Universidade Estadual Paulista

Instituto de Biociências, Letras e Ciências Exatas

Departamento de Ciência da Computação e Estatística

Gabriel José Pellisser Dalalana

Autonomous Mobile Robot Navigation Based On Omnidirectional Voice Commands

São José do Rio Preto - SP

2023

Gabriel José Pellisser Dalalana

Autonomous Mobile Robot Navigation Based On Omnidirectional Voice Commands

Monografia apresentada ao Programa de graduação em Ciência da Computação da UNESP para obtenção do título de Bacharel.

Orientador: Prof. Dr. Rodrigo Capobianco Guido

São José do Rio Preto - SP
2023

D136a

Dalalana, Gabriel José Pellisser

Autonomous mobile robot navigation based on omnidirectional voice commands / Gabriel José Pellisser Dalalana. -- São José do Rio Preto, 2023

47 f.

Trabalho de conclusão de curso (Bacharelado - Ciência da Computação) - Universidade Estadual Paulista (Unesp), Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto

Orientador: Rodrigo Capobianco Guido

1. Reconhecimento de Voz. 2. Inteligência Artificial. 3. Robótica. I. Título.

Agradecimentos

Gostaria de dedicar e agradecer este trabalho primeiramente aos meu pais, por terem me ajudado em toda minha trajetória de vida, nos meus estudos e no meu crescimento como pessoa. Também gostaria de dedicar meu projeto, e agradecer, a todas as incríveis pessoas que conheci, a meus amigos que sempre me apoiaram e me motivaram, e a todos os professores que instigaram curiosidade e aplicabilidade sobre aquilo que estudo. Afinal, é como dizem: “sobre os ombros de gigantes.”

"There was a time when humanity faced the universe alone and without a friend. Now he has creatures to help him; stronger creatures than himself, more faithful, more useful, and absolutely devoted to him. Mankind is no longer alone."

Dr. Susan Calvin. I, robot

Resumo

DALALANA, G. J. P.. *Autonomous Mobile Robot Navigation Based On Omnidirectional Voice Commands*. 2023. 47f. TCC UNESP 2023.

Atualmente, existe uma falta de pesquisas que envolvam o emprego da língua portuguesa como mecanismo de controle de robôs móveis. Aplicando os conceitos de energia, entropia e *zero crossing rate*, foi possível extrair as características principais contidas em cada uma das quatro classes de comandos de voz reconhecidas pelo dispositivo: frente, trás, direita e esquerda. Obtidas as informações que melhor separam as classes, um classificador do tipo *support vector machine* foi implementado, visando classificar corretamente comandos de voz, da categoria *speaker-dependent*, externos ao conjunto de treinamento da rede neural artificial. Por fim, o classificador foi embarcado em um robô móvel, permitindo a navegação baseada em língua portuguesa.

Palavras-chave: Reconhecimento de voz. Inteligência Artificial. Robótica.

Abstract

DALALANA, G. J. P.. *Autonomous Mobile Robot Navigation Based On Omnidirectional Voice Commands*. 2023. 47f. TCC UNESP 2023.

Nowadays, there is a lack of research involving the use of the Portuguese language as a control mechanism for mobile robots. Applying the concepts of energy, entropy and zero crossing rate, it was possible to extract the main characteristics contained in each of the four classes of voice commands recognized by the device: front, back, right and left. Obtaining the information that best separates the classes, a support vector machine classifier was implemented, aiming to correctly classify voice commands, from the speaker-dependent category, external to the training set of the artificial neural network. Finally, the classifier was embedded in a mobile robot, allowing navigation based on the Portuguese language.

Keywords: Speech Recognition. Artificial Intelligence. Robotics.

List of Figures

List of Tables

List of Abbreviations

ZCR	*Zero Crossing Rate*
SVM	*Support Vector Machine*
GPIO	*General Purpose Input/Output*
ALSA	*Advanced Linux Sound Architecture*
RBF	*Radial Basis Function*

Summary

Chapter 1

Introduction

1.1 Initial considerations

Since ancient times, the idea of creating beings/machines endowed with intelligence, which could coexist in the physical world with human beings, was already something envisioned by engineers and thinkers. One can easily find descriptions of theatrical mechanisms implemented in Ancient Greece, in which actors imitated gods who intervened in conflicting situations during the plot of the play, making use of extraordinary mechanisms that caused great impact on the listening public, to the point that such a climax received the Latin term *Deus-Ex-Machina*, which means “god of the machine” [1]. Moving forward into the Modern Era, Leonardo da Vinci envisioned what would become Leonardo’s Mechanical Knight, a humanoid automaton (a type of mechanism that can mimic characteristic human movements) that drew on da Vinci’s anatomy studies, applying it to in medieval armor in order to perform moves that would make him comparable to a person [2]. Finally, moving forward to the Contemporary Era, Czech author Karel Čapek, through the elaboration of his play Rossum’s Universal Robots, for creating the term “robot”, which is the fusion of the words “compulsory work” and “servant” [3]. From Čapek’s definition, it is possible to define the main applications and motivations that are studied within the robotics area.

Thanks to the great scientific and technological advances that have taken place, especially during the 20th century, such mechanisms were not restricted to mere puppets with their movements dictated by human beings, which ends up leading to the need for a definition of what such machines, now called *robots*, are: an autonomous system existing in the physical world, which can feel and act on the environment in which it finds itself in order to achieve certain goals [3]. Such a definition will mark the entire execution of the project, since most of the research in the field of robotics currently launched does not focus exclusively on purely industrial applications (here, thinking and feeling your environment is practically unnecessary, as the entire assembly line has already been inserted in the control system; therefore, by the definition above, such a mechanism is not a real robot), but in machines capable of interacting autonomously (without external control) and routinely with people and objects in the most efficient way (less movements needed) possible. The robot idealized for this work will follow these principles.

1.2 Objectives

Motivated by the above considerations, this work seeks to generate a robotic device that can be inserted in the last definition. In addition, after an in-depth research using the *Web of Science* tool, we found few articles that proposed the use of the Portuguese language applied in the context of robotics, more specifically in speech recognition. Therefore, this work also seeks to integrate voice commands, in Portuguese, as the main form of interaction between humans and the robot, aiming to correct this scenario, and be a reference for future research.

1.3 Computational motivation

A computational motivation for carrying out this project is the implementation of a classifier that, through certain characteristics extracted from voice signals given as input to the robot, can predict which command is ordered in Portuguese. As discussed in the previous section, the use of Portuguese as a language to control mobile robots is still scarce, so inserting such a classifier in this context will help to advance, even in small steps, a little in this area.

1.4 Work organization

In order to achieve the proposed goals and for a better understanding, this work is organized as follows:

- Chapter 2 will present the main topics to be addressed for the development of the proposed robot and, at the end, some related works involving the topics described will also be discussed.
- Chapter 3 presents, in detail, all the study carried out in the proposed work and what were the components applied in order to achieve the objectives proposed in this work.
- Chapter 4: all the results obtained in the work are reported, from the recognition tests that were carried out.
- Chapter 5: conclusions about the work are presented, as well as proposals for future research.

Chapter 2

Literature Review

In this section, the fundamental points for the structuring of the work will be presented. In the first three sections, the theoretical foundations required for the feasibility of this work will be presented and detailed. Following this introduction, some of the most recent research involving the previously discussed fundamentals will be presented.

2.1 Locomotion

Since the idealized robot needs to be included in the definition presented in Section 1.1, a direct approach is to choose the field of mobile robots, more specifically the means of locomotion, as a starting point for the elaboration of this project. As stated in [4] and [5], there are several ways to implement mobility in robots, but in order to maintain the simplicity and feasibility of this work, two main categories of wheel-based locomotion will have a deeper focus:

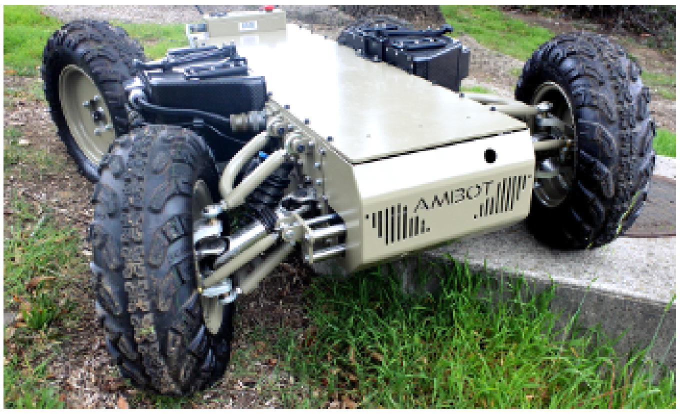

Figure 2.1 – An example of four-wheel independent drive robot. Adapted from http://www.ambot.com/ip-wheel.shtml

2.1.1 Wheeled locomotion

The most popular locomotion mechanism in mobile robots, the wheel offers a simple mechanical implementation and, as reinforced in [3], static stability (the robot does not need to spend work to maintain its balance in the ground, the wheel contact on the floor occurs all the time). An example of a mobile robot with wheels is shown in Figure 2.1. Some problems arise, mainly in traction, control and maneuverability, when this type of mechanism is inserted in unstructured environments (for example, an external area of a building, where human infrastructure tends to be non-existent).

