                                      | 18        |
| 3.6      | C++ Arduino Code . . . . .                                                   | 19        |
| 3.7      | Summary . . . . .                                                            | 20        |
| <b>4</b> | <b>Data Acquisition</b>                                                      | <b>21</b> |
| 4.1      | Creation of the Work Dictionary . . . . .                                    | 21        |
| 4.2      | Data Structure . . . . .                                                     | 23        |
| 4.3      | Initial Analysis . . . . .                                                   | 24        |
| 4.3.1    | Gravity Force . . . . .                                                      | 24        |
| 4.3.2    | Noise . . . . .                                                              | 24        |
| 4.3.3    | Timeline and Motion Equivalence . . . . .                                    | 25        |
| 4.4      | Summary . . . . .                                                            | 26        |

|          |                                                  |           |
|----------|--------------------------------------------------|-----------|
| <b>5</b> | <b>Analysis and Recognition Algorithms</b>       | <b>28</b> |
| 5.1      | Classification Hardware . . . . .                | 28        |
| 5.2      | Data . . . . .                                   | 28        |
| 5.2.1    | Data Sanitization . . . . .                      | 28        |
| 5.2.2    | Labeling . . . . .                               | 29        |
| 5.2.3    | Errors and Flaws in the Acquired Data . . . . .  | 29        |
| 5.3      | Decision Tree . . . . .                          | 30        |
| 5.3.1    | Objects and Actions Classification . . . . .     | 30        |
| 5.3.2    | Letters Classification . . . . .                 | 31        |
| 5.3.3    | Letters Classification - Long exposure . . . . . | 32        |
| 5.4      | Random Forest . . . . .                          | 33        |
| 5.4.1    | Objects and Actions Classification . . . . .     | 34        |
| 5.4.2    | Letters Classification . . . . .                 | 34        |
| 5.4.3    | Letters Classification - Long exposure . . . . . | 35        |
| 5.5      | Summary . . . . .                                | 36        |
| <b>6</b> | <b>Real-time Recognition Approach</b>            | <b>39</b> |
| 6.1      | System Architecture . . . . .                    | 39        |
| 6.2      | Tests . . . . .                                  | 40        |
| 6.3      | Summary . . . . .                                | 41        |
| <b>7</b> | <b>Conclusions and Future Work</b>               | <b>42</b> |
| 7.1      | Conclusions . . . . .                            | 42        |
| 7.2      | Future Work . . . . .                            | 44        |
|          | <b>References</b>                                | <b>46</b> |
| <b>A</b> | <b>C++ Arduino Microcontroler Code Example</b>   | <b>49</b> |

# List of Figures

|     |                                                                                            |    |
|-----|--------------------------------------------------------------------------------------------|----|
| 2.1 | CyberGlove III. <a href="http://cyberglovesystems.com">cyberglovesystems.com</a> . . . . . | 8  |
| 2.2 | Various Sensor Types . . . . .                                                             | 10 |
| 3.1 | Development Board With ESP32-WROOM-32 Microcontroller . . . . .                            | 15 |
| 3.2 | MPU 6050 (GY-521) . . . . .                                                                | 17 |
| 3.3 | Schematics . . . . .                                                                       | 19 |
| 3.4 | Final Hardware - Left Hand . . . . .                                                       | 20 |
| 4.1 | Measuring Set - Right Hand . . . . .                                                       | 23 |
| 4.2 | Constant Gravitational Analysis - Z Axis Accelerometer . . . . .                           | 25 |
| 4.3 | Gyroscope Noise Analysis . . . . .                                                         | 26 |
| 4.4 | Word "Jump" Details Movements . . . . .                                                    | 26 |
| 4.5 | Word "Jump" Data Comparison . . . . .                                                      | 27 |
| 6.1 | Architecture of the Real Time Classification System . . . . .                              | 39 |
| 6.2 | Real Time Recognition System Identifying Letters "C" . . . . .                             | 40 |

# List of Tables

|      |                                                                                                                                  |    |
|------|----------------------------------------------------------------------------------------------------------------------------------|----|
| 2.1  | Hardware - Sensor Analysis . . . . .                                                                                             | 10 |
| 4.1  | Specific Dictionary and Data Acquisition . . . . .                                                                               | 22 |
| 4.2  | Example of Collected Data - Thumb Finger - Right Hand . . . . .                                                                  | 23 |
| 5.1  | Final Data sets For Classification Using Algorithms . . . . .                                                                    | 29 |
| 5.2  | Result of the Decision Tree Classification Model For Objects And Actions . . . .                                                 | 31 |
| 5.3  | Best Result Decision Tree (30%/70%) For Objects and Actions Classification Us-<br>ing Accelerometer And Gyroscope Data . . . . . | 31 |
| 5.4  | Result of the Decision Tree Classification Model For Letters . . . . .                                                           | 31 |
| 5.5  | Best Result Decision Tree (30%/70%) For Letter Classification Using Accelerom-<br>eter And Gyroscope Data . . . . .              | 32 |
| 5.6  | Result of the Decision Tree Classification Model For Letters Long exposure . . .                                                 | 33 |
| 5.7  | Best Result Decision Tree (30%/70%) For Letter Long Exposure Classification<br>Using Accelerometer And Gyroscope Data . . . . .  | 33 |
| 5.8  | Result of the Random Forest Classification Model For Objects and Actions . . .                                                   | 34 |
| 5.9  | Best Result Random Forest (30%/70%) For Objects and Actions Classification<br>Using Accelerometer And Gyroscope Data . . . . .   | 35 |
| 5.10 | Result of the Random Forest Classification Model For Letters . . . . .                                                           | 35 |
| 5.11 | Best Result Random Forest (30%/70%) For Letters Classification Using Accelerom-<br>eter And Gyroscope Data . . . . .             | 36 |
| 5.12 | Result of the Random Forest Classification Model For Letters Long exposure . .                                                   | 36 |
| 5.13 | Best Result Random Forest (30%/70%) For Letters Long Exposure Classification<br>Using Accelerometer And Gyroscope Data . . . . . | 37 |
| 5.14 | Objects and Actions Classification - Decision Tree Vs Random Forest . . . . .                                                    | 38 |
| 5.15 | Letters Classification - Decision Tree Vs Random Forest . . . . .                                                                | 38 |
| 5.16 | Letters (Long Exposure) Classification - Decision Tree Vs Random Forest . . . .                                                  | 38 |

# Abbreviations

|     |                           |
|-----|---------------------------|
| DIY | Do It Yourself            |
| DT  | Decision Tree             |
| LGP | Lingua Gestual Portuguesa |
| ML  | Machine Learning          |
| RF  | Random Forest             |

# Chapter 1

## Introduction

This chapter details and contextualizes the work carried out in the development of the basic idea of this thesis on Section 1.1, a brief explanation about Sign Language by Portuguese on Section 1.2, the importance of the worked idea for the solution of real problems on Section 1.3, the impact on the academic community and Portuguese Sign Language speakers community and the contribution to future studies on Section 1.4, the methodology used since the construction of the hardware for data collection and subsequent analysis on Section 1.5 and the final structuring of the entire study work on Section 1.6.

### 1.1 Context

According to the World Health Organization, 1.5 billion people in the world have some level of hearing loss, and 398.8 million have moderate to total hearing loss (WHO, 2021). For these people, communication using some kind of visual language or sign language becomes the main and most important means of communication.

Sign language, or visual language, is a language of visual communication that has been developed within communities of hearing-impaired people around the world. It is composed of movements of the hands, body, and facial expressions that express feelings, emotions, actions, numbers, and letters of the alphabet.

Although sign languages are based on one or two hand movements, body movements, and facial movements, it is important to note that different sign languages, no matter how visually similar they may be, can be completely different from each other, depending on the location and the context in which they were created. They may have similarities, as different spoken languages do, but they are not universal. They may or may not be connected with their base spoken version and with the culture of the community that originated it.

According to Ethnologue (Ethnologue, 2023), a group dedicated to the study of languages in the world, there are around 157 sign languages currently in use, spread across different countries and communities.

William Stokoe, lauded as the father of signed language linguistics, concluded that sign languages possessed the linguistic aspects of a genuine language, in the lexicon, syntax, and their ability to generate infinite sentences (Xavier, 2006).

## 1.2 Portuguese Sign Language

The Portuguese Sign Language (LGP) was created by Pär Aron Borg, a Swede who had created an education institute for the deaf in Sweden and who was called to Portugal in 1823 by King D. João VI to create an equivalent institution (History, 2023). However, formal and more in-depth studies on this language only started in the 1980s, influenced by the International Year of Disabled People (News, 2003), with the study of Maria Isabel Prata and Raquel Delgado Martins (Prata, 1980), where the first Portuguese Sign Language dictionary was created. Contrary to common sense, Portuguese Sign Language is not based on the spoken Portuguese language but on the Swedish language, which, despite being different, has a common alphabet.

The syntax of Portuguese Sign Language has a specific structure defined as Subject-Object-Verb or Object-Subject-Verb, with no dominant order and thus being simpler and more direct than a spoken language (Gonçalves et al., 2021). Interrogative, declarative, and exclamatory punctuation is performed with facial, shoulder, and eyebrow movements. There are also single hand movements for each letter of the alphabet, allowing for the spelling of proper nouns and eventually a word that has no proper representation in sign language. The Portuguese Sign Language is made up of 26 letters of the alphabet and more than 25,000 entries segmented into actions, subjects, objects, and more (Infopedia, 2023a).

Being spoken by more than 30.000 people, according to the Associação Portuguesa de Surdos (de Surdos, 2023), and by more than 100.000 if we count relatives, teachers, and others (CENSOS, 2023), Portuguese Sign Language is considered well structured and widespread in the community, which makes it a good case study to test the feasibility of a recognition system. Unfortunately, this covers less than 1% of Portugal's population, which makes actions that allow a better integration of hearing and non-hearing people essential (Alves et al., 2022).

## 1.3 Problem

Unfortunately, communication between speakers of voiced languages and speakers of visual languages has a great knowledge barrier, as that knowledge is not normally exchanged outside the community of use. Few people without hearing problems outside the community learn how to use a visual language. This generates a situation in which two people literate in the same country's language cannot communicate due to a lack of exchange between the necessary knowledge for the spoken language and the sign language.

There are alternatives to allow better communication between speakers and users of visual languages, such as using writing or the help of interpreters. However, for common everyday situations, either of these two suggestions becomes complicated or too costly for routine use.

An innovative approach to the problem of communication between speakers and users of visual languages would be the use of equipment that allows the interpretation and translation of the flow of communication from gesture to speech. A device capable of converting sign language movements into text or sound would facilitate communication in its most fragile way, since someone who is not verbose in sign language is not able to understand the movements, but a sign language user can usually read texts and lip movements. Studies focused on this area can provide models and technology for creating new ways of integration for those who depend on sign language for their communication, as well as innovative ways of applying statistical algorithms in the analysis and recognition of patterns in data.

The creation of full systems capable of recognizing sign language movements (which will not be the purpose of this study) with quality has several limitations that are already known and that must be considered, such as:

- Part of the communication carried out through sign language comes from facial, shoulder, and body movements (Soly Mathew Biju, 2023). Considering that the use of facial sensors would become uncomfortable and impeding for day-to-day use, new manual movements must be created to represent punctuation and signals made by the face and body (D. P. Corina and Reilly, 1999), for example: defining exclamatory, interrogative, and declarative punctuation in sentences.
- Despite advances in the interpretation of natural human language by algorithms, the automatic identification of the beginning and end of sentences remains a challenge for a system (Wang et al., 2002). In order not to overload the system, a new manual movement must be created to identify the beginning and end of the sentences, allowing the correct analysis and interpretation of the data flow to be recognized. This would also allow the system to only be triggered when the user wanted it.
- A new motion pattern should be created for communicating large numbers. Possibly using a system based on Roman numerals would be feasible.
- The interpreter must be flexible enough to allow for small variations in movement inherent in each individual but specific enough not to misclassify similar signals.

In addition to the limitations derived from Portuguese Sign Language, we have limitations arising from the technology and approach that will be used in the construction of the recognition system, such as:

- The system must be user-friendly and cannot be limited to specific or unrealistic usage scenarios.
- The total cost of the system cannot be an impediment to its use by the largest portion of those who need it.

- The sign language recognition process must be efficient and be able to correctly perform recognition in at least two-thirds of the attempts to generate confidence in using the equipment.

A system capable of overcoming most or all of these limitations would have an extremely positive impact on the speech-impaired community.

## 1.4 Objectives and Contributions

Considering the problem exposed in the previous session, the main objective of this study is to verify if low-cost sensors (with a final value of €1.00 or less) can generate data with sufficient quality and reliability to feed a Portuguese Sign Language recognition system with good accuracy and performance efficiency. Although it is not part of the main objective described, to achieve it, it will be necessary primarily to build a rudimentary system of collection, analysis, and interpretation of sign language and to test its viability.

To achieve this objective, there are three main steps: build the necessary hardware to generate the movement data that represent the Portuguese Sign Language and build the means of sending this data to a database; collect the data generated by the hardware from different human sources to ensure that the generated data is not biased and guarantee the highest possible quality input for training the interpretation algorithms; and develop and apply one or more classification algorithms to verify that the generated data are of sufficient quality to generate a good degree of accuracy and prediction.

We hope at the end of the study to generate inputs that can feed future studies in the area of software and systems engineering on the use of low-cost sensors to generate quality data from accelerometers and gyroscopes, as well as validate algorithms already used in the statistical area.

## 1.5 Methodology

Currently, there are not many studies on the construction of equipment to perform sign language interpretation and even fewer public databases to train recognition algorithms. There is also no data standardization (quantity, data, or format) for this type of study. For this reason, this study must go through specific steps before being able to train recognition algorithms and carry out feasibility studies.

Considering the final objective of this study, which is to test the reliability of using low-cost sensors in sign language recognition, it is necessary to divide the study into three distinct stages:

**Hardware construction** is the first stage of the study, where the base hardware platform and the sensors that will be used in the data generation will be chosen and the equipment built. To choose the hardware and sensors, we will study the best options on the market, their characteristics, and, mainly, the cost of acquisition.

**Acquisition of data** is the second stage of the study, where the data generated by the sensors will be collected during the use of the hardware while the user performs Portuguese Sign Language

movements. At this stage, we will determine the minimum dictionary to be collected, approach different institutions linked to sign language users, present the project, and request the availability of volunteers to help with data collection, variability strategies in data acquisition, and initial analysis of the generated data.

**Data analysis** is the third stage of the study, where different types of classification algorithms will process the data generated by the hardware and collected during use by users, and it will be determined whether the generated data has sufficient quality to feed a classification and identification model of Portuguese Sign Language.

## 1.6 Document Structure

In addition to the introduction, this dissertation contains six more chapters. In Chapter 2, the state of the art is described and related works are presented. In Chapter 3, the technology, components, and construction schemes of the data acquisition hardware are described. Chapter 4 describes the methodology and quality criteria for acquiring the data used in this study. In Chapter 5 the algorithms used for classification and accuracy of the data collected against the recognition of Portuguese Sign Language are presented. In Chapter 6 the construction of a real-time sign language recognition system and its results are detailed, as well as the points of attention found. Chapter 7 presents the conclusions of the study and opportunities for future studies.

## Chapter 2

# Background and State of the Art

This chapter provides a review of the state-of-the-art in sign language recognition through hardware construction and recognition algorithms. The Section 2.1 presents a brief introduction of what is the basis of studies in the recognition of sign languages. The Section 2.2 details the different types of hardware that can be used to generate data that will be used for interpreting sign languages. The Section 2.3 details the most common recognition algorithms and models used in these studies. The Section 2.4 presents details on how to qualify a sign language recognition system and measure its accuracy and efficiency.

### 2.1 Introduction

Studies focused on the construction of methods, algorithms, software, and hardware that allow for performing sign language recognition have gained strength in the last 15 years. With the popularization of customizable and inexpensive hardware (e.g., Arduino) and techniques and algorithms for analyzing and interpreting data quickly (e.g., machine learning), exploring potential solutions for this problem is possible.

