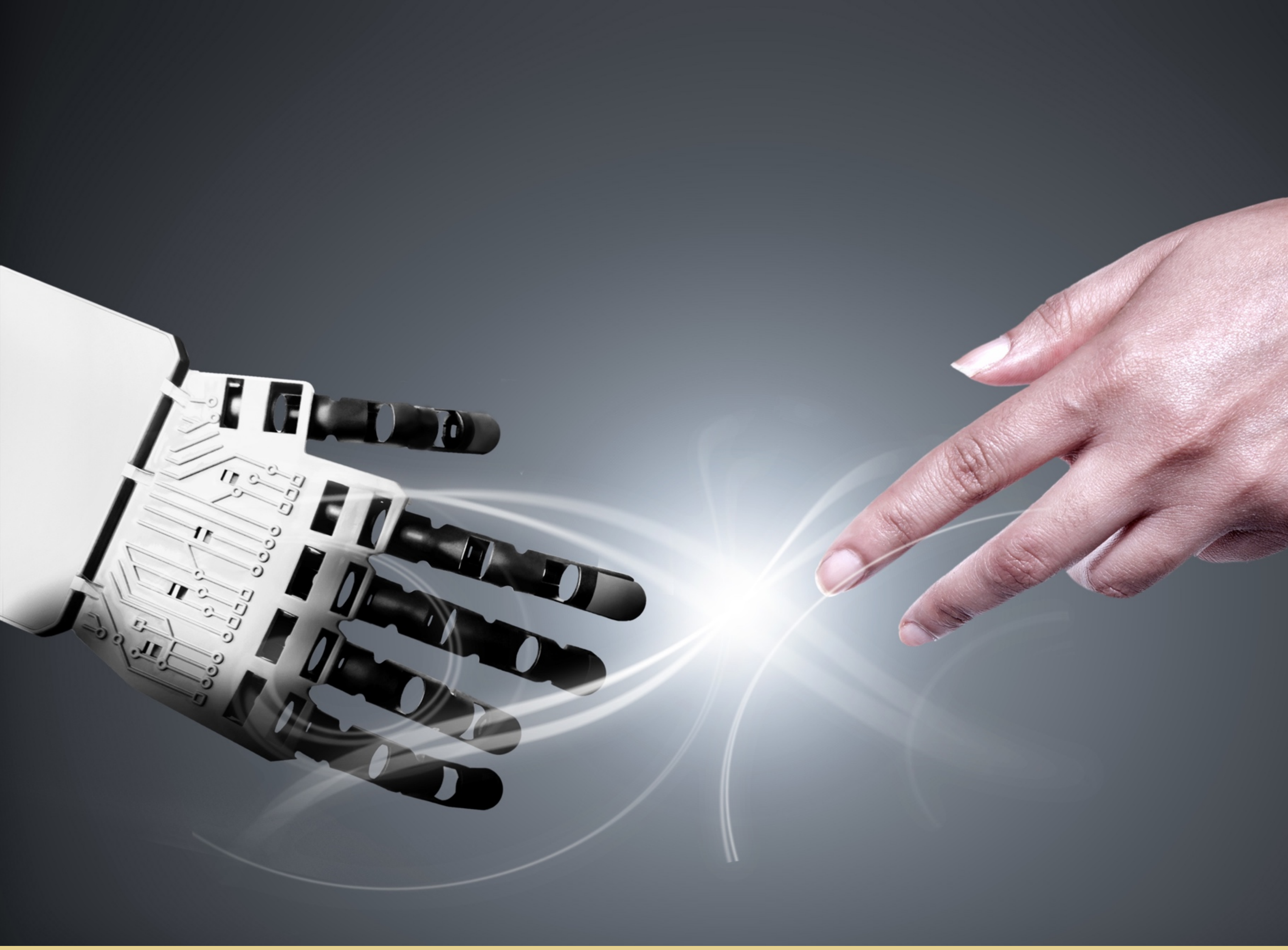


João Pedro Oliveira Santos Paulo

RobotShepherding – Teleoperation of a fleet of robots through a Smartphone

Thesis submitted in partial fulfillment of the requirements for the degree of
Master of Science in Electrical and Computer Engineering

July, 2017

· U C ·

Universidade de Coimbra

· U C ·

FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

RobotShepherding – Teleoperation of a fleet of robots through a Smartphone

João Pedro Oliveira Santos Paulo

Coimbra, July 2017

RobotShepherding – Teleoperation of a fleet of robots through a Smartphone

Supervisor:

Prof. Doutor Rui Paulo Pinto da Rocha

Jury:

Prof. Doutor Luís Alberto da Silva Cruz

Prof. Doutor Rui Pedro Duarte Cortesão

Prof. Doutor Rui Paulo Pinto da Rocha

Dissertation submitted in partial fulfillment for the degree of Master of Science in Electrical and Computer Engineering.

Coimbra, July 2017

Acknowledgements

To my supervisor Professor Rui Rocha, I would like to express my gratitude for all his availability to helping me, for his guidance and encouragement throughout this journey and for the dedication and commitment. It has been a pleasure to be part of this team.

To my friend Francisco Couceiro and my cousin Miguel Oliveira, a special thank you for your availability, advices and for helping me getting here.

To all the AP4ISR team, it has been a pleasure to share this time in the laboratory with you.

To those of you who took some of your time to test my application, without you, this would not be possible. Thank you.

To all the friends I have had the pleasure to meet throughout these five years in Coimbra, in particular Diogo Justo, Francisco Couceiro and Tiago Chaves, thank you for your opinions, suggestions, knowledge and friendship.

To my dear girlfriend Mariana Petrica, thank you for your support, your help, your love and thank you for believing in me.

To my sister Carolina, I thank you for being my inspiration and for your sweet, constant devotion.

To my parents, I thank you for everything. You made this possible.

Abstract

In tasks performed by mobile cooperative robots it is convenient to ensure that their movement follows a certain geometric pattern, for example, in situations in which a user wishes to command a group of robots in order to transport them from a location to another. To do so, a user commands the group as a whole focusing his/her attention on a single robot that is seen as the leader of the formation, following what is called formation control.

The results of a work developed last year, regarding this particular area, are the starting point of this thesis. However, it is now intended to explore the usefulness and benefits of using an Android-based smartphone to interact with the leader and consequently, the other robots of the group. The development of new interfaces that improve substantially the interaction is now the prime focus. It is expected that the user is able to control not only the movement but also the geometric pattern of the formation without having to stop the navigation process. The system should be protected against collisions when robots move towards detected obstacles and provide feedback to the user, through the smartphone, regarding these incidents.

The interfaces aim at exploring different interactions between the user and the smartphone, allowing the creation of a mobile application that can be adjusted to each user's preferences.

Keywords: Human-machine interaction; User-centered design; Android-based mobile platform; Obstacle avoidance; Formation control.

Resumo

Em tarefas realizadas por robôs móveis cooperativos é conveniente garantir que a movimentação dos robôs é feita segundo um determinado padrão geométrico, por exemplo, em situações em que um utilizador humano pretende comandar um grupo de robôs para transportá-los de um local para outro. Para tal, o utilizador comanda o grupo como um todo, focando a sua atenção num robô que assume o papel de líder da formação, adotando assim dinâmicas de controlo de formação.

A presente dissertação tem como base os resultados de um trabalho desenvolvido no âmbito desta área no passado ano letivo. No entanto, pretende-se agora explorar a utilização de um *smartphone* Android na interação que é feita com o robô líder e, consequentemente, com os restantes robôs da formação. Procura-se desenvolver novas interfaces que melhorem significativamente a interação do utilizador com a frota. Além de poder controlar o movimento, espera-se que o utilizador possa escolher o tipo de formação da frota a partir do *smartphone*, sem necessidade de interromper o movimento. O sistema deve ainda estar protegido contra possíveis colisões quando os robôs se movem em direção a obstáculos detetados, alertando o utilizador para esse efeito, através do *smartphone*.

As interfaces pretendem explorar diferentes interações entre o utilizador e o *smartphone*, permitindo a criação de uma aplicação móvel que se ajusta às preferências de cada utilizador.

Palavras-Chave: Interação-homem-máquina; Design focado no utilizador; Plataformas móveis Android; Evitação de obstáculos; Controlo de formação.

"In the twenty-first century, the robot will take the place which slave labor occupied in ancient civilization."