2.1.2 Tracked locomotion

This design was created with a focus on use on uneven terrain, the track implies greater contact with the ground compared to simple wheels, a fact that significantly improves its maneuverability in such a scenario. An example of a crawler mobile robot is shown in Figure 2.2. Due to the slip/skid format of this type of locomotion mechanism, dead reckoning (a spatio-temporal analysis that tries to predict the real position of the robot) is considerably impaired

Figure 2.2 – An example of tracked robot. Adapted from https://research.csiro.au/robotics/our-work/darpa-subt-challenge-2018/

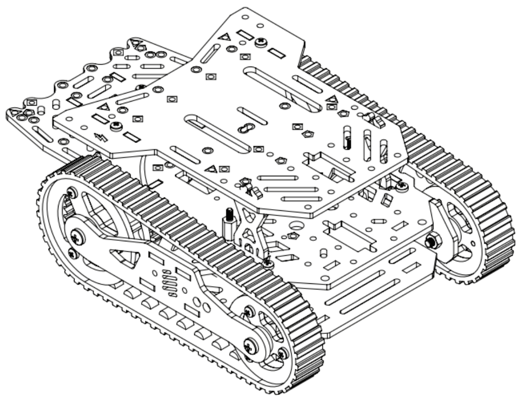

Figure 2.3 – Assembly model of the adopted robot basis for this work. Adapted from https://s3-sa-east-1.amazonaws.com/robocore-lojavirtual/1172/manual_expansao_rocket.pdf

due to the increase in system error. Thus, a more complex positioning mechanism needs to be implemented for this type of locomotion. Also, considering the energy efficiency, this mechanism can only work well on loose terrain, being significantly inefficient otherwise. In the end, this mechanism proves to be ideal in unstructured environments (where the proposed robot tends to be applied), when the advantages outweigh the disadvantages, so this is the type of locomotion mechanism that this work will adopt. Figure 2.3 shows the chosen robot base assembly model.

2.2 Navigation

Navigation can be defined as the way a robot finds its way in the [3] environment. This is an essential question for most robots that can move anyway, being one of the oldest and most research-intensive areas within mobile robots. For navigation to occur, some steps need to be defined and entered, being then:

2.2.1 Localization

In the words of [3], *localization is the process of figuring out where the robot is relative to some model of the environment, using whatever sensor measurements are available*. With this definition, the robot can find itself using several methods, such as vision, GPS measurements or odometry (the process of measuring physical distances using, for example, the number of rotations of a motor), the latter being one of the most used to solve the localization problem. For this work, in addition to knowing the disadvantages imposed in the use of techniques based on odometry, e.g., the inherent uncertainty present in physical measurements (in [6], a deep analysis of the propensity to odometry errors is presented), something that can cause serious errors if not handled correctly, the proposed robot will apply this technique as the first way to implement the localization.

2.2.2 Mapping

After defining the localization mechanism to be implemented, the next step is to define how the robot will build its environment map (an information structure that contains a considerable

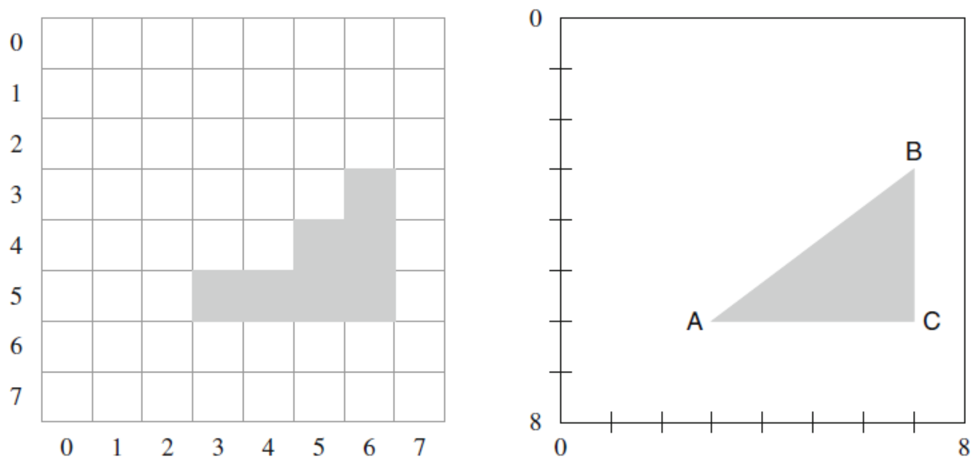

Figure 2.4 – An example of, in left, a discrete map and, in right, of a continuous map. Adapted from [6]

range of data related to the location where the robot is). To store a map, two methods, as indicated in [6], can be used: discrete maps and continuous maps. The first is the most common way of representing information from the environment, which is also adopted in this work. In Figure 2.4, there is an example of a discrete map and a continuous map.

With the information storage chosen, now we need to define how this map will be created. Considering a consecrated technique for this question, this work will use the Frontier Algorithm to address this question. This algorithm was proposed in 1997 by Brian Yamauchi, making use of discrete maps (here, Yamauchi used a *grid map*, a type of discrete map proposed by [8] that uses the probability concept of a grid [a region] is occupied) to start exploring a new place. The idea of this algorithm, in the words of [7], is: *To gain the most new information about the world, move to the boundary between open space and uncharted territory*. Thus, this mapping algorithm ensures that the robot discovers all reachable spaces by moving the robot to the regions of the map that are on the boundary of known and unknown spaces.

In Figure 2.5, we can see an iteration, and the main idea, of the Frontier Algorithm: the robot (represented by the blue arrow) moves to the nearest frontier (the boundary region) on the grid map. After arriving (represented by the blue square) at the border, it uses its sensors to map this unmapped location. For example, in Figure 2.6 we can see that the robot discovered new free and occupied regions (a free region is represented by the probability of 0.1-0.2, and a region occupied by the probability of 0.9-1.0 [this range is due to the uncertainty of the sensor measurements]), as well as a new frontier to be explored.

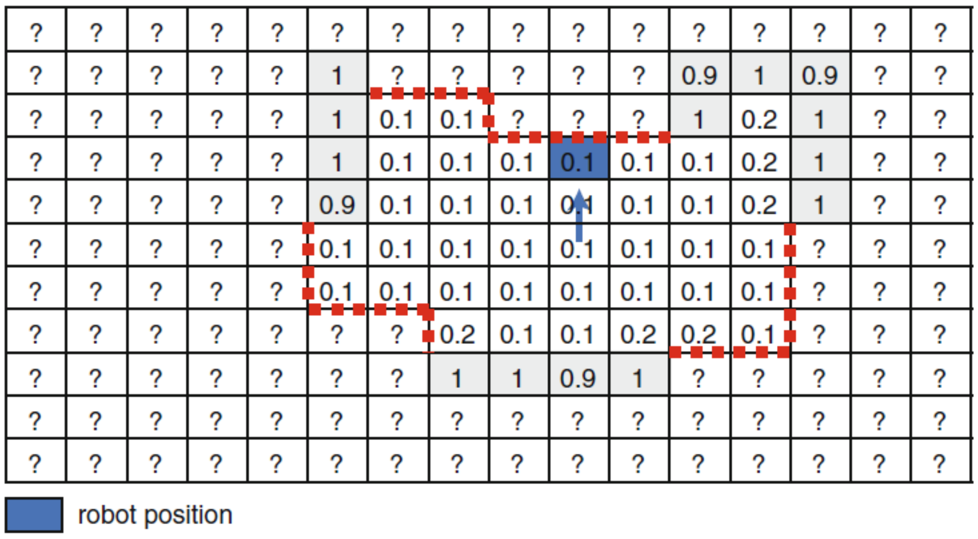

Figure 2.5 – The movement of a robot to a frontier. Adapted from [6]

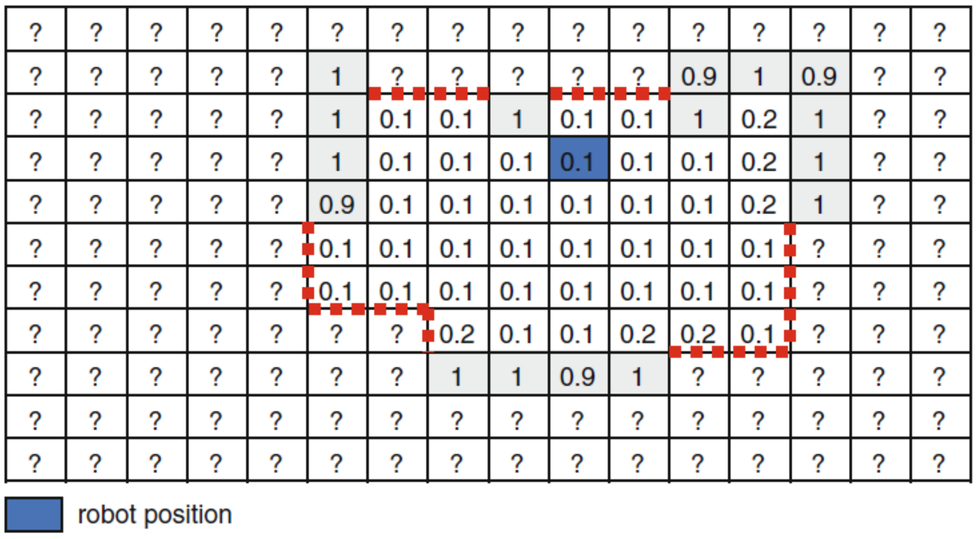

Figure 2.6 – An iteration of the Frontier Algorithm. Adapted from [6]

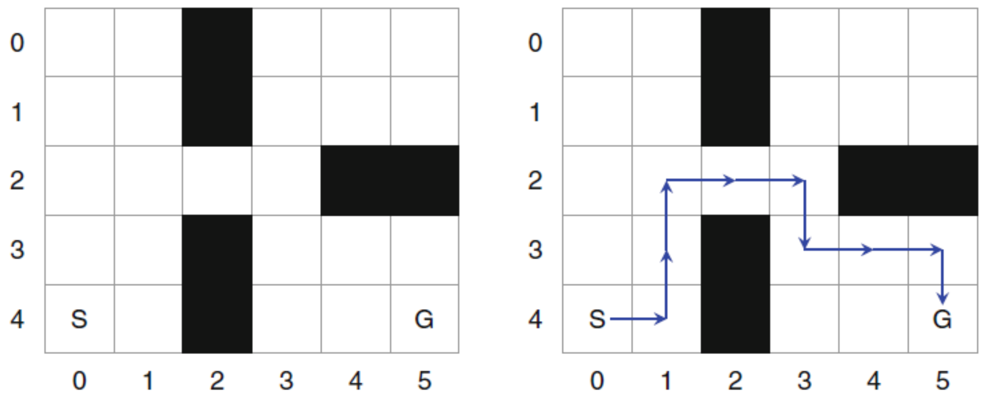

Figure 2.7 – An example of Dijkstra's Algorithm for a grid map. In left, the grid map obtained by the robot; in right, the shortest path found by the algorithm. Black cells represent occupied cells. Adapted from [6]

This algorithm was chosen because, as demonstrated by [7], it can map open and closed places. Here, a closed location might be an office (the first location where Yamauchi ran the program), a warehouse; and a open place a park, a plantation. Considering that the proposed robot can be used in both scenarios above, this algorithm fits perfectly for our purposes. Also, as stated by [4], this algorithm is very popular in the field of mobile robotics.