As the various existing sign languages are basically different from each other, different studies and solutions are carried out for each language (Kudrinko et al., 2021a). However, there are common concerns between the studies that connect the main approaches and developments in this area:

- What is the best way to capture, normalize, and store the data sets to perform the interpretation in a more accurate and efficient way? More data equals better quality, or is there an ideal balance between data - quality - performance?
- What is the best hardware to capture the data? Does having more sensors and data create a better result?
- What is the best algorithm for analysis and interpretation? Are slower but more accurate results better than faster results?

- How do I correctly calculate the accuracy of the recognition system?
- What are the efficiency and usability of the final product developed in a real-world usage scenario?

Although all these points are very important in any study on sign language recognition, the main focus of this study will be on the hardware. More specifically, whether low-cost sensors can generate data with sufficient quality to feed a Portuguese Sign Language recognition system.

## 2.2 Sign Language Recognition Methods - Hardware

Capturing the data that will be used by the recognition system is the starting point of any sign language recognition project. The quality and quantity of the data generated will be decisive in the success of the recognition algorithms. However, there is still no standard to be followed, and there are large differences in the type and amount of data between studies. Different equipment and hardware for capturing and generating data have been used in studies over the years, with some approaches using more refined hardware while others try to focus on usability. Below, we detail the main approaches regarding hardware and data capture.

### 2.2.1 Computer Vision

Computer vision is mainly used for its ease of implementation. In terms of hardware, all that is needed is a camera and a computer to analyze and process the captured images. The camera keeps a framed image of a person's entire body, or from the waist up, keeping in frame all the user's hands, body, and facial movements. The static image or sequence of images is analyzed, and the signals are identified (de Castro et al., 2023).

Despite its simple implementation, the use of computer vision has several negative points. First, there are obvious usability limitations, considering that the camera must be in a fixed user position during the entire usage, and that becomes even more challenging in everyday, uncontrolled environments where this technology could be used. Second, the quality of the captured image can be impacted and degraded by hardware and external factors, which could impair the reliability of the results obtained. We can cite as examples of internal and external factors the light interference, the drop in frames per second, and interference in the image by people or objects in focus (Mitra and Acharya, 2007).

Due to these limitations, this approach is used and applied in highly controlled environments. However, according to studies that use this approach, it generates a good level of quality in the recognition of sign language.

An advantage of this type of initiative is that by considering the image as input data, we create a pattern in which the algorithms can be trained. Therefore, there are already on the market several models of machine learning and other algorithms ready for this need. That is, this technology is easily interchangeable.



Figure 2.1: CyberGlove III. [cyberglovesystems.com](http://cyberglovesystems.com)

### 2.2.2 Commercial Gloves

Commercial gloves are hardware built for a wide variety of uses, such as virtual reality, human-machine interaction, and robotics. They have a wide range of sensors that generate data about position, movement and force. They are usually robust and built with good durability for intense use (Hong and Tan, 1989).

As commercial gloves are ready-made hardware, they are used in several research projects and studies because, without the need to build dedicated hardware, researchers can dedicate more time to build signal recognition algorithms.

There are several models on the market equipped with different types of sensors, which helps when finding the one that best suits the type of research being carried out. An example can be seen in Figure 2.1.

In this approach, facial and body expressions are not directly considered. It would be necessary to replace them with new hand movements to achieve complete integration.

A disadvantage of this type of equipment is usually the very high cost because it is industrial equipment. With costs that can reach USD \$20.000 per glove (Inc, 2023) (depending on the type of glove), it makes the use of this type of equipment by a common user very restrictive.

Another considerable disadvantage is the extremely low usability of this type of glove. What is perceived is that the glove effectively ends up disturbing fine movements performed to express some letters or phrases in sign language. Another point is that the glove would need to be constantly removed for the user to perform other activities, such as eating and handling small objects.

### 2.2.3 Blood Flow Sensors

Blood detection sensors currently exist in several wearable devices, such as smartwatches and physical activity bracelets. Typically used to analyze blood flow beneath the skin and determine the user's heart rate, they can generate a wider range of data for analysis.

Some studies have shown that these sensors are sensitive enough to determine changes in blood flow from people's wrists depending on the movement and placement of the fingers. That is, when one or more fingers move, the blood flow pattern in the main veins of the wrist undergoes slight changes with specific patterns. This data can be extrapolated to determine which finger was moved and at what intensity. This information can be used to determine finger movements in sign languages (Zhao et al., 2018).

In this approach, facial and body expressions are not directly considered. It would be necessary to replace them with new hand movements to achieve complete integration. But due to its lower sensitivity, this replacement could also be impacted.

Although promising, since they are inexpensive sensors and are currently found in several equipments already in use, their detection is limited to finger movement and is not influenced by wrist or body positioning, thus being limited to identifying the letters of a sign language rather than more complex movements. To detect more complex movements, it would be necessary to use inertial sensors on the wrist or fingers (Zhao et al., 2021) which increases the complexity of the system.

#### 2.2.4 Stretch Sensors

Stretch sensors, as shown in Figure 2.2, can measure the force with which they are stretched and contracted with very high accuracy. They are normally used in conjunction with gloves because they need a more robust anchorage for use, and the degree of accuracy depends on them not changing places during use (Ling Li and Gu, 2018).

These sensors can be attached to each finger to measure the closing or opening of the fingers individually. And together, they can generate enough data to determine the position of the fingers, equivalent to a movement referring to a sign language letter.

However, the use of this type of sensor suffers from a problem similar to that of blood sensors, as they alone cannot determine more complex gestural movements and become dependent on the concomitant use of inertial sensors to generate sufficient data.

In this approach, facial and body expressions are not directly considered. It would be necessary to replace them with new hand movements to achieve complete integration.

Another disadvantage is that these sensors have a more pronounced wear, because the generation of data is directly linked to the stretching and contraction of the material itself.

#### 2.2.5 Inertial Sensors

Inertial sensors, as shown in Figure 2.2, have been highlighted in this area of research in recent years. They are inexpensive, compact, low-power, easily found, and easily connected with other technologies (Gu et al., 2022).

Usually, these sensors can generate, individually or together, gyroscope data in three axes and acceleration data in three axes, but it depends on the type of sensor. The combination of gyroscope and accelerometer data is commonly referred to as an Inertial Measurement Unit, or IMU. Using

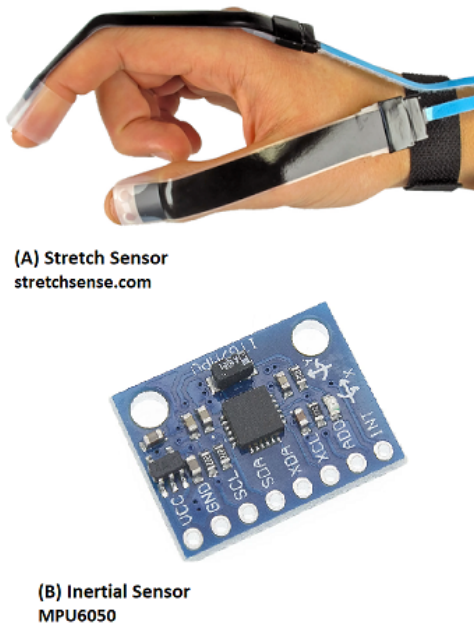


Figure 2.2: Various Sensor Types

this data, it is possible to make measurements of acceleration, velocity, orientation, displacement, and various other metrics related to motion.

In this approach, facial and body expressions are not directly considered. It would be necessary to replace them with new hand movements to achieve complete integration.

A disadvantage of using motion sensors is the number of different sensors available and the need for adaptation in the project that will use them, both in relation to the normalization of the generated data and the other technologies to which it will be connected.

### 2.2.6 Summary

There are several different types of sensors that generate a multitude of different types of data that can be used for such a study. We used data from other studies (Kudrinko et al., 2021b), forums, manufacturer's websites, online testimonials, and on-site tests to determine the characteristics of each type of sensor and compiled the results in Table 2.1.

Table 2.1: Hardware - Sensor Analysis

| Hardware / Sensor Type | Characteristics |           |        |
|------------------------|-----------------|-----------|--------|
|                        | Quality         | Usability | Cost   |
| Commercial Gloves      | High            | Low       | High   |
| Blood Flow Sensors     | High            | High      | Medium |
| Stretch Sensors        | Medium          | Medium    | Low    |
| Inertial Sensors       | High            | High      | Low    |

## 2.3 Classification Models and Algorithms for Sign Language Recognition

Creating an efficient model or algorithm for data classification turns out to be the most challenging step in any study involving sign language recognition. There are several approaches to doing this, and there is still no consensus on the best model or algorithm to perform the classification. In recent years, we have seen studies advance a lot in the use of Neural Networks, but more classic statistical techniques, such as Decision Trees and Random Forest, are more common. Data and signal recognition models and algorithms are present in basically all electronic devices for personal use today, such as smartphones, smartwatches, smartbands, and others.

According to a survey carried out by Karly Kudrinko, Emile Flavin, Xiaodan Zhu, and Qingguo Li (2021) 63.9% of studies in the area use classification models such as Naive Bayes, Decision Trees, Random Forest and Support Vector Machines (Kudrinko et al., 2021b). The use of classification models makes complete sense as the generated output is categorical and the training algorithm will use labeled data.

**Naive Bayes** is a statistical algorithm that generates a table of probabilities by data sorting. It manages to analyze different parameters in an integrated and separate way, working with probabilities, such as that of an event happening in the presence of another, but without being exclusive. Building the classification model is simple, and it is particularly fast in classification jobs. It has been used with promising results in recognition projects linked to gestures, speeches, and emotions (Anu Saini and Kadry, 2023).

**Decision Tree** is a classification algorithm based on a hierarchical structure built of root nodes, inner nodes, and leaf nodes, also called classes. The data enter the root of the tree, where they are segmented and classified separately and together according to the internal nodes until their final classification. As the classification is based on a hierarchical structure, there is a performance gain as the classification is performed (Zhang et al., 2011). The fast performance for a large volume of data makes the use of decision trees very appealing for systems that need to perform classification in real time, as is the case with a sign language recognition system.

**Random forest** is a classification algorithm similar to Decision Tree but based on the ensemble learning method, where a multitude of decision trees are constructed and the final class of choice for the parameters is determined by the class with the highest incidence of results. The creation of multiple decision trees (forests) is done randomly, with many decision trees, and for the construction of each of these trees, the data will not be used in full, but rather a selection of some samples of the data in a random way. To create tree nodes, there will also be a random step where some variables will be randomly selected. Despite producing more accurate results, considering that the small differences in the construction of each tree will reinforce the correct decisions of the final classes, the overall computational cost of this algorithm is higher.

**Support Vector Machines** are a supervised learning model with associated learning algorithms that analyze data for classification. It works by mapping points into an N-dimensional space, with the value of each feature being the value of a coordinate. It works in order to maximize

the gap between the coordinates and classifies the following points according to the proximity of the first mapped coordinates. It is a good-performing algorithm for small data sets but has a higher computational cost for larger data sets. Studies have shown that this algorithm is capable of generating results with greater accuracy than faster algorithms, such as Decision Tree (Wu et al., 2015).

## 2.4 Performance Metrics for Sign Language Recognition

Unlike the multiple approaches to building hardware and models of sign language recognition used in different studies, in this area there is a relative consensus on how to determine the overall effectiveness of a system in relation to sign recognition.

The effectiveness of a sign language recognition system can be measured by the accuracy of the classification model, the overall accuracy of the system, recognition time, and computational cost. These metrics end up adding great value to the system, but other metrics can be created for specific cases.

The *Accuracy of the Classification Model* is the classification accuracy found individually or for a set by each classification model. The accuracy of a classification model is calculated by comparing the number of correct predictions with the total number of predictions made on a given data set. It is a commonly used metric to evaluate the performance of a classification model. The accuracy is calculated using the following formula:

$$Accuracy = \frac{Number of Correct Predictions}{Total Number of Predictions} \quad (2.1)$$

The accuracy value represents the proportion of correct predictions made by the model on the given data set. It is important to note that accuracy alone may not be sufficient to evaluate the model's performance, especially in situations where the data set is imbalanced or when different classes have varying degrees of importance. In such cases, additional metrics like precision, recall, F1 score, or area under the receiver operating characteristic (ROC) curve may provide a more comprehensive assessment of the model's effectiveness.

The *Overall Accuracy of a System* aims to determine the number of successful recognitions in relation to the total number of attempts. The formula most commonly used to determine this metric is (Kudrinko et al., 2021b):

$$Acc = \frac{N - D - S - I}{N} \quad (2.2)$$

Where, N is the total number of words in the dictionary, D is the number of words deleted, S is the number of replacements, and I is the number of insertions.

The variable N will correspond to the total number of words, letters, or punctuation that are part of the total dictionary of the language and that are within the set of tests of the system. Creating sets that are more or less comprehensive than the actual sign language dictionary being used depends on the purpose and scope of the study. The variable D corresponds to the total

number of words, letters, or punctuation that were not recognized by the system. The variable  $S$  corresponds to the total number of incorrect identifications carried out by the system. Variable  $I$  corresponds to the total number of words, letters, or punctuation included during the test.

There are a large number of studies in the area that achieve an accuracy of 90% or more. However, the size of the dictionary is fundamental to knowing if the accuracy percentage is relevant or not. One cannot compare studies whose systems identify 20 signals with systems that identify 1000.

The **Recognition Time** is a determining factor in evaluating the final usability of the system. Despite not being present in many studies, this parameter determines the total time between the performance of the movement and its classification and identification by the system, and the shorter this time is, the more viable this system would be for use in a real-life scenario. The recognition time is directly linked to the algorithm or model used to perform the identification. Variations of more than 11x have been reported in some studies when using SOFM/HMM and Fuzzy Tree algorithms for the same set of dictionary and training data (Fang et al., 2004b).

The **Computational Cost** is directly linked to the time and effort performed by the signal recognition algorithm. Considering that, in real-world uses of this type of technology, the lower the effort and recognition time, the lower the energy cost, this is an important factor when dealing with embedded systems.

## 2.5 Summary

As we can see when detailing the various methodologies in sign language recognition studies, there are several types of equipment and hardware, possible approaches, different types of data, and several training models available. There is still no standardization in this field of study, which allows greater freedom to innovate in this area, but there are also still several approaches that simply should not generate positive or meaningful data.

Another important point is that no studies were found focusing on Portuguese Sign Language in conjunction with the feasibility of low-cost hardware. And it is to take advantage of this still open path in this area of study that the purpose of this study is not only to build a system (hardware + software) capable of recognizing Portuguese Sign Language but mainly to prove whether specific low-cost hardware is capable of generating data with sufficient quality to feed this system and generate reliable results.

## Chapter 3

# Hardware Assembly

In this chapter, we present the process of creating the necessary hardware to build a system capable of capturing and identifying Portuguese Sign Language letters, words, or actions. I will detail the choice of hardware, assembly details, the communication technology used, and the format of the data generated.

The creation of the hardware became one of the most challenging steps in this study, mainly because it required a range of knowledge that is not directly linked to software engineering. I started to think about the creation of the hardware by imagining the minimum criteria that I should follow:

- Use very low-cost components;
- The communication of the data generated by the sensors must be done wirelessly;
- The hardware should be easy to put on and take off;
- It must not have wires coming out of it for connection to other equipment external to the user's body: computers, cell phones, etc.;
- It cannot or should impact as little as possible on the movements of the user's hands and fingers.

A design was created that allows the generation and storage of positioning data for each finger individually, with the aim of ensuring that small variations in the data to represent different letters or words in Portuguese Sign Language are not lost and the final quality is not impaired.

Based on these criteria, the development platform, the sensors, and the general assembly of the equipment were chosen.