— Nikola Tesla

Contents

List of Acronyms

AP4ISR	Artificial Perception Team for Intelligent Systems and Robotics
ISR	Institute of Systems and Robotics
IM	Interaction Mode
HMI	Human-machine interaction
EEG	Electroencephalograph
BCI	Brain-computer interface
API	Application programming interface
GPS	Global Positioning System
RFCOMM	Radio frequency communication
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
IP	Internet Protocol
SDK	Software Development Kit
IDE	Integrated Development Environment
ROS	Robot Operating System
OS	Operating System
UI	User Interface
3D	3 Dimensional
UX	User Experience

OM	Operation Mode

List of Figures

List of Tables

1 Introduction

1.1 Context: Formation Control, Autonomy and Interfaces

The diversification of areas in which robots are used has motivated the search for new processes and approaches that allow increasing the performance in each specific application. It is becoming easier and easier to perform tasks without any human interaction by using autonomous robots that behave differently, depending on the environment in which they are used and the objective to which they were conceived. To grant autonomy to a robot means to endow it with decision making capabilities, in a similar way to what happens in the living world, by implementing artificial intelligence algorithms.

The technological advancements allow us to consistently accomplish more in robots' autonomy, but the perception that Man has of Nature has led researchers to look for new ways to enrich that autonomy, based on cooperation relations that are seen in animals of the same species. In fact, the cooperation and coordination mechanisms that are observed, for instance, in fish shoaling and schooling, in hunting packs of wolves or even in human cooperative tasks, such as team sports, are the basis of formation control that is applied to robots [1] [2] [3]. There are several advantages in using systems with multiple cooperative robots, increasing the chances of achieving the ultimate objective. In the particular case of mobile robots, it is possible to identify specific situations in which it is needed to ensure that the group of robots navigates according to a certain geometric pattern, for example when it is intended that a robots fleet heads towards a defined location.

The purpose of this work is to develop human-machine interfaces that allow an operator to control the motion of a group of robots in formation in an intuitive and simple way, making it behave as a whole, while the operator is essentially abstracted from the group's internal dynamics required to move in formation. To control the fleet, the operator uses one of the interfaces that are part of an application developed to Android smartphones, through which communications are done with one of the fleet's robots, assumed as the leader [3].

1.2 Motivation

Formation control in robots is becoming increasingly popular to perform tasks that are difficult, time consuming or that require several points of action.

The developments that are seen in today's robotics allow to accomplish tasks that would not be possible otherwise, like the exploration of hard-to-reach places caused by natural catastrophes such as earthquakes and hurricanes, or inhospitable environments, like nuclear disasters and planets ready for exploration. These are, generally, situations in which there is a special need to transport various robots to a certain location, and in which the logistics associated with that process are troubling for the final objective of the mission, which is to perform another designated activity and not really the transportation of the robots.

The usage of formation control in such cases allows expediting the process of transportation and, therefore, increase the overall useful time to perform the task. In fact, by treating the fleet as a whole, in the operator's perspective, he needs only to control a single entity, like if he was controlling just one robot. This kind of solution resembles the interaction of a shepherd who conducts an arbitrarily large flock of animals from one point to another.

Generally, robots motion control is done using specific devices, such as joysticks, but these may not be the most intuitive and easy to use. Having specific devices to do specific things is not always the best answer.

The rapid development that the smartphone industry has had in latter years has offered new possibilities based on the increasing complexity of the hardware and software involved, but also on the demands of consumers. Some particular aspects of these devices, including sensors like touch, accelerometers and gyroscopes, seem to be quite appropriate to develop interfaces between humans and robots that are, at the same time, easy and intuitive to use, and present serious advantages regarding signal reach and availability.

1.3 Objectives

The main objective of this dissertation is to develop an Android-based mobile application for smartphones that provides human-machine interfaces to be used with fleets of robots moving in formation. The idea behind this application is to ensure the efficient communication with the fleet's leader robot, giving it simple navigation instructions, which then influence the fleet's behavior, according to the desired geometric pattern, also selected by the user.

Regarding formation control, this thesis is built upon Daniel Marcelino's work developed recently at AP4ISR team of ISR - University of Coimbra [3], wherein a leader-follower formation control system was implemented using the attractor dynamics framework [4].

Thus, the following objectives can be highlighted:

- Further study the leader-follower approach to formation control in the sequel of Daniel Marcelino's M.Sc. dissertation [3];
- Survey human-machine interfaces developed and used throughout the years by the scientific community;
- Investigate human-machine interfaces that relate to smartphones, specifically, the ones using Wi-Fi;
- Learn about the Android operating system and the techniques and details behind application development;
- Design, implement and test an Android application that provides the user with different human-machine Interaction Modes to control a single robot;
- Extend this application to allow to teleoperate a flock of robots;
- Test the system, its functioning and validate the usability of the developed application in real environments with a relevant sample of users.

1.4 Outline of the document

The present dissertation is divided into six chapters. Chapter 1 begins by introducing the context and motivation of this work and outlines the objectives. Chapter 2 compiles the state of the art on HMI regarding smartphones, exploring important related researches. It then converges to the approach taken in this work. Chapter 3 covers the system and explains how the different parts work together to produce the wanted results. Chapter 4 focuses on the process of creating and designing IMs and presents the final solutions created. Chapter 5 presents results of the evaluation and usability validation of the IMs created, from the users' perspectives. Chapter 6 consists on the conclusion of this dissertation and points out future work directions.

2 Related Work

This chapter focuses on the state of the art that relates to human-machine interactions, more specifically, to the interactions that are somehow related with smartphones.

2.1 The Human-Machine Interaction

Human-machine interaction is what we call the set of interactions and communications that occur between humans and machines, done through a human-machine interface.

By machine it is intended any technical system that has its own dynamics, dictated by the joint actions of its automation components, related to supervision and control of the system, and the decision support entities [5].

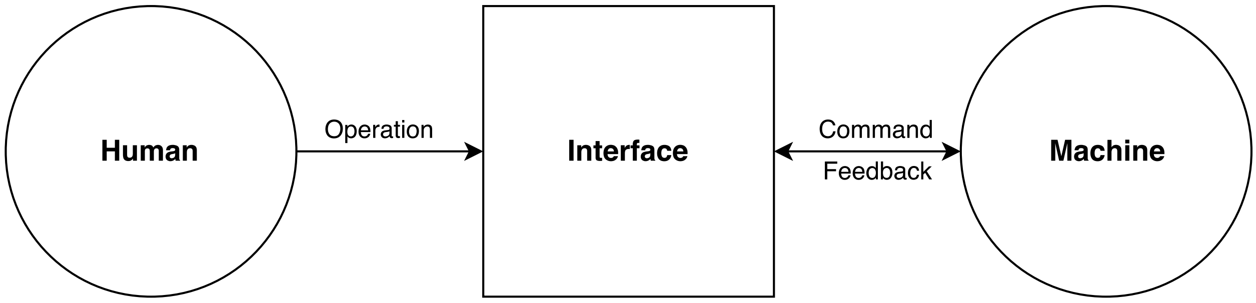


Figure 2.1: Human-Machine Interaction.

Robots are quite recent in the human history scale, but the idea of creating intelligence, not necessarily what we envision in today's intelligence definition, and performing difficult tasks without sacrificing human resources is much older. A little more than 50 years ago, research efforts on human-machine systems began [6] and with the rapidly developing technologies, started to vastly influence communities all over the world.

In fact, the interaction between men and machines is, in today's society, considered to be essential for the successful completion of any process [5].

Almost every technology-dependent activity is somehow related to interactions and communications with machines, even though some of those interactions are not easy to identify.

Regardless of today's need, it has been an iterative and dynamic process. It began with simple interactions that were generally separable from each other, mostly based on how to show information, how to control the machine and when to handle the information received [5]. This is the way traditional control was done throughout the years. It then started to incorporate more ambitious and complex features that contributed to build better working, more dynamic systems that privilege nicer user experiences and allow new tasks to be performed. With some of these advancements, operators tend to become supervisors rather than direct agents [7].