2.2.3 Path planning

The final big question related to robot navigation, path planning is how the robot will go from an arbitrary point A to a point B, with $A \neq B$. As shown in [6], there are two main approaches to dealing with this issue:

Dijkstra's Algorithm for a Grid Map

Let cell S be the starting point of the robot's movement, and cell G the objective of this movement. With this approach, the robot detects and moves to a neighboring cell, aiming to reach the shortest possible path. In Figure 2.7, we can see the behavior of the algorithm.

A* Algorithm

Instead of searching for the objective cell in all directions, as in Dijkstra's Algorithm, the A* Algorithm iterates its search only in the cells that tend to be closer to the objective cell. This approach can be useful in relatively simple environment maps, saving considerable runtime.

2.3 Voice Commands Recognition

The last major pillar to be addressed by this work, the recognition of voice commands by the desired robot will follow a restricted-vocabulary speech recognition approach proposed in [9]. In this work, the author used a handcrafted feature extraction based on entropy (here, entropy can be understood as a measure of unpredictability of the information content of any entity) on raw variable-length data in wave format [10] , more specifically, in the pronunciation data of Portuguese words, such as "esquerda", "direita". With the method described in [9], these words can be grouped together, making it easier to differentiate between such words. This work will apply the above method as a way for the robot to recognize voice commands in Portuguese.

Furthermore, in order to increase the quality of feature extraction for each voice signal, other digital signal processing techniques will also be applied. In this context, feature extraction based on energy and zero-crossing rate (ZCR) will be performed, following the description presented in [19] and [20]. Applying an energy analysis, it may be possible to identify ranges that differ from the ambient noise, focusing on the signal points that contain the most relevant information. This specific technique was inserted as a filter for the voice signal captured by the robot's microphone, and also as a data pre-processing tool, as described in Section 4.1. Now, a ZCR analysis considers the rate of change from positive-to-zero-to-negative or vice versa of a signal, something particularly useful for detecting when speech is present in the voice command.

In addition to using entropy-based handcrafted feature extraction, this work will also implement an omnidirectional wave format data input. This choice is made with the aim of expanding

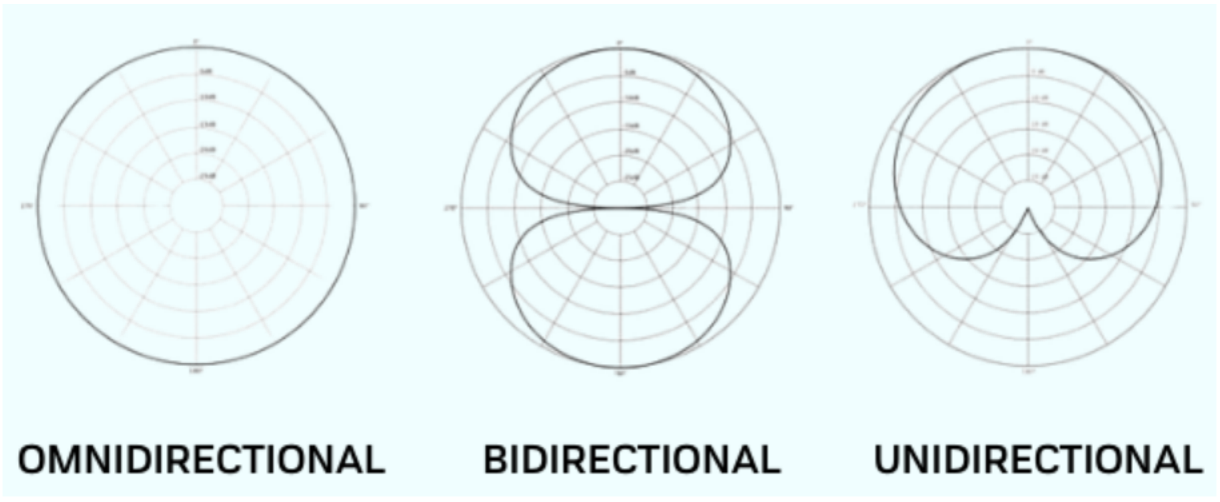

Figure 2.8 – Differences between an omnidirectional (in left) and more focused microphones (in center and in right). Adapted from https://myelearningworld.com/what-is-an-omnidirectional-microphone/

the capabilities for the robot to understand voice commands in multiple directions, rather than focusing on just one audio direction. An illustration of an omnidirectional microphone can be found in Figure 2.8.

2.4 Related Work

Starting with the Frontier Algorithm concept, its popularity still persists today. Aiming to integrate this approach in the Robot Operating System, a consolidated tool-based robotic software development [11], [12] created a package for this tool that inserted this algorithm, making it easier for many researchers to add the Frontier approach in their works. Despite the openness made possible by this last work, one of the most challenging problems related to this algorithm is the correct detection of a boundary, and how to deal with the growing demand for computational resources proportional to more detailed maps. Considering the first factor, [13] proposed an algorithm called Frontier-Tracing Frontier Detection that uses the perimeter of the sensor's current observation in combination with the laser scan endpoints and previously known boundaries to perform the frontier detection, this approach, as concluded by the article, is best suited for applications that require boundary detection to be implemented after each observation. Now, with respect to the last factor, [14] investigates the implementation of this algorithm using the idea behind a randomly exploring random tree. With this approach, after the robot passes through a boundary area, all tree branches in that area are removed to free up memory and speed up the calculation process.

Now, taking the concept of odometry, [15] proposed a way to use odometry, in this work it is a visual odometry, to facilitate the process of locating and mapping a mobile robot. Its approach can overcome internally based odometric measurements (as the proposed robot will adopt), enabling real-time oriented images for decision making, being a possible new feature to be inserted in a new version of this work. Moving forward, in [16] the authors also applied a visual odometry approach to complement the classical odometry measurement in robotics, such as the engine rotation counter, described in Section 2.2.1. They obtained considerable results by merging this approach with deep learning models, more specifically in an edge-oriented environment (the proposed robots are also inserted in this type of environment).

In the end, considering the interpretation of voice commands, [17] encountered the same problems as this work, the scarcity of speech recognition systems applied in robotics in its native language, in this case Indonesian. The authors implemented a Deep Neural Network to train and deploy a classifier to handle speech recognition, achieving good results in the real world scenario. With this last work, the justification of this intended project is reinforced. Furthermore, in [18] the authors also deal with the same kind of space destined for our proposed robot, an environment with possible multiple speakers. They proposed a Neural Network for simultaneous detection and localization of speakers, with the interesting point being the possibility of applying a previous feature extraction step in the network, such as one based on entropy (described in Section 4.1).

Chapter 3

The Proposed Work

As stated in the last sections, this work proposes the creation of an autonomous mobile robot that can interpret voice commands, obtained through an omnidirectional microphone, to perform tasks related to the principles of navigation.

This work was carried out by the author, with the supervision of the guiding professor, so that periodic meetings for guidance were made possible. Initially, a review of the elementary physical concepts applied in this work involving the voice signal processing area was carried out, based on the references [19], [20] and [9]. Based on the tools that such works proposed for extracting features from the analyzed signals, it was possible, as described in the next section, to recognize restricted vocabulary voice commands, that is, the recognition of a few words [9].

3.0.1 Initial Studies and Chosen Materials

After such a review, studies were also carried out involving the fundamentals of the field of robotics relevant to the applicability of what this project proposed to accomplish. Initially, the studies focused on two main works, [3] and [6], given their introductory character, but complete, on all the topics necessary for the feasibility of the mobile robot, such as kinematics and control theory. After this step, and realizing the need to review the concepts of electrical and electronics, especially those of immediate insertion during the structural construction phase of the robotic mechanism, the work [5] was also used. Finally, after studying the fundamentals necessary for building the robot, there was one last topic to be reviewed: mobility. As much as previous works routinely mention this feature, the great importance of choosing the locomotion mechanism to define what the device can or cannot accomplish led to an in-depth study on locomotion, marked by the use of the work [4].

Having carried out the conceptual review of the areas of knowledge linked to the project, the next step was the choice of the required materials, namely:

1. 1x Raspberry Pi 4 Single Board Computer, model with 8 GB of RAM memory;
2. 1x Multilaser Industrial SA PH361 Omnidirectional Microphone; and
3. 1x Robot Explorer Kit, with expansion, by Robocore Tecnologia LTDA.