### 3.1 Platform

The platform on which the hardware was to be developed needed to be easy to use and acquire in the market. There was the possibility of developing our own customized boards and components



Figure 3.1: Development Board With ESP32-WROOM-32 Microcontroller

for this study. However, this would require time and costs that did not fit within this study. Among the existing options on the market, we chose to use components based on the Arduino platform.

Arduino is an open-source electronics platform that consists of hardware and software components designed to facilitate the creation of interactive and DIY projects. It comprises a programmable microcontroller board, a development environment, and a vast community of users and developers.

The Arduino board typically contains various input and output pins that can be connected to sensors, actuators, and other electronic components. It can be programmed using the Arduino programming language, which is a simplified version of C++, MicroPython, or through graphical programming interfaces with other programming languages.

Arduino is widely used by hobbyists, students, artists, and professionals in various fields to create a wide range of projects, such as robots, home automation systems, wearable devices, and much more. Its simplicity, affordability, and extensive library of pre-written code make it accessible to beginners while still offering flexibility for advanced users to develop complex applications.

## 3.2 Microcontroller

The microcontroller is the central part of the hardware and is responsible for managing all the sensors, recovering the data generated by them, and sending this data to a repository. The microcontroller chosen for this project was the ESP32 base model development board, which is manufactured by several different companies but always obeys the same standards and architecture, varying only the central processor and its wireless communication components and some internal components.

The development board used in this study has a microcontroller model ESP-WROOM-32, dual core with a maximum clock of 240MHz and has serial-USB connectivity, Wi-Fi (802.11 b/g/n (802.11n up to 150 Mbps), 2.4 GHz), and Bluetooth (Bluetooth V4.2 BR/EDR, Bluetooth LE) (Figure 3.1). It also has a USB to High Speed Serial Port Chip CH9102X, which can achieve good serial data transmission speeds.

This development board was chosen because it has a microprocessor capable of reaching sufficient clock speeds to handle five sensors at the same time and because it has Bluetooth Low Energy connectivity, which was the transmission method chosen to send the generated data to the computer that executes the algorithms, processing, and analysis.

The communication of the generated data will occur over the air by Bluetooth Low Energy (LE), allowing a longer battery life and a lower latency in relation to, for example, Wi-Fi communication.

The cost of this development board is around €4.00, making it an essential cost-benefit choice for this study.

### 3.3 Sensors

Sensors are the most important hardware component, as they will generate the data to be studied and interpreted. For this study, this was the most carefully chosen component.

For our study, we needed sensors that were small and inexpensive and that generated data that was specific enough for a classification algorithm to be able to discern between them. To identify a sensor that fulfilled these points, it was first necessary to decompose what we were actually trying to study: hand movements.

The movements that make up letters, words, actions, etc. in Portuguese Sign Language can be broken down basically into hand, body, and facial expression movements that carry a pre-established representativeness. Staying with the hand movements, we realize that the movements and positioning of the fingers and hands can be seen as acceleration forces in three axes and spatial positioning in three axes. This means, for example, that the hand to make the letter "A" needs to move in certain directions during a period of time and the data needs to be in a certain position in space during a period of time. Considering movement as something merely mechanical, we see that to perform the letter "A", the hand and the data generate spatial position data (gyroscope X, Y, and Z) and force (accelerometer X, Y, and Z).

With this definition, a sensor was chosen that, in addition to being cheap and small, could generate accelerometer and gyroscope data in three dimensions and be able to communicate with the microcontroller. The chosen sensor, which met all the criteria, was the **MPU-6050** Six-Axis (Gyro + Accelerometer) MEMS Motion Tracking (Figure 3.2).

This specific model of inertial sensor was developed by the TDK company and combines a 3-axis gyroscope and a 3-axis accelerometer on the same silicon die with a 16-bit measurement resolution, together with an onboard Digital Motion Processor™ (DMP™), which processes complex 6-axis MotionFusion algorithms (InvenSense, 2023).

This sensor can generate linear acceleration measurements along three axes (X, Y, and Z). It can sense changes in velocity, tilt, and motion, and it can also generate gyroscope angular velocity measures or rotational motion along the three axes. It also has an onboard DMP that offloads complex motion processing tasks from the host microcontroller and built-in motion detection functionality that can generate interrupts based on configurable motion thresholds.

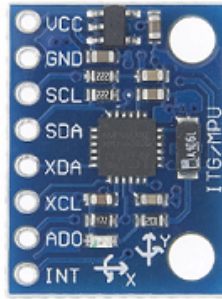


Figure 3.2: MPU 6050 (GY-521)

The MEMS accelerometers work using the basic principle of inertia. Inside the sensor are pairs of masses attached to a set of springs. Each mass provides the movable plate of a variable capacitance formed by an array of interlocking finger structures. When the object to which the sensor is attached is subjected to linear acceleration, the mass tends to stay at rest due to inertia before also starting to move by stretching or compressing the springs. Therefore, the mass and its fingers are displaced in relation to the fixed fingers of the electrode. The gas between the fingers provides a cushioning effect. This displacement induces a differential capacitance between the mobile and fixed silicon fingers that is proportional to the applied acceleration. This change in capacitance is measured with a high-resolution ADC and then the acceleration is calculated from the rate of change in capacitance. In the MPU-6050, this is converted into a readable value and then transferred to the I2C master device (Digest, 2023). A good way of validating the correct reading of the accelerometer data is that one of the axes must, on a static object, measure the earth's gravitational force at a constant rate.

The MEMS gyroscope works on the basis of the Coriolis Effect, which states that when a mass moves in a certain direction with velocity and an angular motion is applied to it, a force is generated, which causes a perpendicular displacement of the mass, where the rate of displacement will be directly related to the angular motion applied. This is the Coriolis Force, and this phenomenon is known as the Coriolis Effect. In the sensor, there is a set of four masses kept in continuous oscillatory motion. During an angular movement, there is a capacitance change between the masses, depending on the angular movement, due to the Coriolis Effect. The capacitance change is detected, converted into data, and then transferred to the I2C master device (Haratiannejadi et al., 2019) (Ziying et al., 2021).

The MPU-6050 communicates with the microcontroller using the I2C (Inter-Integrated Circuit) protocol, which is perfect for an Arduino-based microcontroller.

This sensor module also includes an embedded temperature sensor, which can provide the ambient temperature, but this will not be explored in this study.

In our study, we will use the GY-521 breakout board, which is already equipped with the MPU-6050 sensor (and other sensors), has dimensions of 21.2mm length x 16.4mm width, and is ready for connection with microcontrollers in the Arduino open hardware standard.

In general, it is a very effective sensor and is extremely affordable, with its direct cost being around €1.00 per sensor.

### 3.4 Integration

The main objective of the final assembled hardware is to have a platform that can be placed and removed with relative ease by users, does not generate negative impacts on the movement of hands and fingers, has its own power supply, and is able to transmit all data generated wirelessly.

The final hardware for use and data collection (for each hand) consisted of:

- One development Board with ESP32-WROOM-32 microcontroller;
- Five GY-521 sensors (MPU 6050);
- One TCA9548A Multiplexer board;

The development board with ESP32-WROOM-32 is powered by an external battery that provides up to 5 volts of power, enough to power the development board and all sensors. The development board has a 5V output and a 3.3V output, which are split in series to feed the multiplexer and the sensors. The 5V output powers three GY-521 sensors, and the 3.3V output powers two GY-521 sensors and a TCA9548A multiplexer.

As the development board does not have enough pins to receive input data from the five sensors at the same time, we use the TCA9548A multiplexer to route and select the signal for each of the sensors individually whenever we need to read a new measurement.

Each of the five sensors is connected to the multiplexer (serial clock and serial data), and the multiplexer is connected to the development board. At runtime and in a loop, the development board sequentially chooses each of the 5 routes where there are sensors connected, retrieves the gyroscope and accelerometer data generated at that moment by that specific sensor, and moves on to the next sensor. After reading five sensors, the data is grouped and sent to an external repository.

The final hardware construction schematics can be seen in Figure 3.3.

### 3.5 Communication

The communication between the sensors, the multiplexer, and the development board takes place through the I2C standard and uses Jumper Dupont cables to connect the pins of the boards. Attaching the cables to the connection pins with solder would give the hardware greater durability, but making the connections using jumper cables is sufficient for prototyping and testing.

The microcontroller performs a sensor data reading loop every 120 milliseconds, selecting the reading sensor and recovering the data generated by it with an interval of 7 milliseconds between each sensor change. After reading the data generated by the fifth and last sensor, the data is grouped into a single string and sent to an external repository to be stored.

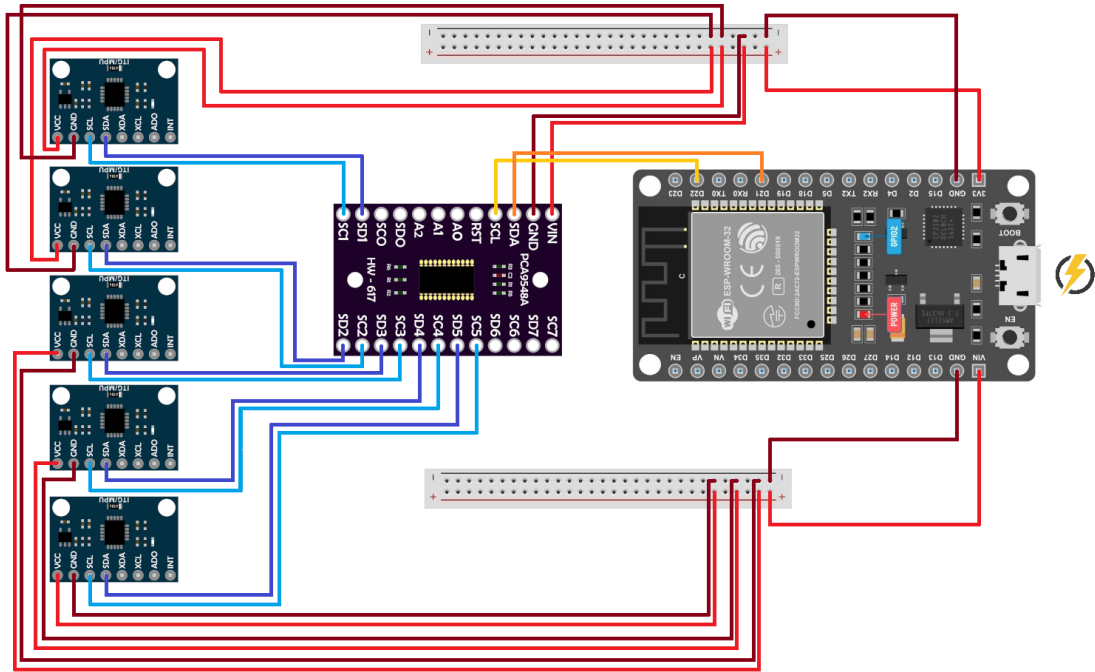


Figure 3.3: Schematics

Data is sent to the external computer via Bluetooth at an average rate of 5 readings per second. The sensors and the development board manage to generate a larger volume of data that would be easily read if the transmission was physical (USB cable), but since the transmission is via Bluetooth, the volume of data is limited to avoid problems of data congestion when sending and reading.

A computer, which is connected to the development board via Bluetooth, reads the bytes received in a loop, and whenever it receives a complete string (containing X, Y, and Z data for the gyroscope and accelerometer for each of the five fingers of a hand), it writes this data to an SQLite database. It is under this database that the analysis and classification processes take place.

### 3.6 C++ Arduino Code

For programming and code execution, the development board supports the use of C++ (Arduino base), MicroPython, and LUA. For this study, I used C++ code (Arduino base) because it has the largest online knowledge base in the DIY community.

We also used some Adafruit (Adafruit MPU6050.h and Adafruit Sensor.h) public libraries built specifically to work with MPU-6050 sensors, saving time in development. The sample code can be seen in Appendix A.

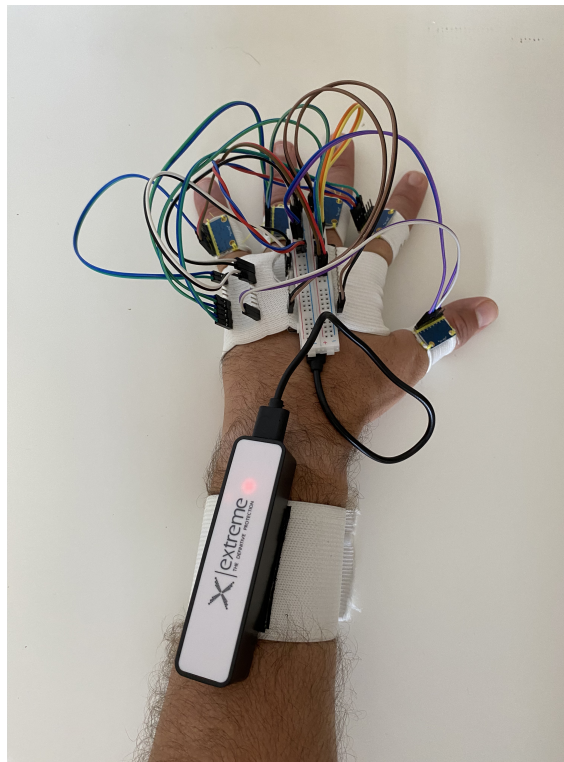


Figure 3.4: Final Hardware - Left Hand

### 3.7 Summary

The biggest difficulty faced during the hardware construction step was the acquisition of the necessary knowledge to interconnect all the components in a way that would generate quality data. Another challenge was to write the algorithm in C++ (for Arduino), considering that all the examples found online for this type of hardware only considered one sensor and not five working together.

Another challenge was understanding how serial communication between Bluetooth devices works, as this is a completely different area of knowledge in programming.

Briefly, to assemble the final hardware, it was necessary to have knowledge of soldering (to put together the boards and their connection pins), Arduino (to understand how the components work and how the I2C communication between components takes place), programming in C++ (to write the code that runs on the microcontroller), Python (to write the code that runs on the computer that receives the data via Bluetooth), and the SQLite database (used to store the data).

The final product of the hardware analysis and construction process needed for the study was equipment that operated independently of each other, generated gyroscope and accelerometer data individually for each finger of each hand, and sent this information to an external repository over Bluetooth. With an average generation of five sets of data per second for each hand, I hope to be able to detect small variations in positioning and movement and increase the quality of identification and analysis of the generated data.

## Chapter 4

# Data Acquisition

In this chapter, we present the steps and processes related to the capture and use of data generated by the hardware with Portuguese Sign Language users and how it will feed the statistical recognition models.

We will detail how we created the base dictionary used in this study and how the data was captured. I will also detail how the data are structured and some initial analyses performed on the data.

### 4.1 Creation of the Work Dictionary

During the preparation of this study and the analysis of other studies in the same area, it became evident that the correct acquisition of data is one of the most important and critical points for a reliable result. As the study is based on Portuguese Sign Language and uses a specific MPU-6050 sensor for data acquisition, it was not possible to find any public or private database already compiled and normalized for the execution of this study. All training and test data acquisition and normalization must be done during the execution of this study.

The first step in data acquisition was to determine the scope of the study in relation to the complete dictionary of Portuguese Sign Language. The Portuguese Sign Language has 26 signs for letters of the alphabet and more than 25.000 other entries to determine actions, objects, characteristics, and other components of a language. For this study, we will limit data acquisition to the 26 letters of the alphabet and 20 words divided between actions and objects, as detailed in Table 4.1. All words chosen for the study dictionary were validated primarily using a Portuguese Sign Language dictionary (Infopedia, 2023b).

As data acquisition is a time-consuming task, the objective of reducing the dictionary to be used is to allow more concise and faster work in data acquisition and in the construction and training of recognition models.