2.1.1 Human-Machine Interactions technologies

Many different kinds of technologies arose and revolutionized the traditional industrial machines [8] [9]:

- **Optic-based** - one of the most researched HMI areas, it is related to light acquisition and processing and it is the basis of Computer Vision. It is widely used for objects and features detection using cameras, can also be used for gesture recognition and is still one of the most promising technologies. Tornow, Al-Hamadi and Borrmann [10] use gesture and hand postures to interact with a team of mobile robots. Boehme, Brakensiek, Braumann, Krabbes and Gross [11] presented a neural architecture for gesture-based interactions between users and mobile robots, which can easily be extended for other applications;

- **Acoustic-based** - focuses on capturing and processing audio, mainly used for speech recognition. It has many applications, such as machine manipulation and communication, text conversion, voice recognition, etc. Parlitz, Baum, Reiser and Hägele [12] make use of speech input and output capabilities as part of a more complex system, designed as a household robot companion;

- **Bionic** - probably the one that best describes and relates to science fiction, bionic technology and the possibilities it provides have been fuel for creativity on the film and book industries for many years. In latter years, though, many of the imagined applications started to turn into reality. This area is a combination of several other scientific areas, such as biology, robotics and computer science and consists on any technology that makes use of biological signals to perform specific functions. Stephygraph, Arunkumar and Venkatraman [13] use EEG technologies to communicate with mobile robots so people that suffer from motor disabilities can interact with them, using a BCI;

- **Tactile** - it is the only technology that requires humans to touch a device to interact with the machine. As the name suggests, it involves every technology that relates to touching. It can be the simplest button, used on traditional interactions and still widely used, but it can also be the well-known keyboards, etc. The tactile principles are actually combined with vision-based technologies in modern devices to create touch screens that require both light and pressure capabilities. Recent smartphone-based HMIs hugely depend on this technology;

- **Motion** - this kind of technology involves anything that can detect motion. It comprises gyroscopes and accelerometers which are normally combined to provide reliable information on motion. Gyroscopes and accelerometers are pretty much in almost every portable device that is used nowadays and have proven to be quite important and present many different applications.

Although it is possible to divide these technologies according to the principles and areas to which they relate the best, most of the modern solutions combine two or more technologies to improve reliability and expand the possibilities of interactions. For instance, every solution that makes use of touch screens depends on both optic-based and tactile-based technologies. Another example is non-traditional teleoperation, where an operator has to control a machine without being physically in the same room as the machine. Although the user might be sending commands through a touch-based controller, this clearly needs some kind of feedback, most likely visual (optic-based). The same happens with every solution that is based on smartphones, which are themselves a combination of different kinds of sensors and technologies.

It is important to notice that smartphones, due to their outstanding versatility, offer almost all of these technologies. In fact, it is fair to say that perhaps only the bionic technology is not yet available in today's smartphones.

2.2 The smartphone and Android OS

The last 15 years have resulted in immense advancements in computing power and portability, which means that there are many more machines and robots available with much more autonomy and a lot more features, that fit different purposes and need different kinds of handling. That is the reason why HMIs have to keep up with these developments and guarantee that everything works accordingly. This need to create more and better interactions makes it reasonable to create specific solutions to specific problems, which is often the case: a designated application has its own human-machine interface, adequate to fulfill the final objective.

Regardless of the benefits that customized solutions offer to particular problems, sometimes there is no need to create something completely dedicated or it is a waste of resources and time to develop a specific solution. Or sometimes it is just much more appropriate to have a device that is able to be configured and have several different interfaces, depending on the objective or user's preferences.

The industry of portable devices is quite recent due to the requirements in equipments and technology. The smartphone industry, in particular, is even more recent. In fact, a few years ago mobile phones were no more than portable telephone devices used to have conversations with a little more mobility. But the fact is that the exponentially growing demand combined with the amazing discoveries and inventions in technology, started to improve dramatically the capabilities, features and computational power of these small, pocket devices. So much that they are no longer simple mobile phones but smartphones, built with state of the art technology and software, including dedicated Operating Systems [14]. And the best part is that these devices easily fit the size of a hand and are lightweight, have reasonable battery life and most importantly, are cheaper than other devices [15].

Recent smartphones are often equipped with several important sensors such as GPS, Wi-Fi and Bluetooth communication adapters, accelerometers, gyroscopes, microphones, cameras, etc [16] [6]. Perhaps even more important, application development for certain Operating Systems like Android is free, open and presents a large community of developers. Furthermore, the nature and philosophy behind Android allows developers to have almost full control of what happens, which makes it the perfect tool to create configurable human-machine interfaces. Android is mostly based on Java language, which provides simple APIs to developers and is useful to create state of the art applications [17].

Having this large variety of sensors and modules, when considering HMI, smartphones allow diverse approaches when it comes to data acquisition, feedback to the user and communication architectures.

2.3 Communication

When using smartphones as a key element of Human Machine Interface, there are two kinds of wireless technologies that come to mind: Wi-Fi and Bluetooth. To get the most out of the capabilities of a smartphone, wired communication is often out of the equation. In fact, literature tends to focus on Wi-Fi and Bluetooth technologies when dealing with communications between Android smartphones and robots.

2.3.1 Bluetooth

Bluetooth technology has been associated with phones and smartphones for years, essentially for data exchange between devices. It was created as a wireless alternative to data cables and is based on Ultra High Frequency radio waves transmissions. It is widely used because it allows direct connections between devices that have Bluetooth modules without the need for an established infrastructure and it is quite affordable compared to other options. Is also provides reliable data transfer with low power consumption, something that seems to fit the interests of communication modules in smartphones.

One key characteristic of this technology is that it was designed to be used in short-range distances, with few versions allowing more than a few meters communication range. Nonetheless, for most portable devices and applications, it lays within the intended range, making it a powerful communication technology [18].

Furthermore, Android offers a set of packages and interfaces that make it easier to implement Bluetooth functionalities. Zhang, Wang, Yang, Fu and Wang [15] rely on Bluetooth's RFCOMM channel to establish connections between an interactive interface and the motion control system of a hand rehabilitation robot. Mohan, Tripathi and Gangadharan [19] use a smartphone and a computer, both equipped with Bluetooth adapters, to transmit smartphone's relevant data to a LabView application. Mester [20] refers a wireless sensor network to control mobile robots within a laboratory, in which uses both the Web and Bluetooth. The Web allows users to be anywhere in world and still be able to see what is happening in real time, as well as send control commands to the robots. On the other hand, the Bluetooth technology is used to transmit those control commands from a computer module placed in the same area as the receiving end of these commands, to the robots, in order to avoid unnecessary waste of battery energy as Bluetooth modules have a lower energy consumption. Nádvorník and Smutný [14] chose Bluetooth technology in their Android-based remote controller for mobile robots, because the distance reach was sufficient and it provided better battery life to the smartphone when compared to Wi-Fi. Santos [21] also relied on Bluetooth in his mobile robot navigation system, due to implementation decisions regarding the middleware that he developed to this specific situation. One important decision factor to use Bluetooth might have been the highly computationally expensive algorithms that were executed in the smartphone and drain a lot of battery energy.

2.3.2 Wi-Fi

Wi-Fi (Wireless Fidelity) [22] is nowadays widely spread and it is present in almost every device, from smartphones, to computers and smartwatches, etc. The idea behind Wi-Fi is deeply related to the prospective concept of creating wireless local area networks. Wi-Fi allows users to surf the Web at very high speeds, close to a typical cable broadband. This technology is used both through access points connections and ad hoc. Although ad hoc connections are easier to implement and eliminate the need for third party elements, they tend to be limited in distance range, when compared with access point connections [16] [23].

A downside of this technology, when compared to others, is that it has a higher energy consumption. However, the improvements in Wi-Fi modules and protocols as well as the advances in smartphones hardware and software, have lately enabled people to use Wi-Fi almost as a always-on feature, without worrying too much about battery life.

Within the Wi-Fi realm, there are two main transport protocols that can be considered: UDP and TCP. The first is connectionless and without guaranties of data delivery, although interesting in real time applications. The second one is more complex but with a higher reliability and with data delivery guaranties. Lu, Liu, Wang and Sun [24] relied on TCP, ad hoc configuration Wi-Fi communication between a smartphone and a robot. One key point in their decision to use TCP over UDP is the fact that the integrity of the data associated with the communication is much more important than the real-time performance. The ad hoc configuration seems to fit their intentions of creating point-to-point connections between the robot and the smartphone. Parga, Li and Yu [8] designed a Wi-Fi communication HMI system that uses an Android smartphone. They used Wi-Fi TCP access point connections because the equipment necessary to establish the network is cheap, easy to use and delivers with high transfer speeds and due to the standards, there is no need to be too careful about the details. Also, a communication of this type is transparent to the user. Bansode and Singh [6] also relied on Wi-Fi TCP connections on their work. They designed a system to control a robot through a smartphone and used Wi-Fi to exchange data between the robot and the smartphone. Cruz, Agnese, et al. [16] used Wi-Fi, TCP connections because of the type of robot that they used (Wifibot). They worked on making sure that the user could establish both ad hoc and access point connections. The approach to use Wi-Fi is related to the fact that the user can keep a larger distance from the robot.

2.3.3 Comparison of Bluetooth and Wi-Fi

Based on a literature review [6] [8] [14] [15] [16] [18] [19] [20] [22] [23] [24], Table 2.1 presents a comparison of Bluetooth and Wi-Fi along four essential features. The main characteristics of each of them were taken into account and an evaluation was assigned to each feature.