As described in [30], choosing the Raspberry Pi board, illustrated by Figure 3.1 brings numerous advantages:

- Speed and computational power: the board has a microprocessor capable of supporting the full execution of an operating system, specifically versions of Linux. Added to this fact, and as its name indicates, this small computer allows the use of the Python programming language (considerably slower and computationally more expensive than C/C++), ideal for tasks that involve the application of functionalities contained in libraries on the processed data, such as image processing and, within the scope of this project, voice signal processing; and
- Connectivity and network: as it has four USB-type inputs, HDMI input, Wifi and Bluetooth already integrated into the hardware, and dozens of General Purpose Input/Output connections (GPIO), the board provides several possibilities for integration of sensors and external equipment. Taking as an example what was witnessed in the execution of this project, the USB port was used to easily integrate the microphone directly into the operating system, while the HDMI input and two more USB ports were used to connect an external monitor, keyboard and mouse in the board configuration step. In addition,

GPIOs pins were intensively used to connect ultrasound sensors, used to allow navigation in the robot's work environment.

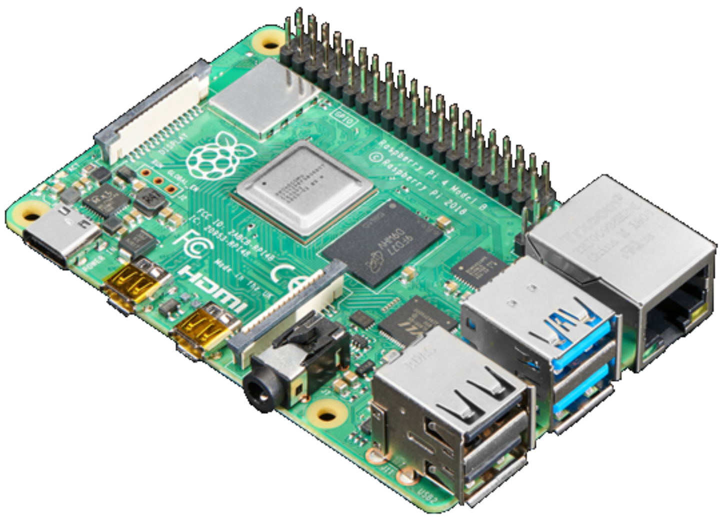

Figure 3.1 – Image of the Raspberry Pi 4 board. Adapted from https://www.robocore.net/placa-raspberry-pi/raspberry-pi-4-8gb

Moving now to the reason for choosing an omnidirectional microphone as the main way of capturing voice commands by the robot, an explanation is facilitated by the illustration in Figure 2.8. As you can see, when choosing an omnidirectional microphone, it becomes feasible to capture sound signals, here characterized by voice commands, in multiple directions, a clear advantage compared to configurations that deal with obtaining sounds in different directions. Specifically, such as coming from a source located in only one or two locations. Even more, considering that the robot will be, during its operation, inserted in unstructured environments, [3] and [6], sources of voice commands can change constantly while the robot moves. This operating scenario illustrates why an omnidirectional microphone was chosen.

Finally, for the issue of robot assembly, a pre-assembled robotic platform was chosen, resulting in the design shown in Figure 2.3. The final version of the mechanism, including the installation of the microphone, the Raspberry Pi 4, and the ultrasonic sensors, can be seen in Figure 4.9. The reasons for choosing this particular platform are as follows:

1. The modular design, based on floors, allows the attachment of multiple sensors and boards of different sizes and complexities;

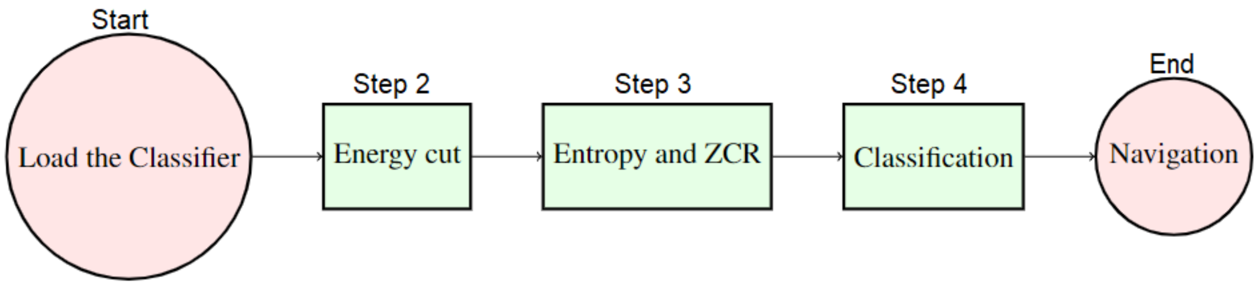

Figure 3.2 – The algorithm proposed in this work. Source: author.

2. Inclusion, together with the structural parts, of all the components necessary for the correct operation of the robot, such as batteries, motors and a microcontroller board to do the “low-level” work of the platform, that is, if the program contained in the Raspberry Pi identify, for example, the word “esquerda”, it will communicate this microcontroller board, which in turn will carry out the maneuver, via pulse modulation to the motors; and

3. Because it has a locomotion system based on treadmills, it makes the mechanism ideal for operations in unstructured environments, such as external environments with a lack of basic infrastructure [4].

In short, the design proposed by the kit fit perfectly with the scope of the project, and its acquisition allowed the concentration of research efforts on the project’s central intelligence issue, that is, the recognition of voice commands with restricted vocabulary and subsequent decision making.

3.0.2 The Proposed Algorithm

In Figure 3.2, the proposed algorithm to meet the Objectives section in Chapter 1 is illustrated. For each of the development steps, we have:

- **Step 1 (START):** in contrast to common workplaces where sorters are tested, which have normally constant power supply, for mobile robots this is not true. In order to avoid doing the same training twice, the Joblib [26] library was applied to first save the state of the model in the permanent memory of the Raspberry Pi board and, considering the developed algorithm, reload it for use. This particular library was chosen because it is optimized to deal specifically with *numpy* arrays [27], the type of data structure adopted to store the main characteristics of the training dataset, discussed in the next section.

- **Step 2:** after restoring the state of the classifier in the runtime memory, the next step is to “listen” to the voice commands. After pressing a physical button on the robot’s chassis, a slice lasting 3 seconds was recorded using Linux ALSA(Advanced Linux Sound Architecture). The main reason for implementing forced speech recognition instead of real-time recognition is the computational constraints of the robot.

  With the possible recorded voice command, an energy-based cut is inserted into the algorithm design. Applying this digital signal processing technique, the environmental noise recorded before and/or after a noticeable energy change occurs (for the robot’s workplace, this probably indicates a voice command) is removed to cut the extra computation of the robots. entropy and ZCR calculations, speeding up these calculations six times on average. This technique starts by calculating the total energy of the signal, using the following mathematical expression: $E(s[.]) = \sum_{i=0}^{M-1}(s_i)^2$, where M represents the length of the signal $s[.]$ [19].

  It is important to note that $s[.]$ was obtained using Python’s SoundFile library [21] applied to the record. Furthermore, while the main code, that is, the program that runs until the robot is turned off, is in Python language, energy, entropy and ZCR calculations are coded in C++ language, aiming to speed up these costly techniques. Therefore, when one of these methods is needed, the main code executes a system call to the required C++ code.

  Once the total energy is found, a control variable is declared containing 95% (tends to be less aggressive when removing information, smaller values can significantly modify the signal format) of the total energy of the signal. This is the core of this type of cut. At the end, two pointers are declared at the beginning and at the end of the original signal, moving forward and backward, respectively, until they find the signal point (both in the initial and final part) that contains this energy, and saving the signal without the other analyzed points.

- **Step 3:** with the filtered signal, the concepts of entropy (amount of information required to specify the system microstate), defined as $H = -\sum_{i=0}^{K-1} p_i \cdot \log_\beta(p_i)$ [9], and ZCR (rate of change from positive-to-zero-to-negative or vice-versa of a signal), expressed as $ZCR(s[.]) = \frac{1}{2}\sum_{j=0}^{M-2} |sign(s_j) - sign(s_{j+1})|$ [20], are calculated, since, together in the same data structure, they reach the best accuracy of the classifier, as shown in the next section.

- **Step 4:** with all pre-processing steps completed, the data extract from Step 3 is inserted into a Support Vector Machine (SVM) instantiated from Python's scikit-learn library [28], retrained in Step 1, and using *predict* method, the classifier's prediction is used to move the robot, via serial communication between the Raspberry Pi board and the robot's microcontroller, to the desired state.

- **Step 5 (END):** this final step uses the classifier response to bring the robot to the desired state. For example, if the command "frente" is recognized, the robot will move forward until it encounters an obstacle, an occupied grid in the context of the Frontier Algorithm.

Chapter 4

Tests and Results

4.1 Voice Data Pre-Processing

Several tests and experiments were performed in order to prepare the data for a further classification step. This primary data preparation step aims to extract a set of characteristics that facilitate the distinction of the worked information, eliminating an additional workload that the classifier would have to deal with. Taking as an example the extraction of features based on the entropy of the input signal [9], the total process to obtain the expected result, considering the preparation step discussed above, can be seen in Figure 4.1. It is worth mentioning that this process also has the same structure when working with the energy [19] and the ZCRs [20] of the input signal.