For data acquisition, we contacted Portuguese Sign Language teaching institutions and support institutions for people with hearing impairments. In order to have a good variety of data and be able to make comparisons, it would be necessary to have a mix of data captured by users not

Table 4.1: Specific Dictionary and Data Acquisition

| Type    | Data                                                                            |
|---------|---------------------------------------------------------------------------------|
| Letters | A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, X, Y, W and Z |
| Actions | Give, Take, Walk, Go, Stop, Skip, Write, Buy, Ask and Ignore                    |
| Objects | Home, Mobile Phone, Car, Boat, Computer, Book, Key, Bed, Television and Chair   |

trained in Portuguese Sign Language as well as data captured by people who are already fluent in Portuguese Sign Language and use it in their daily lives. The main institution currently involved with the study is the ASP - Associação de Surdos do Porto (ASP, 2023) for which we presented the studies and got support in identifying native sign language users available to use the hardware and allow data capture.

When studying the use of Portuguese Sign Language, we realized that there are small differences in the execution of sign language movements by each person. These differences vary depending on the user, their physical constitution (height, gender), and their own particular interpretation and understanding of each sign language movement.

Communication between two sign language users involves a great deal of interpretation beyond the accuracy of the movement itself. Therefore, to guarantee a good set of data, we decided to segment the acquisition into two characteristics to be fulfilled: *Quantity and quality*.

To achieve a good *quantity* of data, we will capture the data for each item in the dictionary in the following pattern: for each letter of the alphabet, the hand position will be captured and maintained in the shape of the letter between 5 and 10 seconds; for each word and action, the movement will be captured three times in sequence with a difference of one second between each capture. And each dictionary item will be captured by four people participating in the study, two fluent in sign language and two non-fluent. In this way, we will not only be able to perceive differences in the execution of a specific movement by the same person but also between different people. In order to guarantee the *quality* of the data, we will capture the movements simultaneously with a video capture of the participants. A tag will be added to the video containing the date, hour, minute, and second of the shooting moment. The objective is, after capturing and during the data preparation phase for training and testing the recognition algorithm, to have a clear indicator of the initial and final moments of the movement, thus preventing the algorithm from being trained with noise data that will come together with the good data of the movement itself.

To have data that allowed a good level of abstraction by the recognition algorithm in the training phase, the data was collected from four volunteers (two men and two women), of whom one man and women were native speakers of Portuguese Sign Language and others were not. For non-native speakers, only the letters of the alphabet were captured.

Data collection was performed using one or two sets of capture hardware simultaneously. When the sign language involved only one of the hands, in the data processing phase, the data

generated by the other hand was ignored.

## 4.2 Data Structure

Each piece of hardware generates a set of three gyroscope data (x, y, and z) and three accelerometer data (x, y, and z) for each finger of each hand for each measurement performed. Gyroscope measurements are made in rad/s and accelerometer measurements are made in  $m/s^2$ , and all values are limited to a precision of 2 decimal places.

In Table 4.2, the result of a measurement of a sensor for the thumb of the right hand is shown.

Table 4.2: Example of Collected Data - Thumb Finger - Right Hand

| Finger ID | Finger Description              | Value |
|-----------|---------------------------------|-------|
| ax1       | Accelerometer X Axis Finger One | 5.76  |
| ay1       | Accelerometer Y Axis Finger One | 6.53  |
| az1       | Accelerometer Z Axis Finger One | 4.31  |
| gx1       | Gyroscope X Axis Finger One     | 0.04  |
| gy1       | Gyroscope Y Axis Finger One     | -0.01 |
| gz1       | Gyroscope Z Axis Finger One     | 0.02  |

The data generated for the right and left hands is stored in separate tables in SQLite databases. The process was built this way so that in situations where both hands are used at the same time, and considering that the data flow is in real time, there would be no competition from processing and recording the data in the base. In the database, for each measurement, a single line is created containing the accelerometer and gyroscope data of all fingers. In this way, the statistical models used in the data analysis will have visibility into the positioning parameters of all the fingers of each hand throughout the measurements (timeline).

For each measurement line added to the database, a unique identifier column and a column with the date (MM/DD/YYYY HH:MM:SS) on which the measurement took place are included. Time precision was not extended to milliseconds because we already had a column with a unique identifier for each measurement, which would end up being a redundancy.

The final result of a set of measurements performed on the right hand can be seen in Figure 4.1.

| id   | inputdatetime       | ax1   | ay1   | az1   | gx1   | gy1   | gz1   | ax2  | ay2   | az2   | gx2   | gy2   | gz2   | ax3  | ay3   | az3   | gx3   | gy3   | gz3   | ax4  | ay4   | az4   | gx4   | gy4   | gz4   | ax5  | ay5   | az5   | gx5   | gy5   | gz5   |
|------|---------------------|-------|-------|-------|-------|-------|-------|------|-------|-------|-------|-------|-------|------|-------|-------|-------|-------|-------|------|-------|-------|-------|-------|-------|------|-------|-------|-------|-------|-------|
| 1025 | 15/05/2023 17:10:04 | -5.02 | -4.37 | -7.68 | 0.01  | 0.02  | -0.07 | 4.54 | -3.49 | -8.8  | 0.02  | 0.08  | -0.02 | 4.92 | -5.49 | -8.43 | 0.05  | -0.02 | -0.06 | 4.81 | -4.74 | -7.91 | -0.12 | -0.09 | 0.02  | 4.48 | -4.93 | -8.49 | 0.09  | -0.05 | -0.02 |
| 1026 | 15/05/2023 17:10:04 | -5.01 | -4.46 | -7.8  | -0.02 | -0.01 | 0.32  | 4.67 | -3.38 | -8.71 | -0.3  | -0.08 | 0.14  | 5.04 | -5.32 | -8.27 | -0.28 | -0.07 | 0.05  | 4.84 | -4.55 | -7.66 | -0.44 | -0.13 | 0.06  | 4.45 | -4.71 | -8.17 | -0.21 | -0.05 | 0.06  |
| 1027 | 15/05/2023 17:10:04 | -5.05 | -3.99 | -7.93 | -0.01 | -0.04 | 0.01  | 4.58 | -3.13 | -9.01 | -0.05 | -0.03 | -0.01 | 4.97 | -5.12 | -8.57 | -0.04 | -0.08 | -0.05 | 4.77 | -4.31 | -7.88 | -0.2  | -0.11 | -0.03 | 4.44 | -4.51 | -8.32 | 0.02  | -0.04 | -0.02 |
| 1028 | 15/05/2023 17:10:05 | -5.16 | -4.26 | -7.68 | -0.07 | -0.03 | 0.03  | 4.38 | -3.47 | -8.93 | -0.09 | -0.02 | -0.04 | 4.68 | -5.43 | -8.6  | -0.07 | -0.08 | -0.08 | 4.46 | -4.7  | -8.05 | -0.25 | -0.13 | -0.03 | 4.06 | -4.96 | -8.64 | -0.05 | -0.07 | 0.01  |
| 1029 | 15/05/2023 17:10:05 | -4.74 | -4.18 | -8.05 | -0.27 | 0.03  | 0.27  | 4.79 | -2.81 | -8.65 | -0.41 | -0.05 | -0.14 | 5.19 | -4.51 | -8.22 | -0.31 | -0.05 | -0.28 | 5.02 | -3.6  | -7.53 | -0.42 | -0.07 | -0.27 | 4.69 | -3.76 | -8.13 | -0.14 | -0.02 | -0.27 |
| 1030 | 15/05/2023 17:10:05 | -5.97 | -3.53 | -7.79 | -0.02 | -0.17 | 0.11  | 3.91 | -2.92 | -9.67 | -0.11 | -0.15 | 0.03  | 4.53 | -4.85 | -9.31 | -0.13 | -0.18 | -0.02 | 4.44 | -3.93 | -8.56 | -0.32 | -0.22 | -0.02 | 4.19 | -4.02 | -8.95 | -0.11 | -0.13 | -0.03 |
| 1031 | 15/05/2023 17:10:06 | -5.9  | -3.28 | -7.65 | -0.03 | 0.08  | -0.02 | 3.76 | -2.65 | -9.56 | -0.08 | 0.1   | -0.04 | 4.48 | -4.59 | -9.32 | -0.03 | 0.05  | -0.09 | 4.44 | -3.72 | -8.62 | -0.2  | 0.01  | -0.05 | 4.26 | -3.83 | -9.1  | 0.03  | 0.07  | -0.04 |
| 1032 | 15/05/2023 17:10:06 | -5.39 | -3.58 | -7.91 | 0.04  | 0.04  | -0.01 | 4.33 | -2.76 | -9.33 | -0.03 | 0.06  | 0.02  | 4.86 | -4.78 | -9.06 | -0.01 | 0.0   | -0.03 | 4.73 | -3.95 | -8.4  | -0.18 | -0.05 | 0.01  | 4.44 | -4.07 | -8.92 | 0.03  | 0.0   | 0.01  |
| 1033 | 15/05/2023 17:10:06 | -5.37 | -3.46 | -8.04 | -0.01 | 0.03  | 0.01  | 4.44 | -2.62 | -9.31 | -0.06 | 0.06  | -0.01 | 5.05 | -4.7  | -9.01 | -0.03 | -0.01 | -0.06 | 4.96 | -3.92 | -8.36 | -0.2  | -0.07 | -0.02 | 4.67 | -4.09 | -8.88 | 0.01  | -0.03 | -0.02 |
| 1034 | 15/05/2023 17:10:06 | -5.48 | -3.41 | -7.86 | 0.03  | -0.01 | 0.09  | 4.16 | -2.61 | -9.37 | -0.1  | 0.0   | 0.06  | 4.79 | -4.6  | -9.13 | -0.09 | -0.03 | 0.01  | 4.74 | -3.71 | -8.43 | -0.27 | -0.08 | 0.04  | 4.55 | -3.81 | -8.93 | -0.05 | -0.02 | 0.05  |

Figure 4.1: Measuring Set - Right Hand

### 4.3 Initial Analysis

The objective of this initial analysis was to, even before submitting the generated data to classification models to test the feasibility and quality of the data generated by the sensors, identify possible failures in the way of storing and grouping the collected data and, mainly, to verify the quality of the data generated by the sensors and the need for changes in hardware or replacement of defective components.

#### 4.3.1 Gravity Force

The first test performed was on the quality of the accelerometer data generated by the sensor. The MPU-6050 sensor does not directly measure the force of gravity but can be used to estimate an object's orientation and inclination with respect to gravity. Through the accelerometer data created in the MPU-6050, it is possible to determine the gravitational acceleration component along the X, Y, and Z axes.

The gravitational acceleration is approximately equal to  $9,80m/s^2$  and acts in the opposite direction to the gravitational vector. By placing the MPU-6050 in a static position without movement, the accelerometer reading will provide a value proportional to the gravitational acceleration along the Z axis. This inclination information can be used to estimate the force of gravity, considering that the gravitational acceleration acts directly along the Z axis.

Considering that in this study we will not be using any filtering or compensation algorithm, this validation could determine the quality of the data generated by the sensor.

To generate the test data, a sensor was left at rest on a table in a horizontal position with as little external interference as possible. Accelerometer data generated by the sensor on the X, Y, and Z axes was collected for 10 seconds, totaling 62 different measurements. Analyzing the result, we realized that a constant force with an average value of  $9,29m/s^2$  is exerted on the Z axis, very close to the standard  $9,80m/s^2$  of the acceleration due to gravity. The variation detected during the measurement was a maximum of  $9,30m/s^2$  for more and  $9,26m/s^2$  for less, representing respectively a variation of 0.11% and -0.32%, which we consider small and totally acceptable for a low-cost sensor. In Figure 4.2 we detail the analyzed data.

#### 4.3.2 Noise

The second test performed was on the volume of noise present in the gyroscope data generated by the sensor. The three-axis gyroscope, which measures angular velocity or rotational motion along the X, Y, and Z axes, uses microelectromechanical system (MEMS) technology based on the Coriolis effect to sense changes in rotation.

The gyroscope data, typically in the form of 16-bit signed integers representing the angular velocity along each axis, should be stable as long as there is no change in sensor orientation. Keeping the sensor stable, the noise variation level should be zero or close to zero. This would

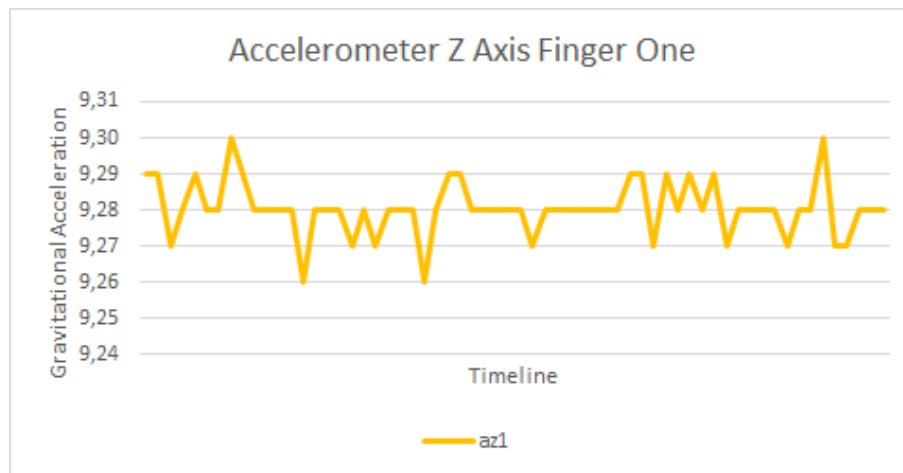


Figure 4.2: Constant Gravitational Analysis - Z Axis Accelerometer

guarantee that the sensors only generate gyroscope changes against the real changes in their positioning, avoiding false positives and noise that could generate a loss of quality in the sign language recognition algorithms.

To generate the test data, a sensor was left at rest on a table in a horizontal position with as little external interference as possible. Gyroscope data generated by the sensor on the X, Y, and Z axes was collected for 10 seconds, totaling 62 different measurements. Analyzing the result, we realized that on the X, Y, and Z axes, the values remained constant throughout the time. No variation was detected in the generated data until external forces effectively changed the position of the sensor, which we consider fully acceptable for a low-cost sensor. In Figure 4.3, we detail the analyzed data.

### 4.3.3 Timeline and Motion Equivalence

The third test carried out was to determine whether, for a specific word and action of the Portuguese Sign Language, considering the data generated in relation to quantity and quality, it would be sufficient to identify a visible movement pattern and whether this pattern would be repeated. That is, we wanted to identify whether, when looking at the data generated at different times for the same movement, they would be visibly similar and equivalent. This similarity would exemplify a good quality of data acquisition and would increase the reliability of the data with which the classification algorithm would work.

To generate test data, we asked a Portuguese Sign Language user to perform the movement that represents the action "jump" three times, with an interval between each execution. As exemplified in Figure 4.4, the movement for the word "jump" in Portuguese Sign Language consists of holding the left hand open, facing it with the palm up, and standing still, while the right hand performs with the index and middle fingers a movement similar to a person jumping.

The gyroscope and accelerometer data generated by the sensors for the X, Y, and Z axes and for each hand, after being collected, represent between 22 and 23 measurements for each execution

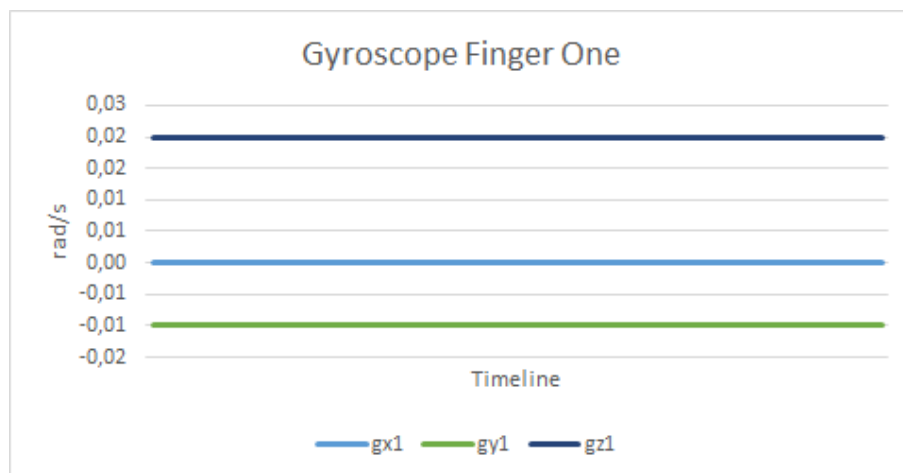


Figure 4.3: Gyroscope Noise Analysis



Figure 4.4: Word "Jump" Details Movements

of the movement (A and B), which last about 4 seconds.