Table 2.1: Comparison of Bluetooth and Wi-Fi Technologies.

Technology	**Speed**	**Power Consumption**	**Max Distance Range**	**Overall Implementation**
Bluetooth		Lower		
Wi-Fi	Higher		Higher	Easier

From what was analyzed, Wi-Fi connections seem to be beneficial and well engaged on what is expected to be achieved with this thesis. The easiness to implement, the support that the Android community offers to the usage of the Wi-Fi API, the distance range it offers, the broadband speeds that can be reached as well as the reliability of this technology are all strong points to take into consideration. Furthermore, Wi-Fi networks are becoming increasingly ubiquitous, there is no need for a pairing process in each session and it is useful to guarantee that two devices are connected to a SSID, even if not necessarily the same. All these reasons add up to make Wi-Fi the candidate for handling communications.

2.4 Summary

This chapter consisted on the relevant works that are part of the community regarding smartphones and robots systems, for building Human-Machine Interactions. The types of sensors that smartphones have these days offer several points of interaction and allow the creation of many different interfaces.

That being said and taking into consideration the ultimate objective and purpose of this work, there are three kinds of HMI technologies that will most certainly be the basis of the interfaces that are going to be created: optic, tactile and motion. Optic because of the visual feedback; tactile, as it presents a good starting point to control a robot (touchscreen); motion, as smartphones' sensors are quite relevant to control robots and enable new types of interactions. Acoustic is also a possibility in smartphone's HMI. However, the field behind this technology is large enough to be studied alone. Bionic is, for the time being, a technology that is not yet available or explored within smartphones.

3 The System

This chapter describes the developed system, its architecture and functioning, providing some details on each of the specific key parts that make up the system. Firstly, a brief explanation of how the system works as a whole is given. The focus is then turned to the implemented ROS modules that support the solution and finally the structure and technical aspects of the Android application are covered.

3.1 System Overview

Taking on the results of the State of the Art review, the objectives of this thesis make it worth considering dividing the system into three main elements, as represented in Fig. 3.1.

Two of these three elements constitute the core logical components, producing and interpreting data to generate viable commands or feedback signals. They are responsible for the integrity of the system and ensure that data flows correctly throughout the duration of each session. To do so, they rely on a communication module. The three elements can be identified as the following:

- The Android **smartphone**, which directly reflects the user's intentions;
- The **robots**;
- The **mean of communication** between these two parts, in this case, a TCP/IP socket Wi-Fi connection.

It is important to keep in mind that the interaction between the user and the group of robots is simplified in the user's perspective. That is because the user interacts directly only with the robot that is the leader of the formation. The indirect interaction between the user and the rest of the robots is sustained by the collective behavior of the group. In fact, the group ensures the collective behavior of moving in formation by following the leader that is directly navigated by the user of the application.

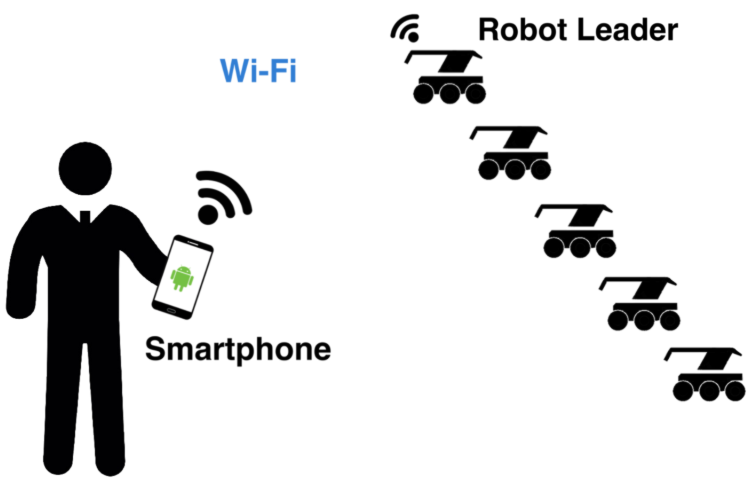


Figure 3.1: High level representation of the system: the smartphone on the left; the fleet of robots, more specifically, the leader being the first of the formation; the Wi-Fi communication.

3.2 System Architecture

On a more formal perspective, the architecture of the developed system, illustrated in Fig. 3.2, shows how these components are related and gives some insights regarding the dynamics of the system.

3.2.1 Android OS smartphone

On the left side of Fig. 3.2, it is possible to identify the Android-based smartphone. The Android OS is worth mentioning because it provides the needed software layer to create applications by giving the programmer the abstraction necessary to produce quality software without worrying too much about how things are processed internally.

Android is an OS created by Google to be incorporated in mobile devices [25]. Throughout the years, it has had an increasingly large community of developers and collaborators that design Android-specific applications, making it the most widely used OS. The reasons behind this success are related to the fact that is it open-source and application development is free of charge and also to the fact that Google promotes and supports this platform [17]. The official development languages are Java [26] and Kotlin [27] as of July 2017, both of them running on the Java Virtual Machine [28].

For the purpose of this work, the Java programming language was used because it has been the long term solution for Android developers until the recent admission of Kotlin.

The Android SDK [29] is the standard set of development tools to use when developing

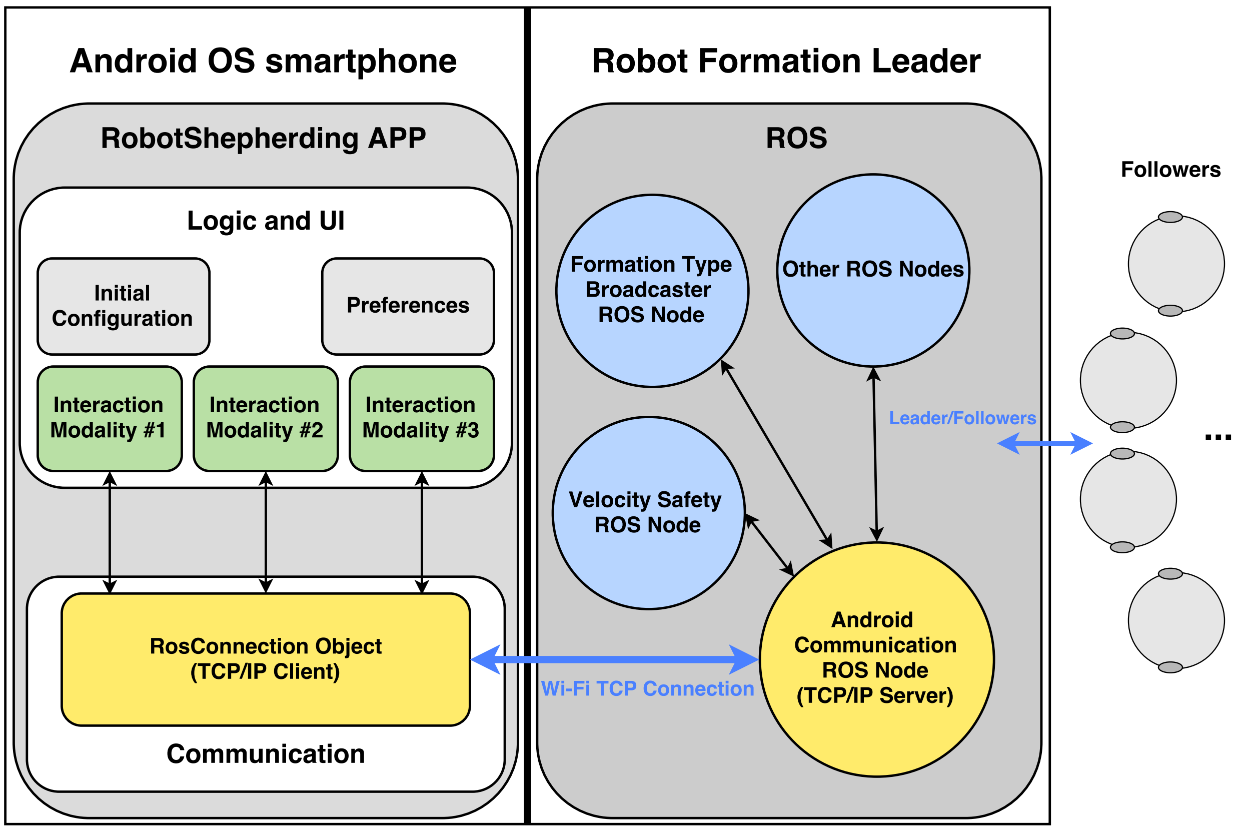


Figure 3.2: Architecture of the system.

applications. It incorporates both development and debugging tools, including Android emulators, that allow a much more fluid experience for the programmer. The current official IDE is Android Studio [30], made by Google and powered by IntelliJ IDEA [31], and was the one chosen to develop the application.