To obtain information regarding the voice signal worked on, the SoundFile [21] library,

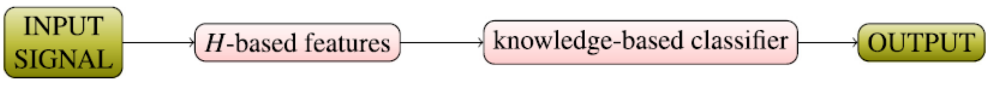

Figure 4.1 – A combination of the features obtained by calculating Entropy with the knowledge-based classifier. Adapted from [9]

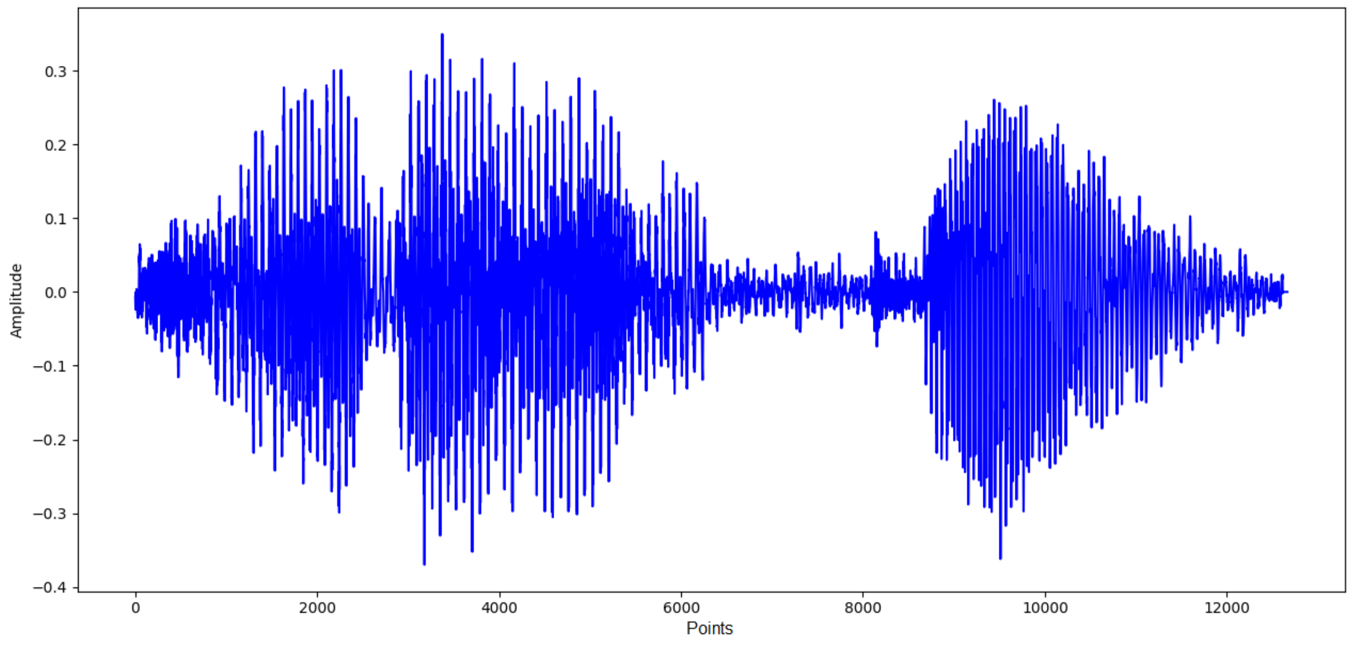

Figure 4.2 – An example of a discretized signal [29] for the "direita" command. Source: author.

present in the Python language, was used. The database used was presented in the article [9], containing 80 voice commands distributed in four classes: Right, Left, Up and Down, digitized at 16000 Hz, 16 bits, single channel. For this work, the Right and Left classes were used to represent the above digital speech processing techniques. Graphs that contain the behaviors present for a given command "direita" and "esquerda" are shown, respectively, in Figures 4.2 and 4.3.

After obtaining the information present in each of the samples present for these two classes, the next step was to apply the concepts of energy, entropy and ZCR on the data obtained, in order to obtain the characteristics that best define and differentiate each class. Both for energy and for ZCR, a characteristic vector of dimension 19 was obtained, while for entropy, this vector had a total dimension of 70. Following the standardization present in the figures that contain the information of the voice commands, the class Right is represented by the blue color, while the Left by red color. Starting with energy, the result obtained is represented in Figure 4.4. Returning now to entropy, the result obtained can be seen in Figure 4.5. Ending with the ZCR, the experimental result obtained is illustrated in Figure 4.6. Visually, it can be seen that the data can be separated in a binary way. Therefore, as a final step in the classification process, a binary classifier was applied, more specifically a support vector machine (SVM) [22].

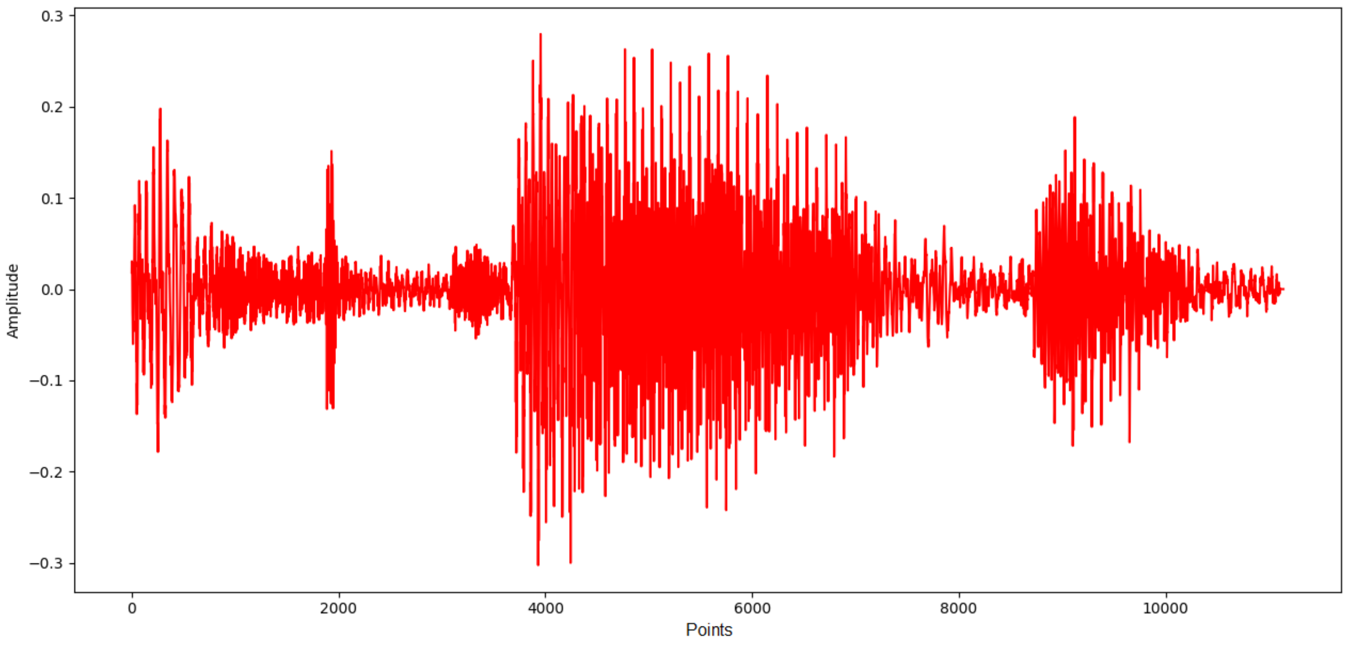

Figure 4.3 – An example of a discretized signal [29] for the “esquerda” command. Source: author.

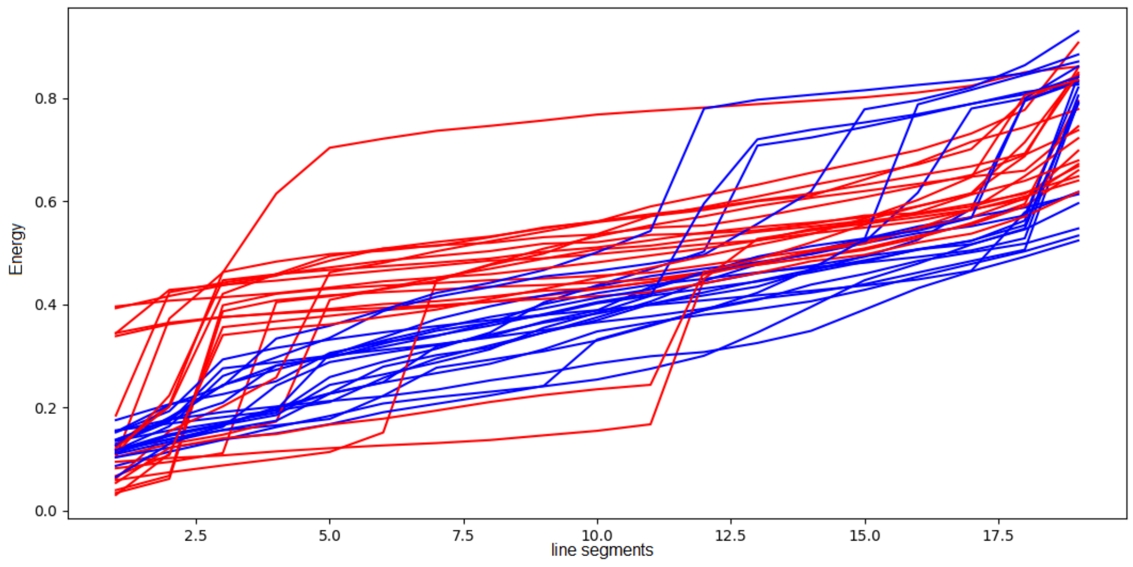

Figure 4.4 – The signal energy distribution obtained for the “direita” and “esquerda” commands. Source: author.