By plotting the sensor data in graphs, we can visually discern that it is possible to see the execution of the movement in the timeline, as well as the similarity between movement A and movement B, both in the accelerometer data and in the gyroscope data. It is also visible that, even for the Portuguese Sign Language user performing the same movement, differences in strength, intensity, and positioning of fingers and hands exist and are perceptible.

In Figure 4.5, it is possible to verify the equivalences of the data generated for executions A and B of the "jump" movement by comparing the data and both hands for gyroscope and accelerometer.

## 4.4 Summary

The data collection process proved to be an excellent opportunity to initially understand Portuguese Sign Language in details that are not found in books or websites. By talking to real Portuguese Sign Language users, we were able to perceive important details and adapt the data collection process to better reflect reality.

Collecting data from users fluent in Portuguese Sign Language, as well as from people with no experience, allows us a deeper analysis of the subtle details of the language and will allow us to feed the classification algorithms with more comprehensive data.

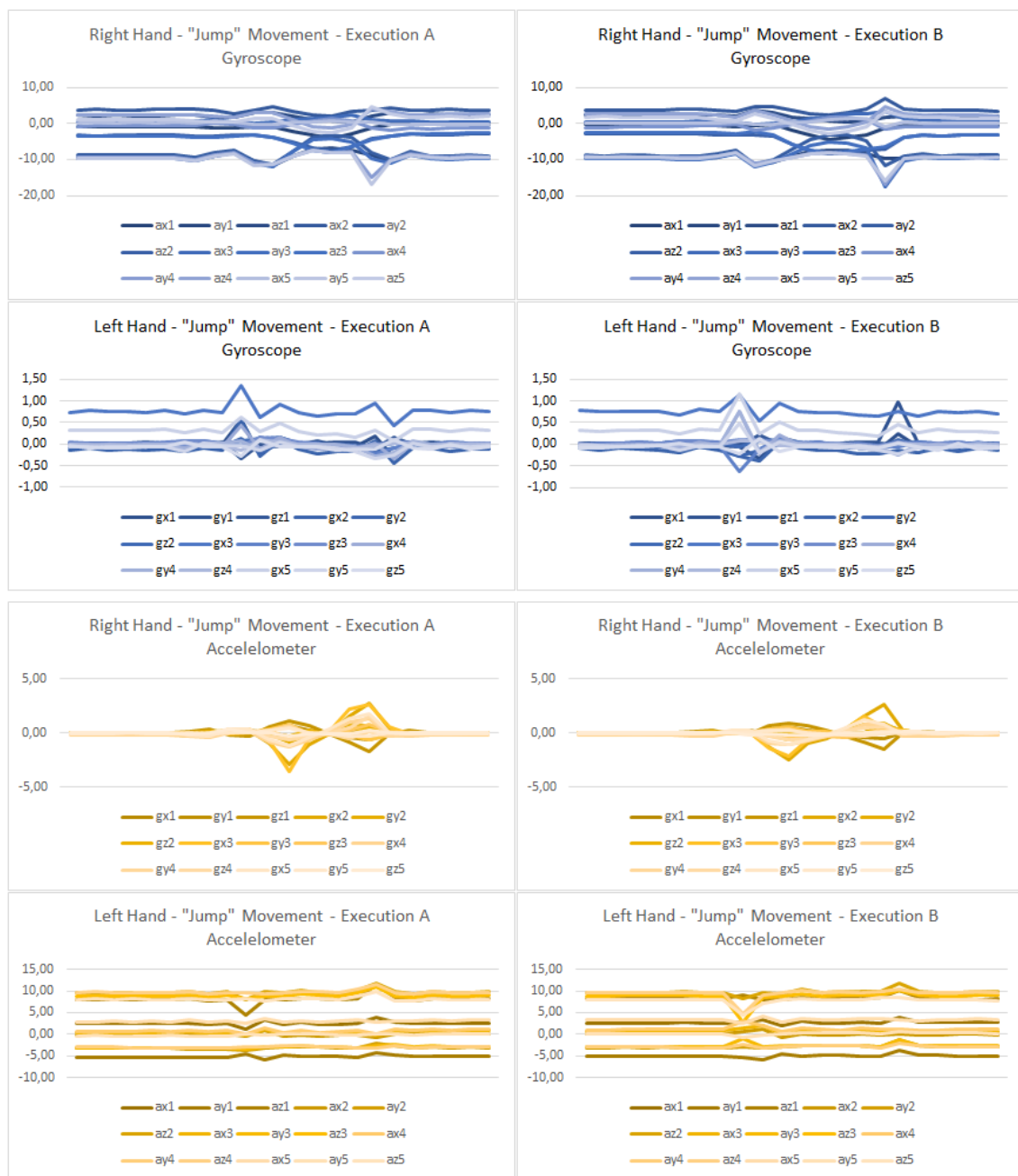


Figure 4.5: Word "Jump" Data Comparison

With the initial analysis of the data, it was already possible to perceive subtle differences in the collected data, as well as verify that the data generated by the MPU-6050 sensors are sufficiently accurate to allow us to proceed with the study and test different classification algorithms to verify if, in addition to simple examples, the sensors can feed a more robust Portuguese Sign Language recognition system.

## Chapter 5

# Analysis and Recognition Algorithms

In this chapter, the analysis, construction, and application of different classification algorithms to the data generated by the hardware built to capture Portuguese Sign Language usage will be detailed. Also included are the final data set configuration, the efficiency of the different identification algorithms, and the performance and comparison of the obtained results.

### 5.1 Classification Hardware

The classifications carried out in this chapter were made on a notebook equipped with an Intel Core i7 8550U (8th Gen, 4 cores) CPU and 16 GB of DDR4 memory. Any consideration of the classification speed of each algorithm must take this configuration into account.

### 5.2 Data

The database submitted to the classification algorithms is a subset of the data generated from multiple captures carried out by two people fluent and two non-fluent in Portuguese Sign Language. The data capture process, detailed in chapter 4, involved performing sign language signs with one or two hands, gesturing a list of 26 letters of the Portuguese Sign Language alphabet, 10 words representing actions, and 10 words representing objects. Details of the specific dictionary created for this study can be seen in Table 4.1.

The result of the capture process was two data sets, one for the left hand and one for the right hand.

The original database, before data sanitization and with no labels, had 12.779 records for the right hand and 5.004 for the left hand. The number of records for the right hand is greater because the letters in the dictionary were recorded only with the right hand.

#### 5.2.1 Data Sanitization

After capturing the data, the database was cleaned in order to only keep data that was significant for the test and that could be classified as a Portuguese Sign Language item. Initially, all data that

did not represent a specific signal from the dictionary created for this study was removed, such as intervals between performed signals, impacting errors on data recovery from the sensors, and repetition of movements.

### 5.2.2 Labeling

After cleaning, the remaining data set was labeled to identify the meaning of each record. The labeling of movements is done considering the movement as the data from beginning to end captured for that specific movement, with each record labeled individually with the same information.

The labeling process was made based on videos recorded during data capture sessions. Recorded videos have a date, hour, minute, and second tag, helping to identify exactly where the movement started and where it ended.

After labeling the data, three data sets were created for use with the classification algorithms: a data set of objects and actions, with 2.006 records for the right hand and 1.858 for the left hand; a data set of letters of the alphabet, with 8.792 records for the right hand; and a sub-data set of letters of the alphabet, containing only data captured with long exposure for each letter, with 5.314 records for the right hand. A final compilation can be seen in Table 5.1.

Table 5.1: Final Data sets For Classification Using Algorithms

| Data set                 | Description                                                                                                                                                                                                     | Size  |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|
| Words<br>Left Hand       | Data set containing data captured by the hardware and classified for movements performed by the left hand for the 20-words dictionary (10 actions and 10 objects) by two native Portuguese Sign Language users  | 2.006 |
| Words<br>Right Hand      | Data set containing data captured by the hardware and classified for movements performed by the right hand for the 20-words dictionary (10 actions and 10 objects) by two native Portuguese Sign Language users | 1.858 |
| Letters                  | Data set containing data captured by the hardware and classified for movements performed by the right hand for the 26-letter dictionary by two native Portuguese Sign Language users and two non-native users   | 8.792 |
| Letters<br>Long Exposure | Specific data set, based on the Letters data set, containing only data captured from letters through long exposure of movements                                                                                 | 5.314 |

### 5.2.3 Errors and Flaws in the Acquired Data

During the data sanitization and labeling steps, we identified problems in the data generated by the hardware that can be differentiated as having a partial or total impact on the quality and use of the data.

Partial impact errors are usually characterized by one or two sensors failing to return, during the execution of a movement, the correct accelerometer or gyroscope data for that particular movement. That is, during some records, between one and two sensors generated zero data. In cases where 40% of the sensor data failed or less, we kept the data as valid in order to further study the impact of this failure on the ability of the algorithms to perform the identification. Total impact errors are characterized when more than 40% of the sensors fail to send data. In these situations, we considered the data "invalid" and discarded it.

The items of the study dictionary impacted by a total failure were the objects "Book", "Key", "Bed", "Television" and "Chair" generated for the individual male fluent in Portuguese sign language and only for the left hand data. Because of this, data from the left hand will not be used in the remainder of this study.

### 5.3 Decision Tree

The decision tree algorithm is a popular and widely used machine learning algorithm that is used for both classification and regression tasks. It is a simple yet powerful model that builds a tree-like structure to make predictions based on input features.

For this study, we used the Python language and the Scikit-learn module, commonly referred to as sklearn. It is a popular machine-learning library for Python. It provides a wide range of tools and algorithms for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and model selection.

The execution of the Decision Tree algorithm on the Objects and Actions data set, Letters data set and Letters (Long exposure) data set was carried out in three different methods, with the aim of achieving the best possible classification.

- The first classification method was performed using a separation of 30% for training and 70% for classification (test size = 0.3) and accelerometer and gyroscope data were considered.
- The second classification method was performed using a separation of 40% for training and 60% for classification (test size = 0.4) and accelerometer and gyroscope data were considered.
- The third classification method was performed using a separation of 30% for training and 70% for sorting (test size = 0.3). However, for this test only gyroscope data were considered.

The Random State (random state = 1) and shuffle (shuffle = True) parameters were also activated for all tests.

#### 5.3.1 Objects and Actions Classification

The results obtained with the Decision Tree algorithm, for the three methods, for classifying Objects and Actions are shown in table 5.2. It is possible to notice that there is not a relevant variation in the classification time but that there is a great degradation in accuracy when using only the gyroscope data in the classification.

The best classification results were obtained in method 1, using a separation of 30% for training and 70% for classification (test size = 0.3) and accelerometer and gyroscope data, with a accuracy of 72,3%, as detailed in table 5.3.

Table 5.2: Result of the Decision Tree Classification Model For Objects And Actions

| Method | Training/Classification Ratio | Considered Data           | Other parameters                   | Accuracy         | Classification Time |
|--------|-------------------------------|---------------------------|------------------------------------|------------------|---------------------|
| 1      | 30% / 70%                     | accelerometer + gyroscope | random state = 1<br>shuffle = True | 72,3%<br>(0.723) | <0.1s               |
| 2      | 40% / 60%                     | accelerometer + gyroscope | random state = 1<br>shuffle = True | 69,7%<br>(0.697) | <0.1s               |
| 3      | 30% / 70%                     | gyroscope                 | random state = 1<br>shuffle = True | 19,1%<br>(0.191) | <0.1s               |

Table 5.3: Best Result Decision Tree (30%/70%) For Objects and Actions Classification Using Accelerometer And Gyroscope Data

| Target              | Precision   | Recall      | F1-Score    | Support    |
|---------------------|-------------|-------------|-------------|------------|
| WALK                | 0.74        | 0.74        | 0.74        | 27         |
| BOAT                | 0.90        | 0.70        | 0.79        | 40         |
| CHAIR               | 0.71        | 0.73        | 0.72        | 30         |
| BED                 | 0.57        | 0.75        | 0.65        | 32         |
| CAR                 | 0.82        | 0.87        | 0.84        | 31         |
| WITH                | 0.82        | 0.55        | 0.65        | 33         |
| KEY                 | 0.60        | 0.80        | 0.69        | 15         |
| PURCHASE            | 0.62        | 0.80        | 0.70        | 20         |
| COMPUTER            | 0.68        | 0.63        | 0.66        | 30         |
| IN                  | 0.80        | 0.80        | 0.80        | 30         |
| WRITE               | 0.57        | 0.75        | 0.65        | 28         |
| IGNORE              | 0.90        | 0.79        | 0.84        | 24         |
| GO                  | 0.64        | 0.76        | 0.69        | 33         |
| BOOK                | 0.59        | 0.65        | 0.62        | 20         |
| STOP                | 0.73        | 0.65        | 0.69        | 37         |
| TAKE                | 0.84        | 0.88        | 0.86        | 41         |
| ASK                 | 0.72        | 0.68        | 0.70        | 31         |
| JUMP                | 0.88        | 0.85        | 0.87        | 27         |
| CELLPHONE           | 0.87        | 0.67        | 0.75        | 39         |
| TV                  | 0.53        | 0.50        | 0.52        | 34         |
| <b>Macro Avg</b>    | <b>0.73</b> | <b>0.73</b> | <b>0.72</b> | <b>602</b> |
| <b>Weighted Avg</b> | <b>0.74</b> | <b>0.72</b> | <b>0.72</b> | <b>602</b> |

|                 |      |     |
|-----------------|------|-----|
| <b>Accuracy</b> | 0.72 | 602 |
|-----------------|------|-----|

### 5.3.2 Letters Classification

The results obtained with the Decision Tree algorithm, for the three methods, for classifying Letters are shown in table 5.4. It is possible to notice that there is not a relevant variation in the classification time but that there is a notable degradation in accuracy when using only the gyroscope data in the classification.

Table 5.4: Result of the Decision Tree Classification Model For Letters

| Method | Training/Classification Ratio | Considered Data           | Other parameters                   | Accuracy         | Classification Time |
|--------|-------------------------------|---------------------------|------------------------------------|------------------|---------------------|
| 1      | 30% / 70%                     | accelerometer + gyroscope | random state = 1<br>shuffle = True | 94,9%<br>(0.949) | 0.1s                |
| 2      | 40% / 60%                     | accelerometer + gyroscope | random state = 1<br>shuffle = True | 94,6%<br>(0.946) | 0.1s                |
| 3      | 30% / 70%                     | gyroscope                 | random state = 1<br>shuffle = True | 44,1%<br>(0.441) | 0.1s                |

The best classification results were obtained in method 1, using a separation of 30% for training and 70% for classification (test size = 0.3) and accelerometer and gyroscope data, with a accuracy of 94,9%, as detailed in table 5.5.