One important Android notion to keep in mind, as it will be referred to quite often in this document, is the *Activity*. An Activity in Android corresponds to a focused thing the user can do and almost always reflects some kind of interaction. It is fair to establish in the context of this work that each Activity reflects a different screen.

3.2.2 *RobotShepherding* Application

The designed Android application is called *RobotShepherding*, relating to the metaphoric idea that a user will be controlling a group of robots like a shepherd controls the behavior of a herd.

By using the application, the user is able to not only control the motion of the group, but also to change the movement pattern between two different formations: column and oblique. Because of a safety layer that was created to this specific case, the navigation of the leader is safe from collisions against obstacles and the user is presented with haptic and/or visual feedback in the presence of such obstacles.

The application tries to take advantage of the Android smartphones' capabilities, offering three different types of interaction modes, whose details will be covered in the next chapter.

To establish a coherent and modular hierarchy, the elements within the Android application were divided into two groups: **Logic and UI** and **Communication**.

Logic and UI

This group (Fig. 3.3) is the main part of the application as it comprises the visual design along with interactive features and also the logic behind what the user is able to perceive. The final application is composed of five different *Activities*, thus reflecting five distinct screens, each of them having its own logical entities to handle user inputs and provide the appropriate feedback, depending on the screen.

Each one of the five *Activities* has its own particular purpose, although three types of *Activities* can be identified:

- **Initial Configuration Activity** - this *Activity* corresponds to the first screen that is shown to the user when the application is first launched after installation. This is the first contact that the user has with the application and is where the predefined connection data (regarding the socket destination port and IP address of the robot to which the application is connecting with) is inserted and saved, as well as other important information, like the preferred formation type and interaction mode type (the user chooses from the ones available).

- **Preferences Activity** - deeply connected to the first mentioned *Activity*, the *Preferences Activity* is the screen to which the user must navigate to in order to change the predefined and preferred settings of the application, such as the previously mentioned items, but also other relevant aspects that enhance the interaction. For instance, the user is able to activate/deactivate haptic feedback and choose between three different types of Operation Modes that result in distinct navigation velocities, to fulfill certain expectations, depending on the mission the user aims to accomplish.

- **Interaction Modes Activities** - certainly the most important part of this work, these *Activities* are the final result of the three distinctive interaction modes that were designed and are the key elements for the user to interact with the robots. Because they reflect quite divergent types of interaction, they constitute three separate screens, having specific logic and feedback, as will be covered in the next chapter that will explain the details of each of the modes. One key thing to notice is that every single interaction mode relies on the same type of object in order to send the relevant commands to the robot, which is part of the other group of the application - the *Communication*.

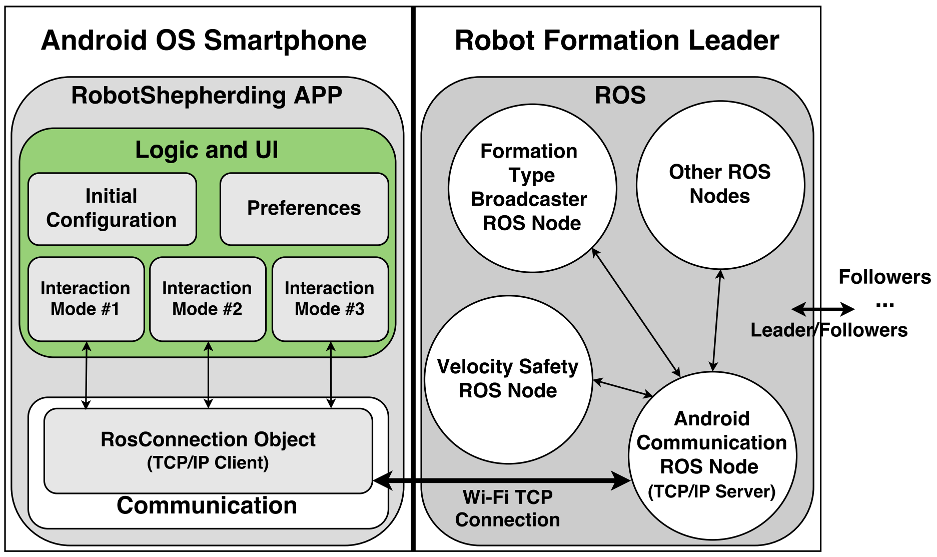


Figure 3.3: Architecture focused on the *Logic and UI* component.

Communication

The other group worth considering inside the developed application is the *Communication* (Fig. 3.4). This part of the application is mainly composed of a class of objects called *RosCommunication* that is responsible for handling the way the smartphone and the robot communicate. This object has a particular set of attributes and methods that allow to send information from one ending point to the other, implementing a TCP/IP client. The way this is done will be covered later in the document.

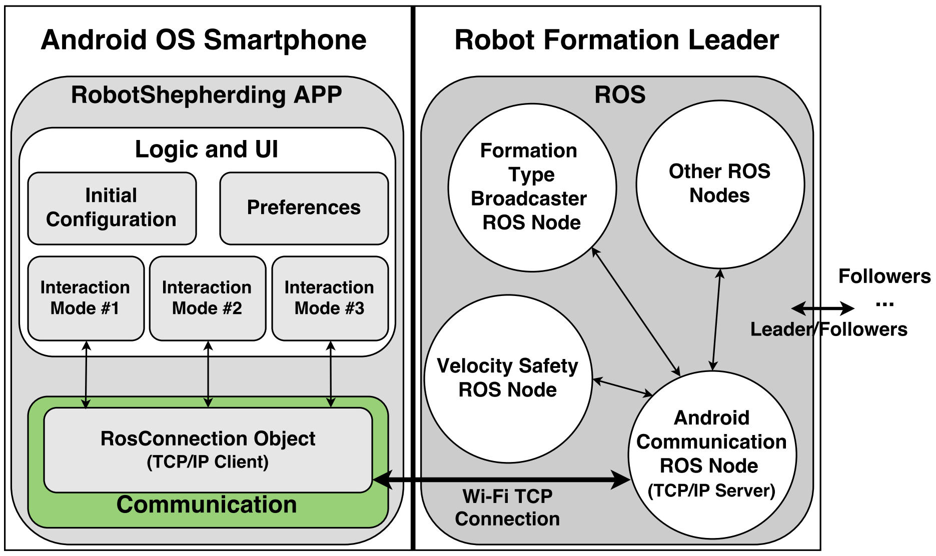


Figure 3.4: Architecture focused on the *Communication* component.

3.2.3 Robot Formation Leader

As it is probably clear by now, the other ending point of the communication is the robot, either the user is operating it alone or within a formation, in which case it is the leader of the formation (in this particular type of formation control). The robot can be, theoretically, any kind of mobile robot that incorporates the necessary software to fully understand the data coming from the smartphone. To do so, the robot should be running ROS with the appropriate ROS entities, those being the ones related to the robot itself and its drivers, but also the ones created for this specific application to work.

3.2.4 ROS

ROS stands for Robot Operating System and constitutes a set of open-source libraries and utilities that are widely used worldwide for robotics and robotic applications [32]. It provides the basis to build solid and robust systems enabling distributed, modular designs. It has a great community that keeps growing and contributes to keep it up-to-date with recent technologies. The main ROS client libraries are Python and C++, although there is current work on developing Java (*rosjava*) and JavaScript (*roslibjs*) versions to meet the expectancies of programmers and match as many platforms as possible [33].

In the scope of this work, all the developed software regarding ROS was written in C/C++ because of the familiarity and availability of important documentation.

The *rosjava* client library seemed to be a solid candidate to use, because it could be incorporated with Android, thus enabling running ROS on the smartphone, but the decision was to keep unnecessary computational work away from Android, to prevent undesired battery life wastes. That being said, Android prepares and sends data to the robot, which will then treat it accordingly.

Important ROS notions and concepts

There are some notions and concepts that are part of the ROS nomenclature and are referred throughout this document:

- **Node** - a node is a ROS entity which is responsible for performing some kind of computation. The ROS design and modularity is assured by these singular entities. The nodes communicate between themselves by using topics, services or parameter servers [34].
- **Topic** - a topic is a named bus through which nodes exchange messages. For this to work, a node can publish or subscribe to a topic, respectively meaning that it can publish messages to a specific topic or gather messages that other nodes are publishing [35].
- **Message** - a message is a simple data structure that will hold the information that a determined node wants to publish to a topic or subscribe from. There are several primitive message types and also other types that are composed of primitive ones [36].