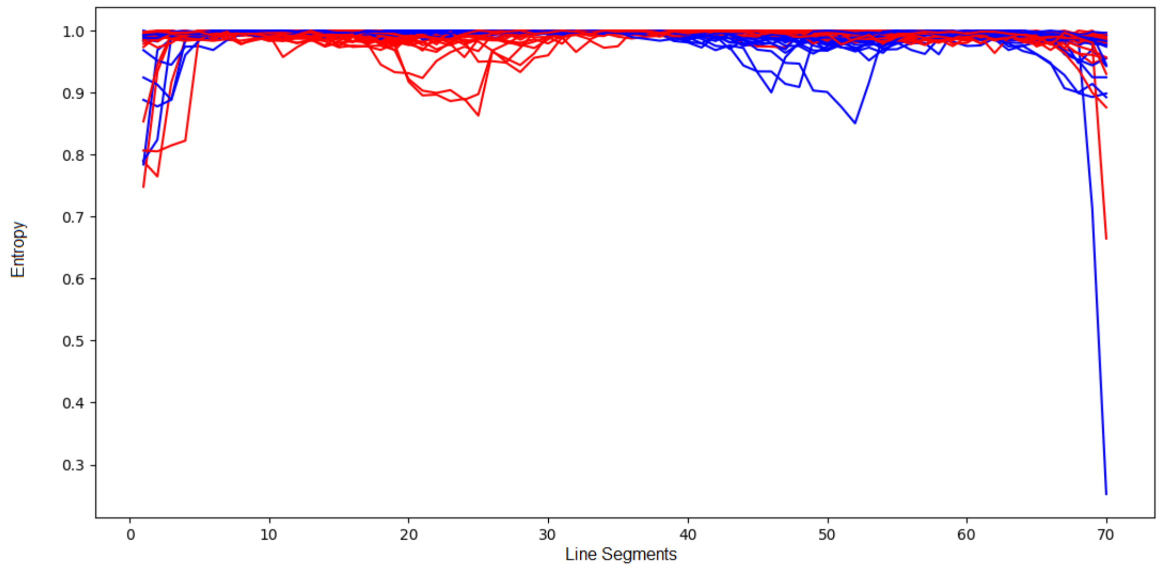

Figure 4.5 – The signal entropy distribution obtained for the "direita" and "esquerda" commands. Source: author.

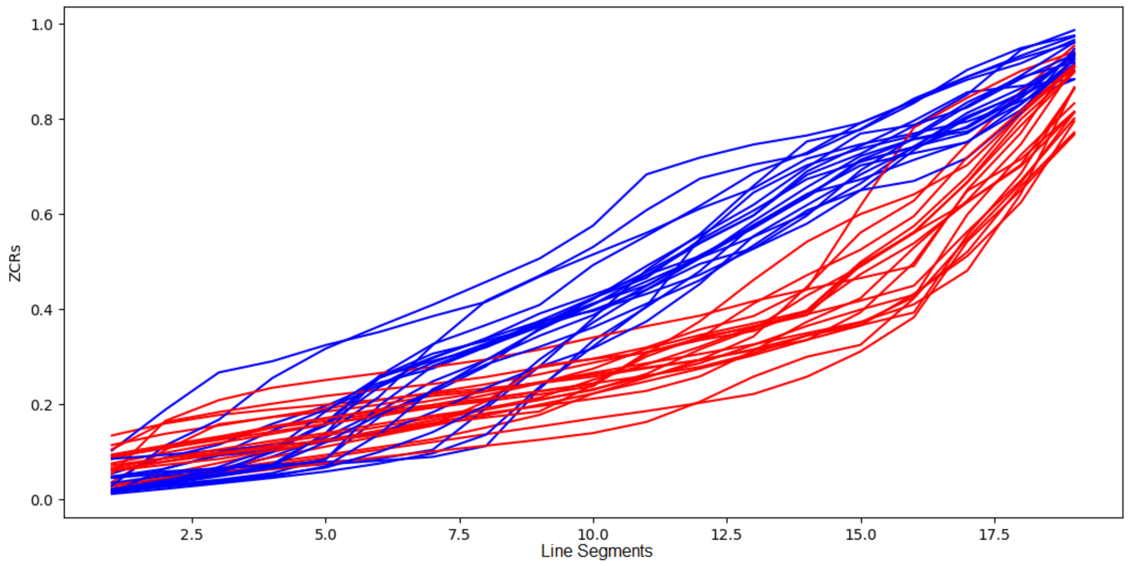

Figure 4.6 – The signal ZCR distribution obtained for the "direita" and "esquerda" commands. Source: author.

4.2 Classifier Training and Hyper-parameter Tuning

The SVM applied in this project has x inputs, where x is the dimension of each feature vector obtained in the previous step, with y active and non-linear processing elements in the hidden layer, where y is the number of training cases, and finally , containing 4 outputs, 4 being the number of words to be recognized.

However, as the microprocessor used to analyze the voice commands has considerably limited computational capacity, a pre-processing of the voice signals detected by the microphone was necessary. Taking a slice with a duration of 3 seconds that contains a movement command, represented here by Figure 4.7, which illustrates the four possible classes to be recognized by the classifier, a energy-based cut was necessary.

Two pointers were generated, one pointing to the beginning of the signal, and the other to the end of it. After that, the total energy of the voice command was calculated, using the concepts present in [19], and a variable that contains 95% of the total energy found was started. This value is essential, since signal portions that exceed this threshold will be discarded by increasing and decreasing the pointers. The effect of such preprocessing is illustrated in Figure 4.8. Notably, when comparing this figure with the previous one, it can be seen that the signal size was considerably reduced, making the subsequent calculation of energy, entropy and ZCR easier and more agile, ideal for the computational environment contained in the robot.

With the pre-processed data, the next step was to perform the adjustment of the SVM hyperparameters [22]. For that, three methodologies were used to adjust the separator plane: grid and random search, and Bayesian optimization. The initial method, that is, the grid search is an exhaustive search method that aims to find the best *kernel* and values of *C* and *gamma* [22] that can correctly separate the classes of the problem [23]. On the other hand, the random search makes use of the same essence as the grid, except that it places the probabilistic element to randomly choose a subgroup of the possible combinations of such hyperparameters and test them, making use of a minimal fraction of computational effort, compared to the grid search [24]. Finally, Bayesian optimization was also tested, a model that tends to be more efficient than the

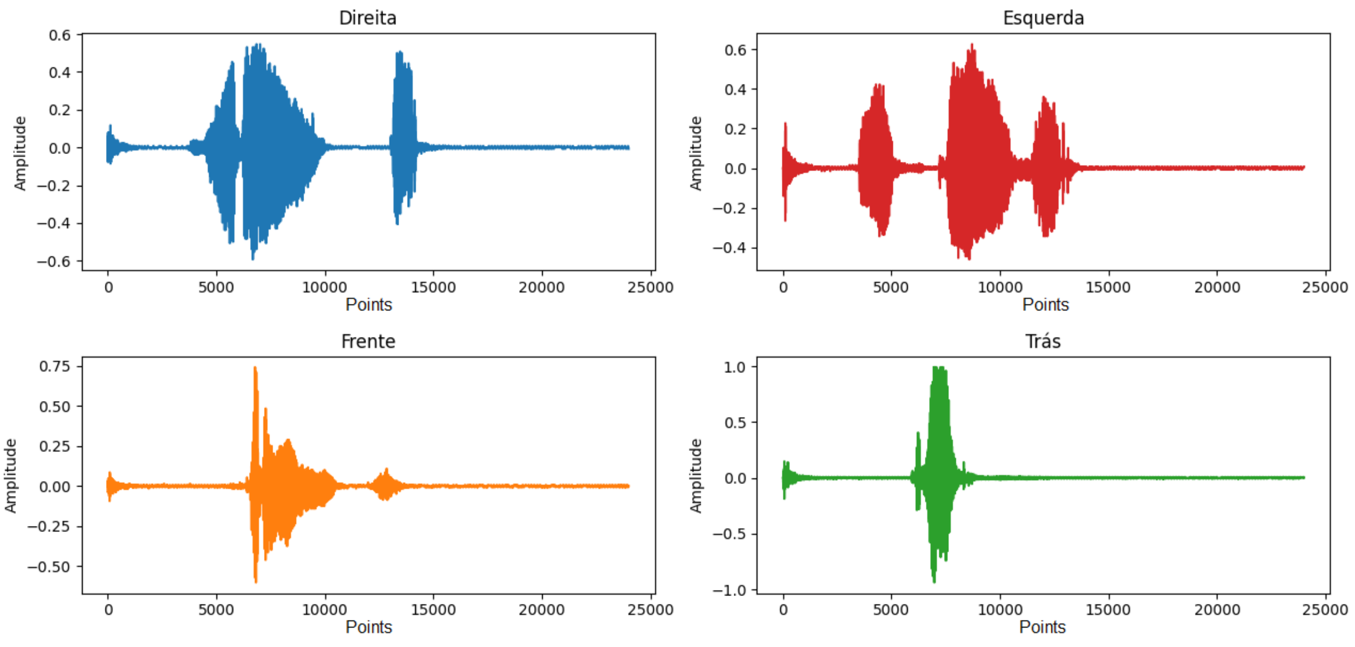

Figure 4.7 – The format of the signal obtained for the “direita”, “esquerda”, “frente” and “trás” commands. Source: author.