Table 5.5: Best Result Decision Tree (30%/70%) For Letter Classification Using Accelerometer And Gyroscope Data

| Target           | Precision   | Recall      | F1-Score    | Support     |
|------------------|-------------|-------------|-------------|-------------|
| A                | 0.96        | 0.97        | 0.97        | 154         |
| B                | 0.99        | 0.97        | 0.98        | 134         |
| C                | 0.98        | 0.98        | 0.98        | 99          |
| D                | 0.93        | 0.86        | 0.90        | 111         |
| E                | 0.94        | 0.98        | 0.96        | 92          |
| F                | 0.96        | 0.96        | 0.96        | 67          |
| G                | 0.94        | 1.00        | 0.97        | 47          |
| H                | 0.96        | 0.99        | 0.97        | 69          |
| I                | 0.96        | 0.98        | 0.97        | 65          |
| J                | 0.98        | 0.98        | 0.98        | 100         |
| K                | 0.93        | 0.93        | 0.93        | 88          |
| L                | 0.94        | 0.91        | 0.92        | 120         |
| M                | 0.96        | 1.00        | 0.98        | 102         |
| N                | 0.98        | 0.97        | 0.98        | 110         |
| O                | 0.98        | 0.99        | 0.99        | 141         |
| P                | 0.98        | 0.95        | 0.96        | 92          |
| Q                | 0.92        | 0.98        | 0.95        | 127         |
| R                | 0.99        | 0.93        | 0.96        | 102         |
| S                | 0.97        | 0.91        | 0.94        | 69          |
| T                | 0.95        | 1.00        | 0.98        | 98          |
| U                | 0.90        | 0.96        | 0.93        | 98          |
| V                | 0.97        | 0.87        | 0.92        | 109         |
| W                | 0.93        | 0.87        | 0.90        | 101         |
| X                | 0.88        | 0.95        | 0.91        | 77          |
| Y                | 0.93        | 0.91        | 0.92        | 94          |
| Z                | 0.89        | 0.91        | 0.90        | 172         |
| <b>Macro avg</b> | <b>0.95</b> | <b>0.95</b> | <b>0.95</b> | <b>2638</b> |
| <b>Weighted</b>  | <b>0.95</b> | <b>0.95</b> | <b>0.95</b> | <b>2638</b> |

|                 |      |      |
|-----------------|------|------|
| <b>Accuracy</b> | 0.95 | 2638 |
|-----------------|------|------|

### 5.3.3 Letters Classification - Long exposure

The results obtained with the Decision Tree algorithm, for the three methods, for classifying Letters (Long exposure) are shown in table 5.6. It is possible to notice that there is not a relevant variation in the classification time but that there is a notable degradation in accuracy when using only the gyroscope data in the classification.

The best classification results were obtained in method 1, using a separation of 30% for training and 70% for classification (test size = 0.3) and accelerometer and gyroscope data, with a accuracy of 97,7%, as detailed in table 5.7.

Table 5.6: Result of the Decision Tree Classification Model For Letters Long exposure

| Method | Training/Classification Ratio | Considered Data           | Other parameters                   | Accuracy         | Classification Time |
|--------|-------------------------------|---------------------------|------------------------------------|------------------|---------------------|
| 1      | 30% / 70%                     | accelerometer + gyroscope | random state = 1<br>shuffle = True | 97,7%<br>(0.977) | <0.1s               |
| 2      | 40% / 60%                     | accelerometer + gyroscope | random state = 1<br>shuffle = True | 96,1%<br>(0.961) | <0.1s               |
| 3      | 30% / 70%                     | gyroscope                 | random state = 1<br>shuffle = True | 27,3%<br>(0.273) | <0.1s               |

Table 5.7: Best Result Decision Tree (30%/70%) For Letter Long Exposure Classification Using Accelerometer And Gyroscope Data

| Target       | Precision   | Recall      | F1-Score    | Support     |
|--------------|-------------|-------------|-------------|-------------|
| A            | 1.00        | 0.99        | 0.99        | 72          |
| B            | 1.00        | 1.00        | 1.00        | 78          |
| C            | 1.00        | 1.00        | 1.00        | 58          |
| D            | 1.00        | 0.92        | 0.96        | 66          |
| E            | 0.97        | 1.00        | 0.99        | 68          |
| F            | 1.00        | 0.98        | 0.99        | 51          |
| G            | 0.98        | 1.00        | 0.99        | 52          |
| H            | 1.00        | 0.98        | 0.99        | 47          |
| I            | 0.98        | 1.00        | 0.99        | 48          |
| J            | 0.97        | 0.98        | 0.97        | 59          |
| K            | 0.98        | 0.95        | 0.96        | 55          |
| L            | 0.97        | 1.00        | 0.98        | 60          |
| M            | 0.98        | 0.97        | 0.97        | 58          |
| N            | 0.96        | 0.92        | 0.94        | 53          |
| O            | 1.00        | 1.00        | 1.00        | 80          |
| P            | 0.98        | 1.00        | 0.99        | 60          |
| Q            | 0.94        | 0.98        | 0.96        | 83          |
| R            | 0.98        | 0.98        | 0.98        | 64          |
| S            | 0.96        | 0.96        | 0.96        | 53          |
| T            | 1.00        | 1.00        | 1.00        | 54          |
| U            | 0.96        | 0.98        | 0.97        | 52          |
| V            | 0.95        | 0.98        | 0.96        | 56          |
| W            | 0.90        | 0.93        | 0.91        | 57          |
| X            | 0.96        | 1.00        | 0.98        | 50          |
| Y            | 1.00        | 0.97        | 0.98        | 59          |
| Z            | 0.97        | 0.94        | 0.96        | 102         |
| Macro Avg    | <b>0.98</b> | <b>0.98</b> | <b>0.98</b> | <b>1595</b> |
| Weighted Avg | <b>0.98</b> | <b>0.98</b> | <b>0.98</b> | <b>1595</b> |

|          |      |      |
|----------|------|------|
| Accuracy | 0.98 | 1595 |
|----------|------|------|

## 5.4 Random Forest

The Random Forest algorithm is an ensemble learning method that combines multiple decision trees to make predictions. It creates a forest of decision trees, where each tree is built independently using a subset of the training data and a random selection of features. The algorithm makes predictions by aggregating the predictions of individual trees through voting for classification. The Random Forest algorithm builds an ensemble of decision trees, whereas the Decision Tree algorithm constructs a single tree. The ensemble aspect of Random Forest should enhance predictive performance and reduce overfitting.

The execution of the Random Forest algorithm on the Objects and Actions data set, Letters data set and Letters (Long exposure) data set was carried out in three different methods, with the aim of achieving the best possible classification:

- The first classification on Objects and Actions data set was performed using a separation of 30% for training and 70% for classification (test size = 0.3) and accelerometer and gyroscope data were considered.
- The second classification on Objects and Actions data set was performed using a separation of 40% for training and 60% for classification (test size = 0.4) and accelerometer and gyroscope data were considered.
- The third classification on Objects and Actions data set was performed using a separation of 30% for training and 70% for sorting (test size = 0.3). However, for this test only gyroscope data were considered.

The default parameters for Random Forest were maintained for all classifications.

#### 5.4.1 Objects and Actions Classification

The results obtained with the Random Forest algorithm, for the three methods, for classifying Objects and Actions are shown in table 5.8. It is possible to notice that there is not a relevant variation in the classification time but that there is degradation in accuracy when using only the gyroscope data in the classification.

Table 5.8: Result of the Random Forest Classification Model For Objects and Actions

| Method | Training/Classification Ratio | Considered Data           | Other parameters | Accuracy         | Classification Time |
|--------|-------------------------------|---------------------------|------------------|------------------|---------------------|
| 1      | 30% / 70%                     | accelerometer + gyroscope | default          | 82,4%<br>(0.824) | 0.6s                |
| 2      | 40% / 60%                     | accelerometer + gyroscope | default          | 79,6%<br>(0.796) | 0.6s                |
| 3      | 30% / 70%                     | gyroscope                 | default          | 34,1%<br>(0.341) | 0.5s                |

The best classification results were obtained in method 1, using a separation of 30% for training and 70% for classification (test size = 0.3) and accelerometer and gyroscope data, with a accuracy of 82,4%, as detailed in table 5.9.

#### 5.4.2 Letters Classification

The results obtained with the Random Forest algorithm, for the three methods, for classifying Letters are shown in table 5.10. It is possible to notice that remarkably shorter classification times were achieved when using only gyroscope data. The accuracy degradation when using only gyroscope data is also not high as observed in the Decision Tree algorithm. It is also possible to notice that, only in this single test, 40% for training and 60% for classification generated greater accuracy.

Table 5.9: Best Result Random Forest (30%/70%) For Objects and Actions Classification Using Accelerometer And Gyroscope Data

| Target             | Precision   | Recall      | F1-Score    | Support    |
|--------------------|-------------|-------------|-------------|------------|
| WALK               | 0.79        | 0.81        | 0.80        | 37         |
| BOAT               | 0.84        | 0.90        | 0.87        | 41         |
| CHAIR              | 0.84        | 0.90        | 0.87        | 29         |
| BED                | 0.81        | 0.81        | 0.81        | 32         |
| CAR                | 0.93        | 0.96        | 0.94        | 26         |
| WITH               | 0.88        | 0.68        | 0.76        | 31         |
| KEY                | 0.93        | 0.88        | 0.90        | 16         |
| PURCHASE           | 0.89        | 0.80        | 0.84        | 30         |
| COMPUTER           | 1.00        | 0.65        | 0.79        | 26         |
| IN                 | 0.86        | 0.90        | 0.88        | 20         |
| WRITE              | 0.94        | 0.84        | 0.89        | 38         |
| IGNORE             | 0.77        | 0.97        | 0.86        | 34         |
| GO                 | 0.86        | 0.88        | 0.87        | 41         |
| BOOK               | 0.75        | 0.87        | 0.81        | 31         |
| STOP               | 0.87        | 0.80        | 0.83        | 25         |
| TAKE               | 1.00        | 0.60        | 0.75        | 25         |
| ASK                | 0.61        | 0.81        | 0.69        | 31         |
| JUMP               | 0.74        | 0.89        | 0.81        | 28         |
| CELL PHONE         | 0.76        | 0.87        | 0.81        | 30         |
| TV                 | 0.79        | 0.61        | 0.69        | 31         |
| <b>Macro Avg</b>   | <b>0.84</b> | <b>0.82</b> | <b>0.82</b> | <b>602</b> |
| <b>Weighted Av</b> | <b>0.84</b> | <b>0.82</b> | <b>0.82</b> | <b>602</b> |

|                 |      |     |
|-----------------|------|-----|
| <b>Accuracy</b> | 0.82 | 602 |
|-----------------|------|-----|

Table 5.10: Result of the Random Forest Classification Model For Letters

| Method | Training/Classification Ratio | Considered Data           | Other parameters | Accuracy         | Classification Time |
|--------|-------------------------------|---------------------------|------------------|------------------|---------------------|
| 1      | 30% / 70%                     | accelerometer + gyroscope | default          | 97,2%<br>(0.972) | 1.9s                |
| 2      | 40% / 60%                     | accelerometer + gyroscope | default          | 97,7%<br>(0.977) | 1.8s                |
| 3      | 30% / 70%                     | gyroscope                 | default          | 55,3%<br>(0.553) | 0.1s                |

The best classification results were obtained in method 2, using a separation of 40% for training and 60% for classification (test size = 0.3) and accelerometer and gyroscope data, with a accuracy of 97,7%, as detailed in table 5.11.

### 5.4.3 Letters Classification - Long exposure

The results obtained with the Random Forest algorithm, for the three methods, for classifying Letters (Long exposure) are shown in table 5.12. It is possible to notice that remarkably shorter classification times were achieved when using only gyroscope data. The accuracy degradation when using only gyroscope data is also not high as observed in the Decision Tree algorithm.

The best classification results were obtained in method 1, using a separation of 30% for training and 70% for classification (test size = 0.3) and accelerometer and gyroscope data, with a accuracy of 99,2%, as detailed in table 5.13.

Table 5.11: Best Result Random Forest (30%/70%) For Letters Classification Using Accelerometer And Gyroscope Data

| Target              | Precision   | Recall      | F1-Score    | Support     |
|---------------------|-------------|-------------|-------------|-------------|
| A                   | 0.97        | 0.97        | 0.97        | 148         |
| B                   | 0.98        | 0.96        | 0.97        | 135         |
| C                   | 0.99        | 1.00        | 1.00        | 102         |
| D                   | 0.95        | 0.94        | 0.95        | 102         |
| E                   | 0.96        | 0.96        | 0.96        | 90          |
| F                   | 0.97        | 0.96        | 0.97        | 72          |
| G                   | 0.98        | 0.95        | 0.97        | 59          |
| H                   | 0.97        | 1.00        | 0.98        | 59          |
| I                   | 0.99        | 0.97        | 0.98        | 69          |
| J                   | 1.00        | 0.98        | 0.99        | 109         |
| K                   | 0.97        | 0.99        | 0.98        | 85          |
| L                   | 0.98        | 0.99        | 0.99        | 120         |
| M                   | 1.00        | 0.99        | 0.99        | 95          |
| N                   | 0.99        | 0.96        | 0.98        | 112         |
| O                   | 0.99        | 0.98        | 0.98        | 139         |
| P                   | 0.99        | 0.99        | 0.99        | 92          |
| Q                   | 0.98        | 0.94        | 0.96        | 140         |
| R                   | 1.00        | 0.99        | 1.00        | 105         |
| S                   | 0.99        | 0.96        | 0.97        | 80          |
| T                   | 1.00        | 0.98        | 0.99        | 107         |
| U                   | 0.95        | 0.96        | 0.95        | 91          |
| V                   | 0.82        | 0.96        | 0.89        | 97          |
| W                   | 0.96        | 1.00        | 0.98        | 100         |
| X                   | 0.98        | 1.00        | 0.99        | 50          |
| Y                   | 0.93        | 0.98        | 0.96        | 57          |
| Z                   | 0.97        | 1.00        | 0.99        | 105         |
| <b>Macro Avg</b>    | <b>0.97</b> | <b>0.97</b> | <b>0.97</b> | <b>2638</b> |
| <b>Weighted Avg</b> | <b>0.97</b> | <b>0.97</b> | <b>0.97</b> | <b>2638</b> |

|                 |      |      |
|-----------------|------|------|
| <b>Accuracy</b> | 0.97 | 2638 |
|-----------------|------|------|

Table 5.12: Result of the Random Forest Classification Model For Letters Long exposure

| Method | Training/Classification Ratio | Considered Data           | Other parameters | Accuracy         | Classification Time |
|--------|-------------------------------|---------------------------|------------------|------------------|---------------------|
| 1      | 30% / 70%                     | accelerometer + gyroscope | default          | 99,2%<br>(0.992) | 1.3s                |
| 2      | 40% / 60%                     | accelerometer + gyroscope | default          | 99,0%<br>(0.990) | 1.1s                |
| 3      | 30% / 70%                     | gyroscope                 | default          | 37,5%<br>(0.375) | 0.1s                |

## 5.5 Summary

As we can see in this chapter, the finalization process of the data sets used and the application and validation of the classification algorithms were detailed.

In the preparation of the data sets, we explained how the data were generated, sanitized, and labeled. We detail the difficulties encountered during the data capture process and the solutions adopted to generate a quality data set that could feed the classification algorithms in an attempt to generate the highest volume of quality results possible.