Although there are other ROS entities, these are the relevant ones to notice in this work, as they are going to be referenced when the ROS nodes created to fulfill the objectives of this thesis will be presented.

Android Communication ROS Node

The *Android Communication ROS Node* (Fig. 3.5) was created in order to support the communication between the Android system and the robot that runs this node. It is one of the most important elements to highlight because it is the other ending point of the TCP/IP connection between the smartphone and the robot. This node is responsible for establishing and maintaining the connection between the two devices, receiving the relevant data and publishing appropriate messages to dedicated topics. To do so, it basically behaves like a TCP/IP server, waiting for the smartphone to connect, then parsing the commands into data structures that can be interpreted as ROS types and published into the correct topics.

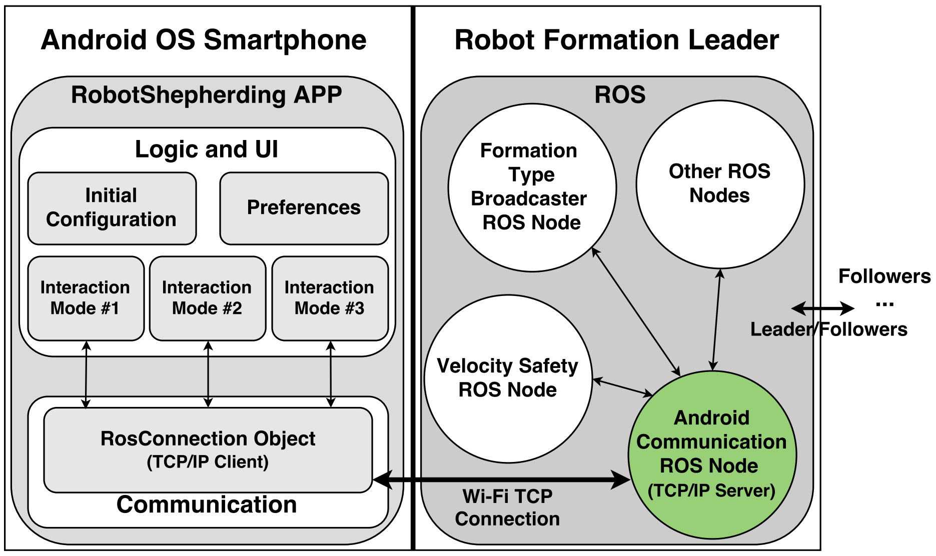


Figure 3.5: Architecture focused on the *Android Communication ROS Node*.

Velocity Safety ROS Node

To make sure that the robot has a safety layer when it comes to collisions, a ROS node was created to evaluate the data coming from the sonar sensors of the robot and this information can be used to verify if the position of the robot in each moment in time is in conflict with a safety distance that can be configured. If that is the case, then the robot will not be able to move any further in the direction that is causing the conflict.

For this mechanism to work, this node has to subscribe the velocity commands coming from the smartphone, that have already been received and published by the *Android Communication ROS Node*. If the velocity in question violates the safety line, then the command is not accepted and the robot shall not move.

As this node has access to the safety distance parameters and is able to check if the robot has crossed them, it has the power to provide that information to other parts of the system. This is actually done by publishing to a topic called *obstacle_ distance*, the distance values that are detected when the robot crosses the safety line. This information can then be used, for instance, to provide feedback back to the user.

A focused representation of the architecture is visible in Fig. 3.6 for better perception of the referred node.

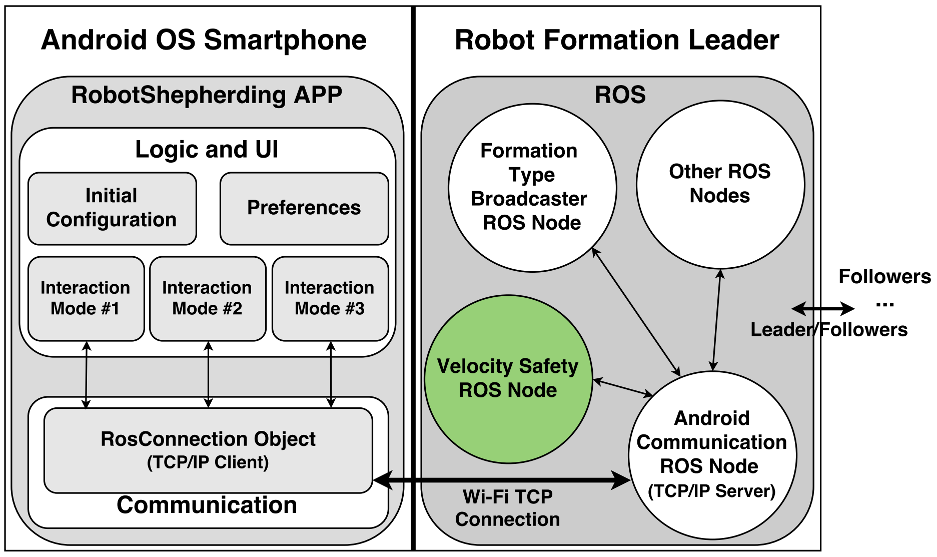


Figure 3.6: Architecture focused on the *Velocity Safety ROS Node*.

Formation Type Broadcaster ROS Node

When considering this type of formation control (leader-follower approach), the robots do not communicate directly with each other, meaning that there is actually no exchange of messages between them. In fact, the navigation of the group is based on the behavior of the leader. Furthermore, the motion of the leader will influence the motion of the other robots in a sequential way: the second robot of the formation follows the leader by identifying a valid marker on the back of the leader, the third robot will follow the second robot using the same principle and so on.

By behaving this way, there seems to be a challenge in making sure that the group can actually change the formation in real time, whenever the user wants to do so through the smartphone because there is no direct way to deliver that information to all the robots of the group (because only the leader is responsible for communicating with Android).

To solve this particular problem, two new nodes were created: one that is able to **broadcast** a signal that tells the robots to change the formation (Fig. 3.7); and other node that is able to **interpret** that information and actually execute that task. This was done by implementing yet again a TCP/IP server in the broadcast node, which will run on the robot leader so that the followers only have to know the IP address of the leader. This means that the followers will have to run a node that implements a TCP/IP client connected to the leader, thus receiving the formation type information whenever it is necessary to change the formation. With this solution, there is no need to establish complex connections and the system is kept simple and efficient.

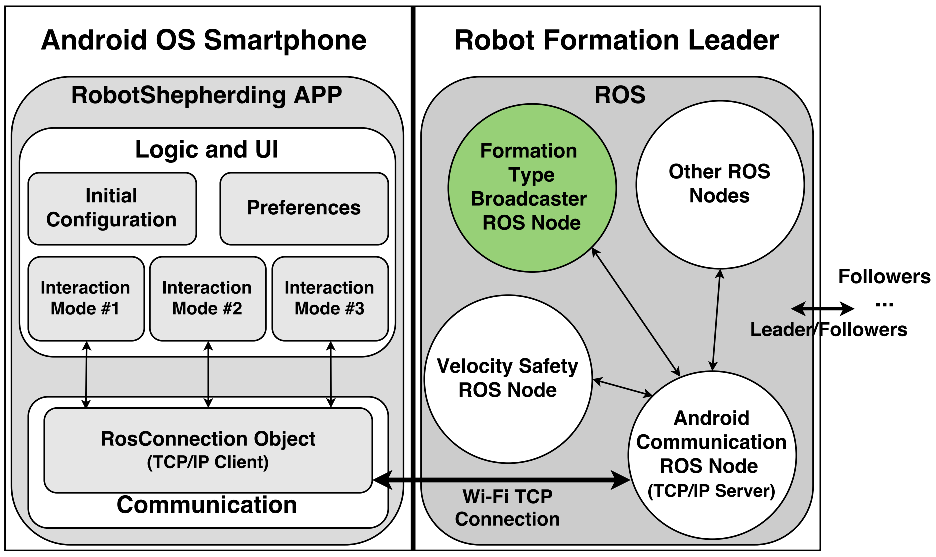


Figure 3.7: Architecture focused on the *Formation Type Broadcaster ROS Node*.

Other ROS Nodes

As mentioned when introducing ROS, to run and operate a robot, there are other nodes that are necessary for the integrity of the system and for the robot to correctly interpret the information that is being fed to it (Fig. 3.8).

This results in a need to run several structural nodes that constitute the robot's drivers, which, despite being fundamental for the system to work, are not the aim of this thesis. In order for the formation control to work properly, a node developed by Daniel Marcelino [3] was altered so that the formation could be changed in real time. Yet again, this node has to be running in each of the followers for the system to function.