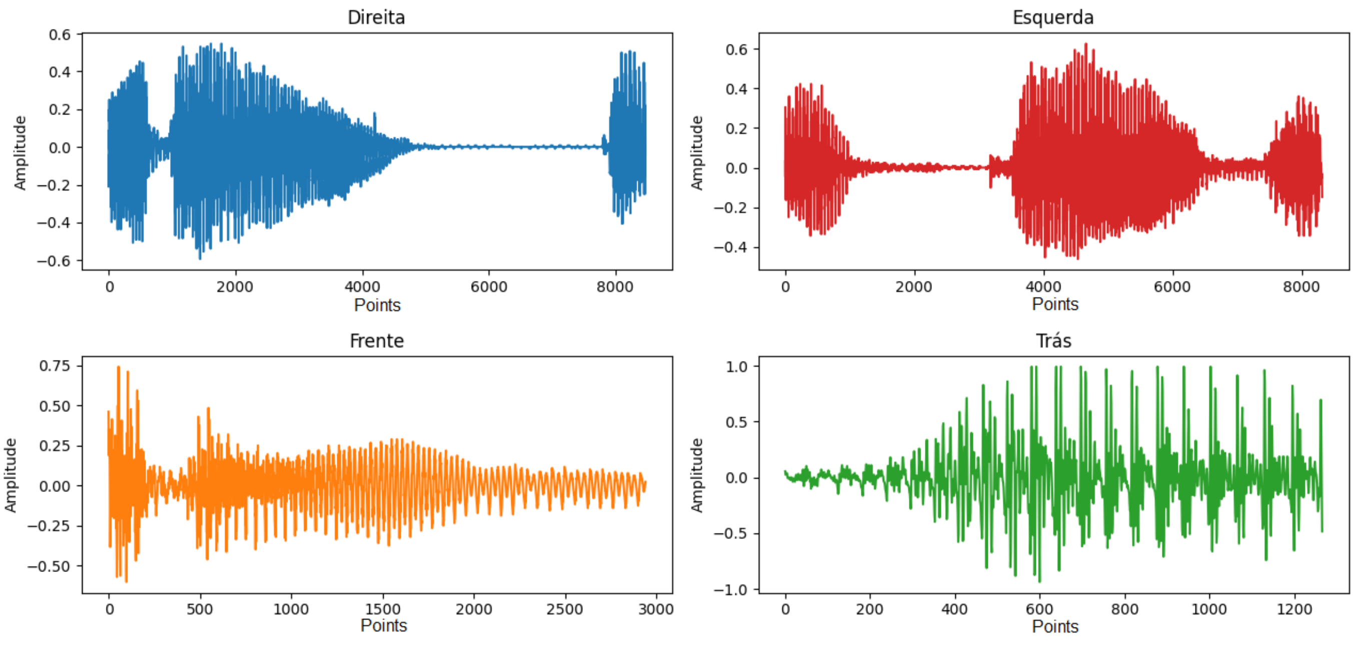

Figure 4.8 – The format of the signal obtained after energy-based filtering for the “direita”, “esquerda”, “frente” and “trás” commands. Source: author.

random search, as it makes use of pre-existing information from the tested models to choose the hyperparameters for the next attempt [25].

The results of accuracy and total time spent for training and running the classifier can be found in Table 4.1. It is emphasized that the results shown are the average of five consecutive executions, 10000 iterations of each algorithm, of the previously discussed adjustment methods, counting on a space of possible values of *C* and *gamma* of dimension 100, obtained through the function *logspace*, present in the NumPy library [27]. The training set consisted of 80 voice commands, distributed among the 4 possible classes, while the test set consisted of 20 voice commands, also distributed among the classes. From the table, it can be seen that the best "cost x benefit" was due to the use of entropy and ZCR data as a discriminating factor between the four possible classes to be recognized, this being the training data chosen for the final training of the classifier. For this reason, the following hyperparameters were obtained:

- Chosen kernel: polynomial;
- Optimal value of *C* found: 0.019179102616724848; and
- Optimal value of *gamma* found: 6280291441.834272.

Such values were defined as definitive hyperparameters of the network.

Just for comparison, Table 4.2 shows the same calculation results, but with a restriction on the number of iterations, now set to 1000. The space of possible values of *C* and *gamma* it still has a dimension value of 100. Analyzing these results, we can see that other kernel types may appear, such as Radial Basis Function (RBF) and Sigmoid and, consequently, new values for *C* and *gamma* will arise, with the disadvantage of loss of model accuracy. So, as a final note, it is crucial to set a large margin to scan the entire search space in order to find the best SVM hyperparameters using the analyzed data type.

With the results obtained by the classifier, it can be inferred whether a given voice command given by a speaker is or not a command present in the robot's vocabulary, and if so, which class it belongs to. With this distinction, the Raspberry Pi board (where the entire acquisition, extraction and classification process took place) communicates to the microcontroller responsible

Techniques	Best Accuracy (%)	Time Spent (ms)
Energy	90.00	72.00
Entropy	85.00	76.00
ZCR	95.00	68.00
Energy+Entropy	98.75	74.50
Energy+ZCR	97.50	71.98
Entropy+ZCR	99.07	71.36
Energy+Entropy+ZCR	99.34	91.98

Table 4.1 – Average of the results obtained from the parameters found by the adjustment methods applied in this work. Source: author.

Techniques	Best Accuracy (%)	Kernel
Energy	90.00	RBF
Entropy	90.00	RBF
ZCR	93.75	RBF
Energy+Entropy	90.00	Sigmoid
Energy+ZCR	95.00	Polynomial
Entropy+ZCR	97.50	RBF
Energy+Entropy+ZCR	95.00	Polynomial

Table 4.2 – Average of the results obtained from the parameters found by the adjustment methods applied in this work, with restriction of iterations. Source: author.

for controlling the robot's motors whether it should, for example, turn left or right. Even more, the control of the motors was given by throwing maximum voltage on them, that is, 5 Volts. The final shape of the robot can be seen in Figure 4.9, and a practical example of navigation movements in Figure 4.10.

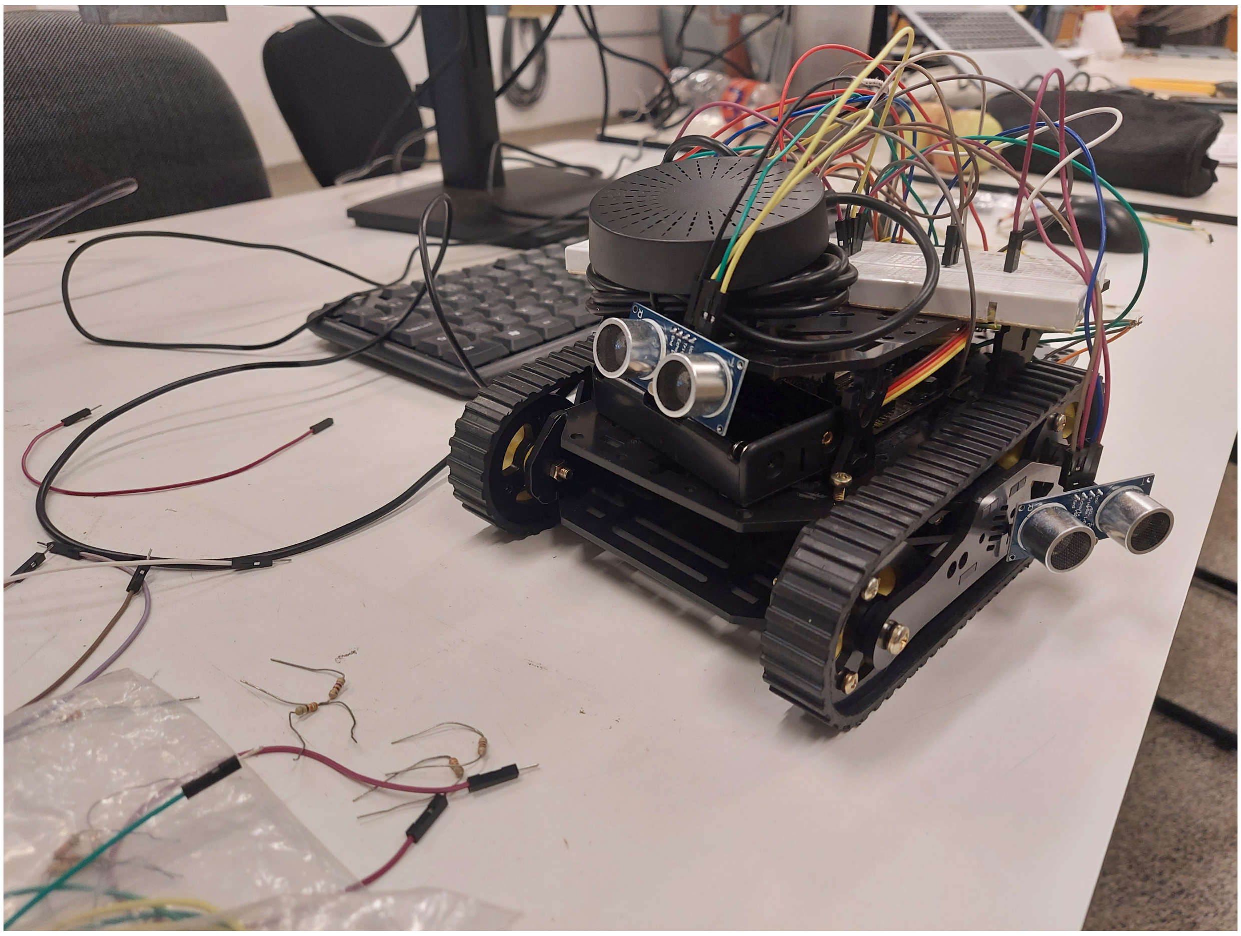

Figure 4.9 – The final shape of the robot. Source: author.