In the application of the algorithms, we gave a brief explanation of the differences between

Table 5.13: Best Result Random Forest (30%/70%) For Letters Long Exposure Classification Using Accelerometer And Gyroscope Data

| Target              | Precision   | Recall      | F1-Score    | Support     |
|---------------------|-------------|-------------|-------------|-------------|
| A                   | 1.00        | 1.00        | 1.00        | 71          |
| B                   | 1.00        | 1.00        | 1.00        | 70          |
| C                   | 1.00        | 1.00        | 1.00        | 60          |
| D                   | 1.00        | 1.00        | 1.00        | 57          |
| E                   | 1.00        | 1.00        | 1.00        | 68          |
| F                   | 1.00        | 1.00        | 1.00        | 47          |
| G                   | 1.00        | 1.00        | 1.00        | 50          |
| H                   | 1.00        | 1.00        | 1.00        | 46          |
| I                   | 1.00        | 1.00        | 1.00        | 50          |
| J                   | 1.00        | 1.00        | 1.00        | 66          |
| K                   | 0.98        | 1.00        | 0.99        | 55          |
| L                   | 1.00        | 1.00        | 1.00        | 67          |
| M                   | 1.00        | 0.98        | 0.99        | 48          |
| N                   | 0.98        | 1.00        | 0.99        | 51          |
| O                   | 1.00        | 1.00        | 1.00        | 89          |
| P                   | 0.98        | 1.00        | 0.99        | 55          |
| Q                   | 1.00        | 0.97        | 0.99        | 72          |
| R                   | 1.00        | 0.97        | 0.98        | 66          |
| S                   | 1.00        | 1.00        | 1.00        | 39          |
| T                   | 1.00        | 1.00        | 1.00        | 76          |
| U                   | 1.00        | 0.98        | 0.99        | 57          |
| V                   | 1.00        | 0.98        | 0.99        | 56          |
| W                   | 0.98        | 0.94        | 0.96        | 67          |
| X                   | 0.98        | 1.00        | 0.99        | 50          |
| Y                   | 0.93        | 0.98        | 0.96        | 57          |
| Z                   | 0.97        | 1.00        | 0.99        | 105         |
| <b>Macro Avg</b>    | <b>0.99</b> | <b>0.99</b> | <b>0.99</b> | <b>1595</b> |
| <b>Weighted Avg</b> | <b>0.99</b> | <b>0.99</b> | <b>0.99</b> | <b>1595</b> |

|                 |      |      |
|-----------------|------|------|
| <b>Accuracy</b> | 0.99 | 1595 |
|-----------------|------|------|

Decision Trees and Random Forest algorithms and performed the application of both classification algorithms in the data sets. Metrics such as accuracy and classification time were recorded, as were multiple attempts with small differences in parameters and data exposed to the algorithms. Considering that no other study was identified that used the same type of data set as this one, multiple classifications with small differences generate the possibility of comparing the results.

Considering only the best results obtained with the Decision Tree and Random Forest classification models, we noticed variations in classification accuracy and classification time.

For the classification of data representing Objects and Actions, the Random Forest model managed to achieve better accuracy with 82%, but the classification time was 0.5s longer than the Decision Tree model, as can be seen in detail in Table 5.14.

For the classification of data representing Letters, the Random Forest model achieved better accuracy with 97%, but the classification time was 1.8 seconds longer than the Decision Tree model, as can be seen in detail in Table 5.15.

For the classification of data representing letters (considering only the use of data captured through long exposure of the movement to the data capture hardware), the Random Forest model

Table 5.14: Objects and Actions Classification - Decision Tree Vs Random Forest

| 30%-70%                   |          |           |        |          |                     |
|---------------------------|----------|-----------|--------|----------|---------------------|
| Accelerometer + Gyroscope |          |           |        |          |                     |
| Model                     | Accuracy | Precision | Recall | F1-Score | Classification Time |
| Decision Tree             | 0.72     | 0.73      | 0.72   | 0.72     | <0.1s               |
| Random Forest             | 0.82     | 0.84      | 0.82   | 0.82     | 0.6s                |

Table 5.15: Letters Classification - Decision Tree Vs Random Forest

| 30%-70%                   |          |           |        |          |                     |
|---------------------------|----------|-----------|--------|----------|---------------------|
| Accelerometer + Gyroscope |          |           |        |          |                     |
| Model                     | Accuracy | Precision | Recall | F1-Score | Classification Time |
| Decision Tree             | 0.95     | 0.95      | 0.95   | 0.95     | 0.1s                |
| Random Forest             | 0.97     | 0.97      | 0.97   | 0.97     | 1.9s                |

managed to achieve better accuracy with 99% and the classification time was 1.2 seconds longer than the Decision Tree model, as can be seen in detail in Table 5.16.

Table 5.16: Letters (Long Exposure) Classification - Decision Tree Vs Random Forest

| 30%-70%                   |          |           |        |          |                     |
|---------------------------|----------|-----------|--------|----------|---------------------|
| Accelerometer + Gyroscope |          |           |        |          |                     |
| Model                     | Accuracy | Precision | Recall | F1-Score | Classification Time |
| Decision Tree             | 0.98     | 0.98      | 0.98   | 0.98     | <0.1s               |
| Random Forest             | 0.99     | 0.99      | 0.99   | 0.99     | 1.3s                |

The Letters data set, which is the largest data set under test, was classified in its entirety by the Random Forest algorithm in 1.9 seconds, we have a classification time of 0.22 milliseconds for each data set record. Considering that the hardware is configured to generate around 8 records per second for each hand, the Random Forest algorithm, even taking a longer classification time, is fully efficient for use in real-time classification (Paudyal et al., 2016) (Fang et al., 2004a).

Considering that most other studies in this field, mainly those mentioned in the state of the art, reach accuracies between 80% and 98% (Kudrinko et al., 2021b), we can consider the results obtained for the classification of Letters very positive. The results for signal classification still have room for improvement. However, it is important to mention that there is no consistency between these cited studies and this one, when considering the base data and the applied classification algorithms, that are essentially different. Therefore, caution should be used when directly comparing the results.

## Chapter 6

# Real-time Recognition Approach

In this chapter, we detail the construction of a real-time Portuguese Sign Language recognition system using the hardware detailed in Chapter 4 and the Random Forest algorithm detailed in Chapter 5 which generated better results in the classification of data sets. The construction of a real-time recognition system for Portuguese Sign Language does not interfere with the final conclusion of this study, and it is not intended to quantify or qualify the results obtained with the algorithms but only to verify the viability of such a system. The main objective is to contribute to greater knowledge in the practical application of this study and help identify points of attention for future studies in this area.

### 6.1 System Architecture

The base architecture for the real-time recognition system is the same as used in the acquisition of test data, with the difference that the data sets already captured and treated in the initial phases of the study are used to train the algorithm at the beginning of the process, allowing the trained algorithm to classify the data generated in real time by the sensors.

For the real-time recognition tests scenario, only one of the two capture hardware devices was used. We chose to use the hardware for the right hand, which has a more complete and robust training data set and also has data for letter classification.

The final design of the architecture of the real-time recognition system can be seen in Figure 6.1.

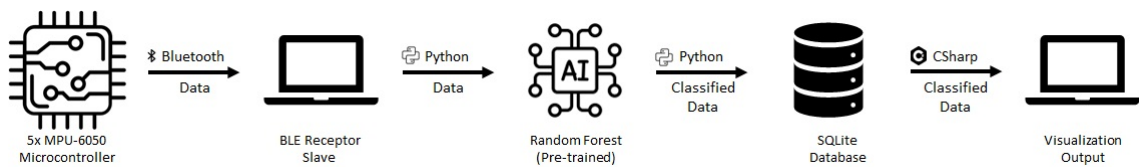


Figure 6.1: Architecture of the Real Time Classification System

Python and CSharp applications work independently of each other. The Python application is responsible for establishing a connection with the microcontroller via Bluetooth, receiving data

from the sensors, submitting this data to the classification algorithm, and recording the results in a database. The CSharp application accesses that database, reads the data, and shows on a interface the generated classification for the last set of data every second.

The Python application works with multiple threads to be able to receive data from hands at the same time and classify it without generating any bottlenecks in the system.

## 6.2 Tests

The Portuguese Sign Language real-time recognition system built worked as expected, managing to correctly classify and display several letters of the alphabet. The classification of words and actions was hampered because we were using only one hand, but even so, it was possible to identify some.

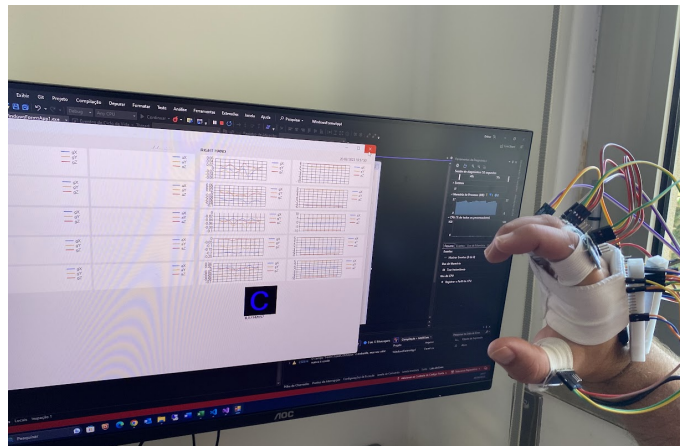


Figure 6.2: Real Time Recognition System Identifying Letters "C"

However, in the first tests, a high level of false positives was also observed since the algorithm tried to classify in the best way all the inputs generated by the sensors. To decrease the amount of false positives visualized, we used the "predict\_proba" method from the Scikit-Learn library to create a filter on which classifications should be displayed. The method "predict\_proba" is used in classification models to obtain the estimated belonging probabilities of each class for a given input instance. It returns a matrix where each row represents an instance and each column represents a class. The values in the matrix indicate the estimated probabilities that the instance belongs to the respective class. In simpler terms, "predict\_proba" is useful for understanding the model's confidence in its class predictions. This is particularly useful when you want to understand not just the predicted class but also how accurate the model is in its prediction.

Based on this method, the system was configured to only display classifications with a high confidence probability equal to or greater than 70%.

As the use of a real-time recognition system was not the focus of this study at the time, no statistical metrics were carried out on its level of recognition accuracy. However, with the recognition system turned on, all letters of the alphabet were tested, and of the 26 letters of the alphabet,

the system was able to recognize 21 letters with a high degree of accuracy.

It was observed that the letter F was mistakenly recognized as the letter J, and the letter H was mistakenly recognized as the letter C. The letters K, L, and O presented high difficulty for recognition by the system.

Another interesting point of attention that was noticed during the tests was the difficulty in classifying hand shapes that are so similar that the sensors, because of the way they are distributed in the hand, cannot generate sufficiently different data for an accurate classification. This was mostly noticed between the letters "N" and "M". The difference between the two letters is just the curvature of the tip of the ring finger. As the sensor is at the base of the ring finger, there is no significant change in the generated data, which impacts the classification. In these cases, a hardware change would have to be made with the addition of an extra sensor on the tip of the finger, or the gesture of one of the letters would have to be done in a different way.

Some words were also tested, even though they were two-handed signs, and during the test only one hand was being considered. The words "bed" and "jump" were tested and satisfactorily recognized by the system.

## **6.3 Summary**

A real-time Portuguese Sign Language recognition system using low-cost sensors proved to be efficient enough in its first version. It was able to correctly recognize and display letters and words previously trained by the classification algorithm.

This system also helped to identify and understand the limitations of the hardware and the data generated. New studies and evolutions throughout the system must be carried out to increase recognition reliability.

## Chapter 7

# Conclusions and Future Work

The purpose of this chapter is to compile all findings and results obtained during this study, not limited to the final objective itself but to all stages, from hardware assembly to the final tests performed. The possibilities for future studies involving the results obtained and the study area will also be discussed.

### 7.1 Conclusions

We can segment the conclusions obtained throughout the study between the different steps performed: Hardware Assembly, Data Acquisition and Analysis and Recognition Algorithms.

- **Hardware Assembly:** the process of acquiring and assembling the hardware proved to be quite challenging, both because of my lack of initial knowledge related to working with hardware and because of the excess of options on the market and the variety of components, connections, prices, manufacturers, and data type generated.

The type of sensor chosen MPU-6050 has all the characteristics that were being looked for: the *cost* of the sensor for sale in the direct market is below €1.00, and it can even be acquired for smaller values in larger purchases; the *size* of the sensor (mounted on the GY-521 board) is perfect for use in prototyping and can be used in this study without impairing the movements performed by sign language users; the existing *connection* in the model is the IC2 standard, perfect for prototyping and connecting with Arduino components; and, mainly, the *data generated* by this sensor proved to be perfect for this type of study. The fact that the sensor is able to generate accelerometer and gyroscope data, as detailed in the section on algorithm analysis, was fundamental for the algorithms to be able to generate high accuracy results in signal classification.

We have not detected any weaknesses in the use of MPU-6050 sensors. What can be considered a possible improvement would be that, if we used only the MPU-6050 sensor on a customized board, we would be able to further reduce the component size, which improves the overall usability of the equipment.

The use of an Arduino microcontroller also allows easy integration between the components. There are thousands of code examples online to build on, and the forums for Arduino enthusiasts are quite active. Using Bluetooth communication to send the generated data to computers was also efficient, as it allowed greater mobility for users when performing sign language. They were not tethered to a computer and could move freely during data capture.

The only point that showed problems in the construction of the hardware turned out to be the connections between the components, which were made with unsoldered jumper cables. Sometimes these cables drop or fail during sudden movements of users, which could generate some partial loss of data or total loss of data generated during a time interval. However, the cables were kept that way to give us more freedom when assembling and disassembling the hardware during tests and validations. For a solution with greater construction durability and quality assurance, the cables must be soldered directly between the boards.

- **Data Acquisition:** the data acquisition process and the creation of a specific data dictionary for the study were essential to speeding up these steps. Considering a current dictionary of Portuguese Sign Language with more than 25,000 entries, it would be impossible to carry out a quick study with the complete dictionary, especially considering that there are no public data sets already compiled with the necessary data used in this study.

The choice to collect study data with fluent and non-fluent users of Portuguese Sign Language also allowed for interesting data analysis. It was possible to identify a higher accuracy rate in the data generated by non-fluent speakers simply because the sign language movements were performed more calmly and for a longer period of time. This shows that the collection of data for the same sign by different people would be important to generate a robust training data set for the recognition of Portuguese Sign Language.

This line of studies also attracted the attention of the "Associação dos Surdos do Porto", which during interviews highlighted the current difficulties in communication between sign language users and those who use a spoken language. This demonstrates that there is a market for any technology that can increase the inclusion of this group in society with non-sign language speakers.

- **Analysis and Recognition Algorithms:** the step of applying the classification algorithms to the collected data proved to be very positive when compared to the initial objectives of this study. The classification algorithms managed to achieve very relevant accuracy results.

We verified that both the Decision Tree and Random Forest algorithms are able to reach accuracies greater than 70% in the classification of Words and Letters in data referring to sign language, with special emphasis on the Random Forest algorithm, which achieved very interesting accuracies.

We were able to identify that the classification of letters of the alphabet reached higher accuracy than the classification of words. We believe this is due to the fact that the letters of the alphabet are mostly static hand shapes, while words are fluid movements. That is,

the algorithm can better classify static shapes. Another point to be considered was that for words, we only used data from the right hand since we had problems with data from the left hand for some words and discarded the entire data set for this study. For words that use both hands, an algorithm with multiple classifications or double validation would be necessary.

We verified that both gyroscope and accelerometer data are important for good classification accuracy. When we discard the accelerometer data and focus only on the gyroscope data, the overall quality of accuracy achieved is much lower.

We identified that the Decision Tree algorithm is around 5x faster than the Random Forest algorithm, but that the Random Forest achieves superior classification accuracy.

We also identified that classification accuracy is directly linked to the amount of data collected. The more quality data collected, the better the ability of the correct classification system. Based on this, but without a proper test to verify this statement, we can argue that if we increase the volume of data generated per second by the sensors, we could achieve greater accuracy by having more base data for analysis. This could also be relevant considering that people make sign language signs at different speeds.

Answering the question of this study, we can say with a high degree of confidence that low-cost sensors are capable of generating data with sufficient quality to feed classification algorithms and build a Portuguese Sign Language recognition system.

## 7.2 Future Work

The results of this study can be used as a basis for other projects and studies in the field of sign language recognition, aiming not only to achieve better classification accuracy results but also to allow the creation of a system for real-world use of portuguese sign language recognition.

The dictionary used in the study (20 words/actions and 26 letters of the alphabet) can be expanded to include new words. This would allow us to feed the algorithms with more data and verify if the classification quality would remain high with the use of more data and if significant impacts on the performance of the algorithms would happen. It would also be important to increase the number of data collections by different users, as each person performs sign language movements slightly differently.

A more robust construction of the hardware could be used, using soldered and not just wired cables, reducing individual sensor failures in data generation during use by users. It would also be useful to implement in the data sanitization step a step to normalize data incorrectly generated by the sensors. That is, when a sensor fails to generate data and generates zero data, these zero data could be replaced by an average of the previous and subsequent data, and we could see if this would improve the overall accuracy of the classification.