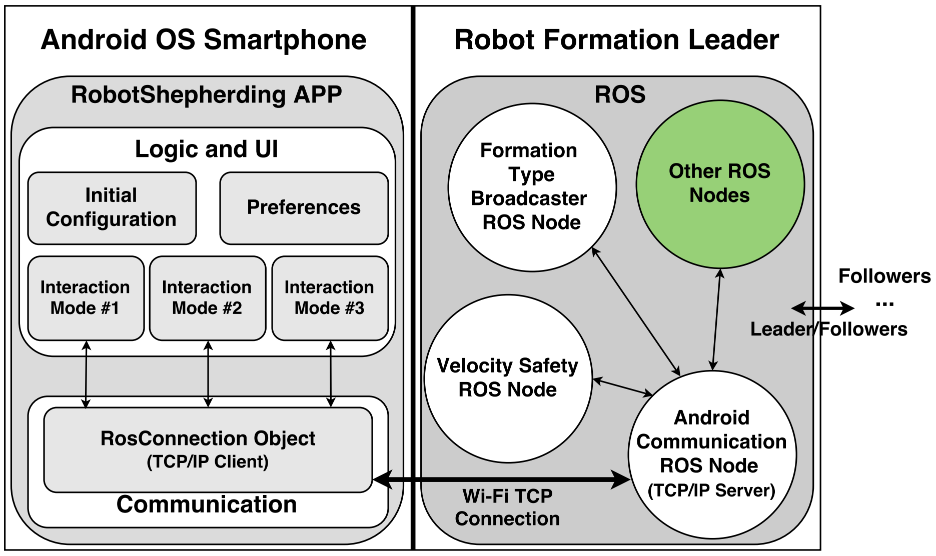


Figure 3.8: Architecture focused on the *Other ROS Nodes*.

3.3 Communication Details

After reviewing some of the most important elements inside the developed system, it is relevant to analyze a specific part of the architecture, focusing on how the elements connect and work together to perform the tasks they were designed to do. In this matter, this section covers the relationship between the *Interaction Modes Activities* (page 15), the *RosConnection* Object (of the Communication group of the Android application) (page 16) and the *Android Connection ROS Node* (page 18).

Figure 3.9 shows a representation of the relationship between the referred elements.

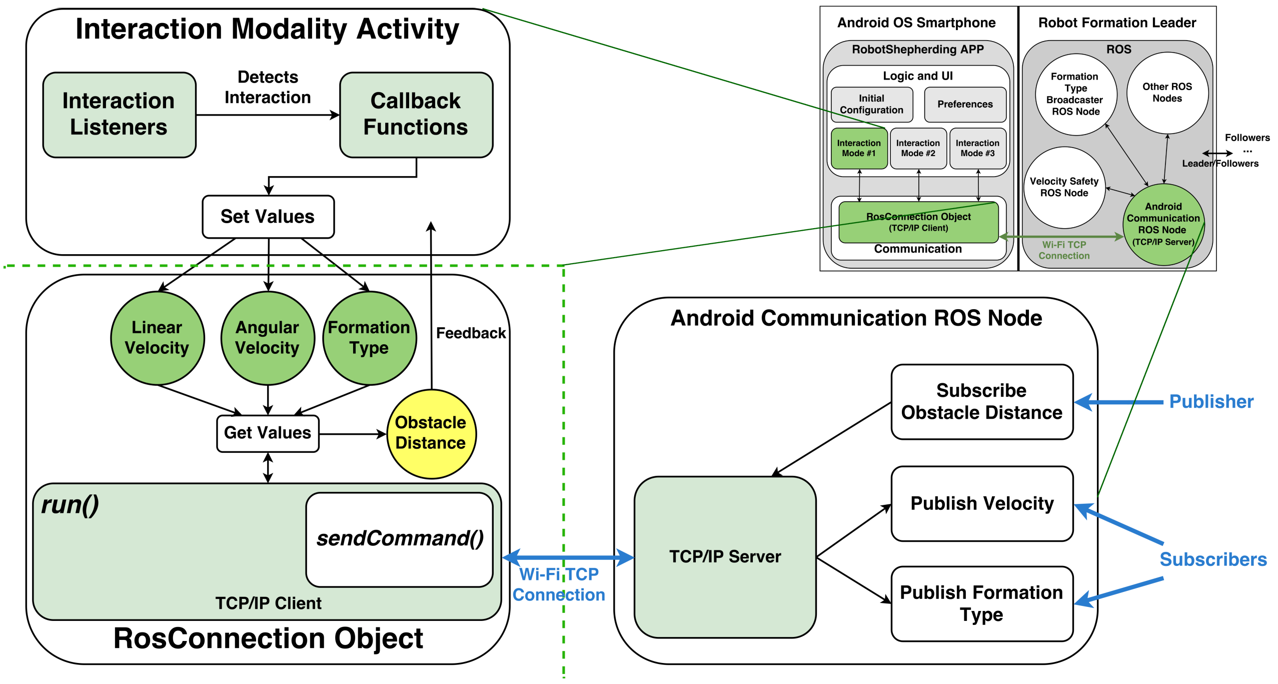


Figure 3.9: Representation of the relationship between each Interaction Modes Activity, the *RosConnection* Object and the Android Connection ROS Node.

In each and every of the IMs Activities that were designed, the same principles are applied regarding the way data is acquired and distributed, despite the differences in terms of the way users interact with the application, which is something that will be covered in the next chapter. So, the logical steps that happen behind the scenes whenever the application is running can be described as the following, keeping in line with what is showed in figure 3.9:

- The current IM Activity is consistently running listeners that wait for user inputs/interactions, depending on the type of interaction that is associated with the Activity.
- Whenever a registered interaction is detected, there is a callback function that is called to handle the input data from the user. That data can be related to the navigation

of the robot or changing the formation. The callback function will then process that information in order to turn it into viable information to be used as Linear and Angular velocity, if it is related to navigation or the identifier of the formation, when the interaction had the purpose to change that. In any of these cases, the callback function with lastly use the acquired and processed data to set the corresponding values of *Linear Velocity*, *Angular Velocity* and *Formation Type* of the *RosConnection* Object that is instantiated in the Activity.

- The *RosConnection* Object will then be running a loop in which checks and gathers its own values of Linear and Angular Velocity, as well as the Formation Type (already with the updated information from the user inputs), preparing a *String* command to send in each iteration at the end of the process, by using the *sendCommand* function. This process is done with validation and keeping it always updated with the Activity, so that no duplicate commands are sent to the robot. As it may be clear by now, this object implements a TCP/IP client and only runs when there is an active connection through the socket.

- On the other ending point of the established connection, there is a ROS node which implements the server-side of the TCP connection (Android Communication ROS Node). This node is crucial on the system's part that runs on the robot, because it maintains the connection to the other side and is responsible for receiving the data necessary to operate the robot and change the formation whenever wanted by the user. What happens here is that the server keeps receiving the commands sent by Android and decodes the messages it receives into three different parts: the Linear Velocity, the Angular Velocity and the Formation Type. This information will then be ready to create messages to publish into the appropriate topics. These messages will consist on velocity commands, relying on the ROS *geometry_msgs Twist Message*, composed of two 3D vectors (related to the linear and angular velocity); and on the formation type, relying on the ROS *std_msgs Int8 Message*. Finally, after the messages are set and ready, they are published: the velocity message is publish to the *unsafe_vel* topic, so that it can be validated by another node (Velocity Safety ROS Node) that aims to avoid collisions; the formation type is published to the *leader_formation_type* topic, which is subscribed by the Formation Type Broadcaster ROS Node, that is responsible for broadcasting the information to the other robots, whenever there is a change in formation. The Velocity Safety ROS Node has a very important role in the integrity of the system because of the validation of velocity that it does. When an obstacle is detected, meaning the minimum acceptable distance has been surpassed, this information can be fed back to the Android Communication ROS Node in order to be sent back to the Android smartphone so that the user can have some kind of feedback other than the direct visual observation of the robot. To do so, this node publishes those distance values into the *obstacle_distance* topic, which in turn is subscribed by the Android Communication ROS Node. Figure 3.10 shows a diagram of the topics that are somehow related with the Android Communication ROS Node.