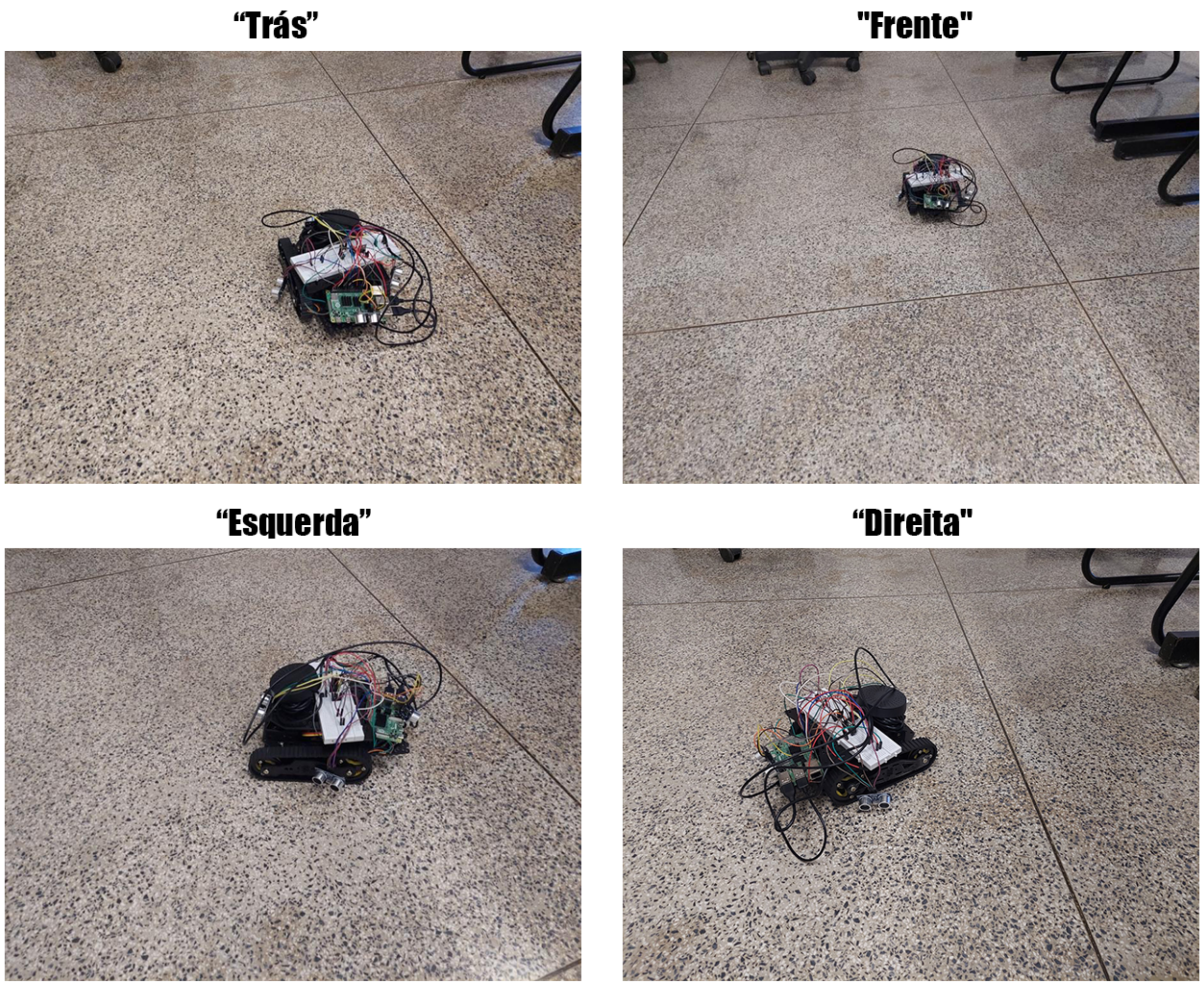

Figure 4.10 – An example of execution of “direita”, “esquerda”, “frente” and “trás” commands. Source: author.

Chapter 5

Conclusions and Future Work

As previously described in the Justification section of this report, the control of a mobile robot based on voice interaction in Portuguese is an area with very scarce resources. For this reason, this work was designed to serve as an initial tool to solve this issue. Even more, even with a considerably restricted vocabulary of voice commands, that is, few commands are actually recognized by the robotic mechanism, the results obtained corroborate to justify the possibility of, in future projects, carrying out an extension to increasingly larger vocabularies, making human-robot interaction increasingly natural and commonplace.

At the end of this work, I would very much like to emphasize that gaining experience, whether working with digital speech processing or in the area of mobile robotics, will probably be with me for the rest of my professional career as a computer scientist. Here, I was able to understand in depth how the scientific method works in the real world, solving applied projects.

Bibliography

[1] CHONDROS, T. G. et al. “Deus-Ex-Machina” reconstruction in the Athens theater of Dionysus. *Mechanism and Machine Theory*, v. 67, p. 172-191, 2013.

[2] MORAN, M. E. The da Vinci Robot. *Journal of Endourology*, v. 20, n. 12, p. 986-990, 2006.

[3] MATARIĆ, M. J. The robotics primer. *MIT press*, 2007.

[4] SIEGWART, R., NOURBAKHSH I. R., and SCARAMUZZA, D.. Introduction to Autonomous Mobile Robots - 2nd ed.. *MIT Press*, 2011.

[5] MCCOMB, G. Robot Builder’s Bonanza. 2019.

[6] BEN-ARI, M.; MONDADA, F. Elements of robotics. *Springer Nature*, 2017.

[7] YAMAUCHI, B. A frontier-based approach for autonomous exploration. In: Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA’97.’Towards New Computational Principles for Robotics and Automation’. IEEE, 1997. p. 146-151.

[8] MORAVEC, H; ELFES, A. High resolution maps from wide angle sonar. In: Proceedings. 1985 IEEE international conference on robotics and automation. IEEE, 1985. p. 116-121.

[9] GUIDO, R. C. A tutorial review on entropy-based handcrafted feature extraction for information fusion. *Information Fusion*, v. 41, p. 161-175, 2018.

[10] BOSI, M.; GOLDBERG, R. E. Introduction to digital audio coding and standards. *Springer Science & Business Media*, 2002.

[11] QUIGLEY, M. ROS: an open-source Robot Operating System. *ICRA workshop on open source software*. 2009. p. 5.

[12] USLU, E. Frontier-based autonomous exploration algorithm implementation. *2015 23nd Signal Processing and Communications Applications Conference (SIU)*. IEEE, 2015. p. 1313-1316.

[13] QUIN, P. Approaches for efficiently detecting frontier cells in robotics exploration. *Frontiers in Robotics and AI*, v. 8, p. 616470, 2021.

[14] QIAO, W.; FANG, Z.; SI, B. A sampling-based multi-tree fusion algorithm for frontier detection. *International Journal of Advanced Robotic Systems*, v. 16, n. 4, p. 1729881419865427, 2019.

[15] VALIENTE, D. Improved omnidirectional odometry for a view-based mapping approach. *Sensors*, v. 17, n. 2, p. 325, 2017.

[16] DE SOUSA, F. L. M.; SILVA, M. C.; OLIVEIRA, R. A. R. Applying Edge AI towards Deep-learning-based Monocular Visual Odometry Model for Mobile Robotics. *ICEIS*, v. 1 . 2022. p. 561-568.

[17] ANINDYA, C.; PURWANTO, D.; RICOIDA, D. I. Development of Indonesian speech recognition with deep neural network for robotic command. *2019 International Seminar on Intelligent Technology and Its Applications (ISITIA)*. IEEE, 2019. p. 434-438.

[18] HE, W.; MOTLICEK, P.; ODOBEZ, J.-M. Deep neural networks for multiple speaker detection and localization. *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018. p. 74-79.

[19] GUIDO, R. C. A tutorial on signal energy and its applications. *Neurocomputing*, v. 179, p. 264-282, 2016.

[20] GUIDO, R. C. ZCR-aided Neurocomputing: a study with applications. *Knowledge-based Systems*, v. 105, p. 248-269, 2016.

[21] PySoundFile. url: https://pysoundfile.readthedocs.io/en/latest/ (accessed in 01/09/2022).

[22] CRISTIANINI, N. An introduction to support vector machines and other kernel-based learning methods. **Cambridge university press**, 2000.

[23] FAYED, H. A.; ATIYA, A. F. Speed up grid-search for parameter selection of support vector machines. *Applied Soft Computing*, v. 80, p. 202-210, 2019.

[24] BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. *Journal of machine learning research*, v. 13, n. 2, 2012.

[25] NGUYEN, V. Bayesian optimization for accelerating hyper-parameter tuning. In: *2019 IEEE second international conference on artificial intelligence and knowledge engineering (AIKE).* IEEE, 2019. p. 302-305.

[26] Joblib. url: https://joblib.readthedocs.io/en/latest/ (accessed in 16/10/2022).

[27] NumPy. url: https ://numpy.org/doc/stable/reference/generated/numpy.logspace.html (accessed in 16/10/2022).

[28] Scikit-learn. url: https://scikit-learn.org/stable/index.html (accessed in 16/10/2022).

[29] Suresh R Devasahayam. *Signals and systems in biomedical engineering: signal processing and physiological systems modeling*. **Springer Science & Business Media**, 2012.

[30] Danny Staple. *Learn Robotics Programming: Build and control AI-enabled autonomous robots using the Raspberry Pi and Python*. **Packt Publishing Ltd**, 2021.