The best approach to integrating the recognition of both hands at the same time must be analyzed and identified in order to prioritize word recognition, taking advantage of all the data generated and avoiding discordant recognition between the hands.

The application of new classification algorithms (in addition to Decision Tree and Random Forest) would also be important to achieve a good correlation between classification quality and classification performance.

The real-time classification system can be improved and its accuracy statistically determined so that it is possible to verify the applicability of the solutions in real-world scenarios.

# References

- Vasco Alves, Jorge Ribeiro, Pedro Faria, and Luís Romero. Neural machine translation approach in automatic translations between portuguese language and portuguese sign language glosses. In *2022 17th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–7, 2022. doi: 10.23919/CISTI54924.2022.9820212.
- Sunita Kumari Salim Shamsheer Jeevanandam Joteeswaran Anu Saini, Amit Ramesh Khaparde and Seifedine Kadry. An investigation of machine learning techniques in speech emotion recognition. *The Indonesian Journal of Electrical Engineering and Computer Science (IJECS)*, 29 (2):875–882, 2023. doi: 10.1109/TSMCA.2004.824852.
- ASP. Associação de surdos do porto, 2023. Available at <https://asurdosporto.org.pt/>.
- CENSOS. Portuguese national institute of statistics - population and housing censuses, 2023. Available at <https://ine.pt>.
- U. Bellugi D. P. Corina and J. Reilly. Neuropsychological studies of linguistic and affective facial expressions in deaf signers. *Lang. Speech*, 42, 1999. URL <https://pubmed.ncbi.nlm.nih.gov/10767992/>.
- Giulia Zanon de Castro, Rúbia Reis Guerra, and Frederico Gadelha Guimarães. Automatic translation of sign language with multi-stream 3d cnn and generation of artificial depth maps. *Expert Systems with Applications*, 215:119394, 2023. ISSN 0957-4174. doi: <https://doi.org/10.1016/j.eswa.2022.119394>. URL <https://www.sciencedirect.com/science/article/pii/S0957417422024125>.
- Associação Portuguesa de Surdos, 2023. Available at <https://apsurdos.org.pt/>.
- Circuit Digest. Circuit digest, 2023. Available at <https://circuitdigest.com/microcontroller-projects/interfacing-mpu6050-module-with-arduino>.
- Ethnologue, 2023. Available at <https://www.ethnologue.com/subgroups/sign-language>.
- Gaolin Fang, Wen Gao, and Debin Zhao. Large vocabulary sign language recognition based on fuzzy decision trees. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 34(3):305–314, 2004a. doi: 10.1109/TSMCA.2004.824852.
- Gaolin Fang, Wen Gao, and Debin Zhao. Large vocabulary sign language recognition based on fuzzy decision trees. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 34(3):305–314, 2004b. doi: 10.1109/TSMCA.2004.824852.

- Matilde Gonçalves, Luísa Coheur, Hugo Nicolau, and Ana Mineiro. Pe2lgp: tradutor de português europeu para língua gestual portuguesa em glosas. *Linguamatica*, 13(1):3–21, June 2021. ISSN 1647-0818. doi: 10.21814/lm.13.1.338. Publisher Copyright: © 2021 Universidade do Minho. All rights reserved.
- Yutong Gu, Chao Zheng, Masahiro Todoh, and Fusheng Zha. American sign language translation using wearable inertial and electromyography sensors for tracking hand movements and facial expressions. *Frontiers in Neuroscience*, 16, 2022. ISSN 1662-453X. doi: 10.3389/fnins.2022.962141. URL <https://www.frontiersin.org/articles/10.3389/fnins.2022.962141>.
- Kianoush Haratiannejadi, Neshat Elhami Fard, and Rastko Selmic. Smart glove and hand gesture-based control interface for multi-rotor aerial vehicles. pages 1956–1962, 10 2019. doi: 10.1109/SMC.2019.8914464.
- Deaf History. Deaf history, 2023. Available at <https://deafhistory.eu/index.php/component/zoo/item/paer-arón-borg>.
- J. Hong and X. Tan. Calibrating a vpl dataglove for teleoperating the utah/mit hand. In *Proceedings, 1989 International Conference on Robotics and Automation*, pages 1752–1757 vol.3, 1989. doi: 10.1109/ROBOT.1989.100228.
- CyberGlove Systems Inc. Cyberglove systems product, 2023. Available at <http://www.cyberglovesystems.com>.
- Infopedia. Portuguese sign language, 2023a. Available at <https://www.infopedia.pt/dicionarios/lingua-gestual>.
- Infopedia. Portuguese sign language dictionary, 2023b. Available at <https://www.infopedia.pt/dicionarios/lingua-gestual/>.
- TDK InvenSense. Tdk invensense, 2023. Available at <https://invensense.tdk.com/products/motion-tracking/6-axis/mpu-6050/>.
- Karly Kudrinko, Emile Flavin, Xiaodan Zhu, and Qingguo Li. Wearable sensor-based sign language recognition: A comprehensive review. *IEEE Reviews in Biomedical Engineering*, 14: 82–97, 2021a. doi: 10.1109/RBME.2020.3019769.
- Karly Kudrinko, Emile Flavin, Xiaodan Zhu, and Qingguo Li. Wearable sensor-based sign language recognition: A comprehensive review. *IEEE Reviews in Biomedical Engineering*, 14: 82–97, 2021b. doi: 10.1109/RBME.2020.3019769.
- Peter B. Shull Ling Li, Shuo Jiang and Guoying Gu. Skingest: artificial skin for gesture recognition via filmy stretchable strain sensors\*. *Advanced Robotics*, 32(21):1112–1121, 2018. doi: 10.1080/01691864.2018.1490666. URL <https://doi.org/10.1080/01691864.2018.1490666>.
- Sushmita Mitra and Tinku Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(3):311–324, 2007. doi: 10.1109/TSMCC.2007.893280.
- WFD News. Magazine of the world federation of the deaf. In *WFD News - Magazine of the World Federation of the Deaf*, volume 16, page 7, 2003.

- Prajwal Paudyal, Ayan Banerjee, and Sandeep K.S. Gupta. Sceptre: A pervasive, non-invasive, and programmable gesture recognition technology. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, IUI '16, page 282–293, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450341370. doi: 10.1145/2856767.2856794. URL <https://doi.org/10.1145/2856767.2856794>.
- M.I. Prata. *Maos que falam*. Laboratorio de Fonetica, Faculdade de Letras, Universidade de Lisboa, 1980. URL <https://books.google.pt/books?id=wBYyzwEACAAJ>.
- Hashir Zahid Sheikh Soly Mathew Biju, Obada Al Khatib. A review of factors that impact the design of a glove based wearable devices. *International Journal of Artificial Intelligence (IAES)*, 12(2):522–531, 2023. doi: 10.11591/ijai.v12.i2.pp522-531.
- Chunli Wang, Wen Gao, and Shiguang Shan. An approach based on phonemes to large vocabulary chinese sign language recognition. In *Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition*, pages 411–416, 2002. doi: 10.1109/AFGR.2002.1004188.
- WHO. Deafness and hearing loss, 2021. Available at <https://www.who.int/publications/i/item/9789240020481/>.
- Jian Wu, Zhongjun Tian, Lu Sun, Leonardo Estevez, and Roozbeh Jafari. Real-time american sign language recognition using wrist-worn motion and surface emg sensors. In *2015 IEEE 12th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, pages 1–6, 2015. doi: 10.1109/BSN.2015.7299393.
- Andre Nogueira Xavier. *Phonetic-phonological description of Brazilian Sign Language signs (LIBRAS)*. 2006.
- Xu Zhang, Xiang Chen, Yun Li, Vuokko Lantz, Kongqiao Wang, and Jihai Yang. A framework for hand gesture recognition based on accelerometer and emg sensors. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 41(6):1064–1076, 2011. doi: 10.1109/TSMCA.2011.2116004.
- Tianming Zhao, Jian Liu, Yan Wang, Hongbo Liu, and Yingying Chen. Ppg-based finger-level gesture recognition leveraging wearables. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 1457–1465, 2018. doi: 10.1109/INFOCOM.2018.8486006.
- Tianming Zhao, Jian Liu, Yan Wang, Hongbo Liu, and Yingying Chen. Towards low-cost sign language gesture recognition leveraging wearables. *IEEE Transactions on Mobile Computing*, 20(4):1685–1701, 2021. doi: 10.1109/TMC.2019.2962760.
- Ye Ziyang, Guo Hong, and Ji Wenhao. Design and implementation of a mems-based attitude angle measuring system for moving objects. In *2021 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS)*, pages 145–149, 2021. doi: 10.1109/ICPICS52425.2021.9524095.

## Appendix A

# C++ Arduino Microcontroller Code Example

```
#include <Arduino.h>
#include <BluetoothSerial.h>
#include <Wire.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>

#define TCAADDR 0x70

#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run 'make menuconfig' to and enable it
#endif

BluetoothSerial SerialBT;

Adafruit_MPU6050 mpu1 = Adafruit_MPU6050();
Adafruit_MPU6050 mpu2 = Adafruit_MPU6050();
Adafruit_MPU6050 mpu3 = Adafruit_MPU6050();
Adafruit_MPU6050 mpu4 = Adafruit_MPU6050();
Adafruit_MPU6050 mpu5 = Adafruit_MPU6050();

bool sendSerial = false;
bool sendBluetooth = true;

String stringThree = "";

void displaySensorDetails(Adafruit_MPU6050 *mpu)
```

```

{
  sensor_t sensor;
  mpu->getGyroSensor();
  Serial.println("-----");
  Serial.print ("Sensor: "); Serial.println(sensor.name);
  Serial.print ("Driver Ver: "); Serial.println(sensor.version);
  Serial.print ("Unique ID: "); Serial.println(sensor.sensor_id);
  Serial.print ("Max Value: "); Serial.print(sensor.max_value);
                                   Serial.println(" uT");
  Serial.print ("Min Value: "); Serial.print(sensor.min_value);
                                   Serial.println(" uT");
  Serial.print ("Resolution: "); Serial.print(sensor.resolution);
                                   Serial.println(" uT");

  Serial.println("-----");
  Serial.println("");
  delay(500);
}

void configureSensor(Adafruit_MPU6050 *mpu)
{
  Serial.print(" Configuring ... ");
  mpu->setAccelerometerRange(MPU6050_RANGE_8_G);
  mpu->setGyroRange(MPU6050_RANGE_500_DEG);
  mpu->setFilterBandwidth(MPU6050_BAND_5_HZ);
  delay(500);
  Serial.println("Done.");
}

void tcasselect(uint8_t i) {
  Wire.beginTransmission(TCAADDR);
  Wire.write(1 << i);
  Wire.endTransmission();
}

void getSensorData(Adafruit_MPU6050 *mpu, int i) {
  sensors_event_t a, g, temp;
  mpu->getEvent(&a, &g, &temp);

  /* Print out the values */
  if (sendSerial) {

```

```
Serial.print("rhf"); Serial.print(i); Serial.print(", ");
Serial.print("Acceleration X: ");
Serial.print(a.acceleration.x);
Serial.print(", Y: ");
Serial.print(a.acceleration.y);
Serial.print(", Z: ");
Serial.print(a.acceleration.z);
Serial.print(" m/s^2. ");

Serial.print("Rotation X: ");
Serial.print(g.gyro.x);
Serial.print(", Y: ");
Serial.print(g.gyro.y);
Serial.print(", Z: ");
Serial.print(g.gyro.z);
Serial.println(" rad/s.");
}

if (sendBluetooth) {
  if (SerialBT.connected()) {
    digitalWrite(LED_BUILTIN, HIGH);
    delay(7);

    SerialBT.print("rh|f"); SerialBT.print(i); SerialBT.print("|");
    SerialBT.print(a.acceleration.x);
    SerialBT.print("|");
    SerialBT.print(a.acceleration.y);
    SerialBT.print("|");
    SerialBT.print(a.acceleration.z);
    SerialBT.print("|");
    SerialBT.print(g.gyro.x);
    SerialBT.print("|");
    SerialBT.print(g.gyro.y);
    SerialBT.print("|");
    SerialBT.print(g.gyro.z);
    SerialBT.print("#");
    SerialBT.println();

    digitalWrite(LED_BUILTIN, LOW);
    delay(7);
```

```
    }  
  }  
}  
  
void getSensorDataX(Adafruit_MPU6050 *mpu, int i) {  
  sensors_event_t a, g, temp;  
  mpu->getEvent(&a, &g, &temp);  
  
  /* Print out the values */  
  
  if (i == 1) {  
    stringThree = "rh|";  
  }  
  else {  
    stringThree = stringThree + "|";  
  }  
  
  stringThree = stringThree + a.acceleration.x;  
  stringThree = stringThree + "|";  
  stringThree = stringThree + a.acceleration.y;  
  stringThree = stringThree + "|";  
  stringThree = stringThree + a.acceleration.z;  
  stringThree = stringThree + "|";  
  
  stringThree = stringThree + g.gyro.x;  
  stringThree = stringThree + "|";  
  stringThree = stringThree + g.gyro.y;  
  stringThree = stringThree + "|";  
  stringThree = stringThree + g.gyro.z;  
  
  if (i == 5) {  
    stringThree = stringThree + "#";  
  
    if (sendSerial) {  
      Serial.println(stringThree);  
    }  
  
    if (sendBluetooth) {  
      if (SerialBT.connected()) {  
        digitalWrite(LED_BUILTIN, HIGH);  
      }  
    }  
  }  
}
```

```
        delay(7);

        SerialBT.println(stringThree);

        digitalWrite(LED_BUILTIN, LOW);
        delay(7);
    }
}

}

void setup() {
    Serial.begin(115200);
    pinMode(LED_BUILTIN, OUTPUT);
    Wire.begin();

    Serial.println("");
    delay(5000);

    // Initialize Bluetooth
    SerialBT.begin("ESP32RHTest");
    Serial.println("Bluetooth started");

    // Initialize MPU6050 1
    tcselect(1);
    while (!mpu1.begin()) {
        Serial.println("Failed to find MPU6050 1 chip");
        delay(200);
    }
    Serial.print("MPU6050 1 Found.");
    configureSensor(&mpu1); //Configure sensor

    // Initialize MPU6050 2
    tcselect(2);
    while (!mpu2.begin()) {
        Serial.println("Failed to find MPU6050 2 chip");
        delay(200);
    }
    Serial.print("MPU6050 2 Found.");
    configureSensor(&mpu2); //Configure sensor
```

```
// Initialize MPU6050 3
tcselect(3);
while (!mpu3.begin()) {
    Serial.println("Failed to find MPU6050 3 chip");
    delay(200);
}
Serial.print("MPU6050 3 Found.");
configureSensor(&mpu3); //Configure sensor

// Initialize MPU6050 4
tcselect(4);
while (!mpu4.begin()) {
    Serial.println("Failed to find MPU6050 4 chip");
    delay(200);
}
Serial.print("MPU6050 4 Found.");
configureSensor(&mpu4); //Configure sensor

// Initialize MPU6050 5
tcselect(7);
while (!mpu5.begin()) {
    Serial.println("Failed to find MPU6050 5 chip");
    delay(200);
}
Serial.print("MPU6050 5 Found.");
configureSensor(&mpu5); //Configure sensor

/* Display some basic information on this sensor */
/*tcselect(1);
displaySensorDetails(&mpu1);*/
}

void loop() {

    tcselect(1);
    getSensorDataX(&mpu1, 1);
    delay(7);

    tcselect(2);
```

```
getSensorDataX(&mpu2, 2);  
delay(7);  
  
tcaselect(3);  
getSensorDataX(&mpu3, 3);  
delay(7);  
  
tcaselect(4);  
getSensorDataX(&mpu4, 4);  
delay(7);  
  
tcaselect(7);  
getSensorDataX(&mpu5, 5);  
delay(120);  
}
```