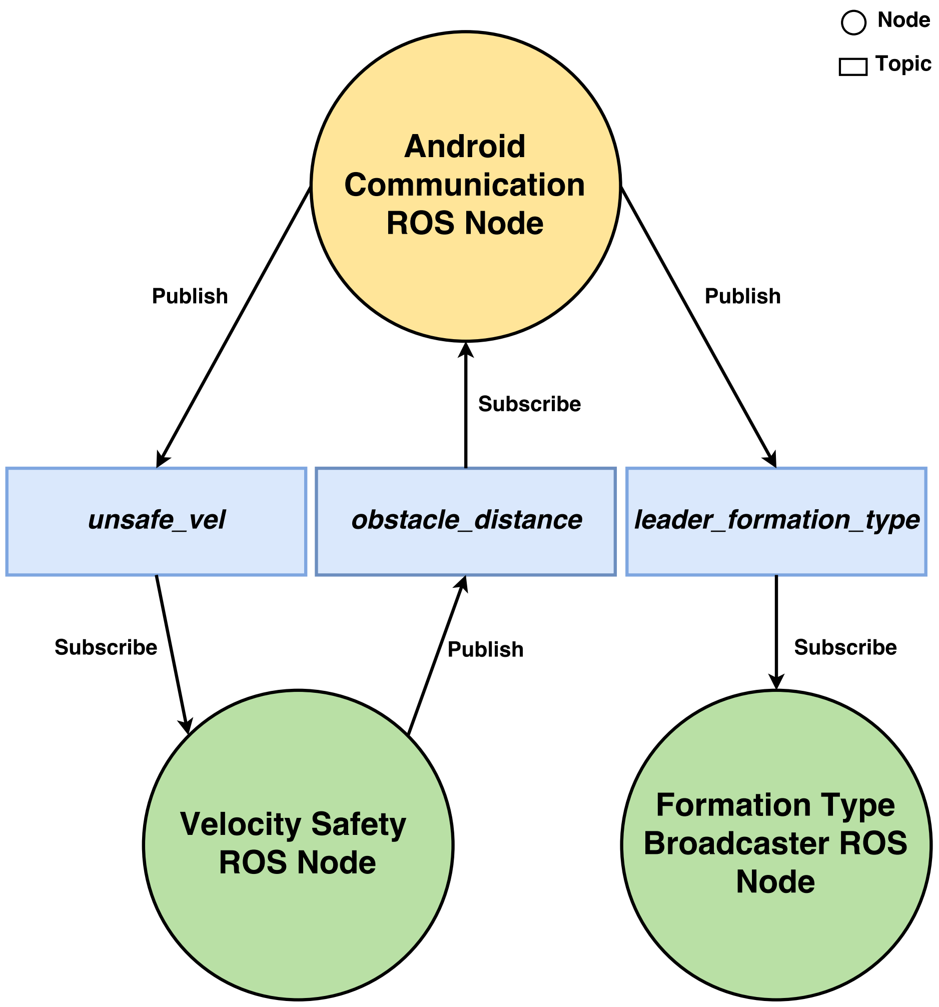


Figure 3.10: Diagram that represents the topics published and subscribed by the Android Communication ROS Node.

This is how the communication between the two independent, separate parts of the system (the smartphone and the robot) is set.

As previously mentioned, every IM relies on a *RosConnection* Object to be able to send information to the robot. Due to the fact that this application is designed to privilege user interaction, it is imperative that user actions are not affected by computation, data processing or communications. Therefore, the *RosConnection* Object implements a Java *Runnable* [37], which is nothing more than an interface that runs the object on a separate thread. With this approach, the *RosConnection* is always available and never blocks the user interaction with the smartphone.

3.4 Summary

This chapter focused on the system and on how the different parts related to each other. In a high level perspective, the system can be divided into three parts, being the smartphone, the robot (leader) and the communication (Wi-Fi).

When considering a more detailed perspective, the smartphone's part of the system runs the developed Android application which is divided into Logic and UI and Communication modules. The Logic and UI is responsible for acquiring and treating the input data from the user's interactions with the smartphone and the Communication makes sure that the data is sent to the other side of the TCP/IP socket connection.

On the other side, a ROS node runs a TCP/IP server that after successfully connecting to the smartphone, keeps receiving data that is mapped to the other ROS nodes of the system, including velocity commands and formation type.

There are other important ROS nodes that guarantee the integrity of the system.

4 Interaction Modes

Interaction Modes and the way they can reflect users' intentions are one of the key aspects of this work and the biggest challenge to reach the objective of improving the overall interaction between humans and robots.

This chapter focuses on this problem, starting by analyzing the different kinds of possibilities that smartphones offer in terms of ways of interaction, then proceeding with explaining the process of design and creation of the final three IMs that integrate the application. It also covers the details behind each type of interaction, providing information on the choices made, specifically, implementation and user's feedback decisions.

4.1 Interaction Possibilities

The revision of the State of the Art presented in the Related Work (section 2) has shown the categories that can be identified within HMIs and has provided the relevant divergences between them. By analyzing the current state of smartphones' development and the technologies that they incorporate, it is fair to say that most of the described categories are available in smartphones. In fact, after a detailed observation of smartphones' specifications and their capabilities, only bionic technology seems to be far from what can currently be achieved with these amazing devices:

- **Optic-based** - smartphones incorporate cameras that can be the starting point of visual data acquisition, for example, to process that data using Computer Vision algorithms and produce outstanding results in terms of gesture recognition, body language interpreters or motion detection.
- **Acoustic-based** - if not for the later developments, then for the initial purpose of why mobile phones were created: to receive and perform phone calls. To do so, they have always relied on speakers and microphones, thus enabling from the early days the possibility of using them as a source for interactions to be created. The current state

of smartphones, with more advanced systems and hardware provides the possibility of using those resources for speech and voice recognition, perhaps even to interact directly with robots.

- **Tactile** - in conjunction with optic-based technologies, smartphones are now mainly used by clicks, touches and drags within the touchscreen they provide. This is currently the most basic mode of interaction with smartphones and can be used to perform a large number of actions. Even the physical buttons, like the volume and screen lock keys are a way of tactile interacting with the smartphone.

- **Motion** - a technology that is probably of one the latest to be featured in smartphones is also a starting point to many possibilities of interaction. By using gyroscope, accelerometer and magnetometer, the motion of the device can be detected and used to perform a variety of actions. In fact, the accelerometer provides the gravity vector and the magnetometer behaves like a compass. The joint usage of these values allows to estimate the smartphone's orientation, even though it is not very accurate. On the other hand, the gyroscope is much more accurate but presents itself with serious drifting problems. To avoid these problems and rely on a much more robust system, what happens is that the gyroscope values are used for registering changes in short periods and the joint values of the accelerometer and magnetometer work as support information for longer periods of time. This results in a fused method, that uses the three sensors [39].

Another thing to notice is that most of these technologies are interconnected and in smartphones that occurs too. Most of the interactions that users have with them can be identified as a mix of several of these technologies, instead of isolated, simple interactions.

In the particular case of designing new ways of interacting with robots, the idea of combining these technologies seems to favor usability. By promoting this approach in this work, it is intended that the user feels more confident and comfortable when using the application.

After debating the pros and cons of each type of technology, it was decided to use optic-based, tactile and motion types. The application should provide simple but efficient ways of interacting with robots and keep away complexity as it is a potential source of problems. The decision of leaving out the acoustic-based ones is due to the fact that voice and speech recognition are a very complex field of research and matter for a fully independent thesis on interaction. Also, gesture recognition seemed to be a field too large and complex to use here, so it was kept aside, at least as far as this work goes, even though both of these technologies are very promising in robotics.

4.2 Process of Creation

Interaction is fundamental in robotics and in the way people accept robots. This work started off to solve the problem of interaction between users and mobile robots in the laboratory, because the previous methods of interaction were not fully exploring the possibilities and relied on a *Wiimote* (Nintendo Wii Remote controller [38]). In fact, the creation of smartphones moves a whole industry and is researched over the globe to find the best designs that fit the most diverse people. Because of this, smartphones are already well designed devices that people are accustomed to use.

The idea here was to, firstly, understand the full potential of smartphones and then, after deciding the technologies to use, evaluate the best ways to enable the user to navigate a robot. Usability and easiness of understanding how to use the application were always the guidelines for the development of the software.

This was, of course, an iterative and interactive process, in which different opinions were taken into consideration, so that the final decisions reflect different perspectives. The first ideas were based on what was and currently is the trend in videogames and on the origins of controlling the motion of a virtual character. This seemed to be a good starting point as most of these mechanisms are already evaluated and validated worldwide. So the first sketches began taking form.

4.2.1 First Iteration

The first idea was to keep the interface as simple as possible. To do that, an interface in which 4 arrows were presented to the user (an up and down arrow to navigate the linear velocity and a right and left arrow to rotate accordingly) was sketched, with the results visible in figure 4.1.

This IM was designed to be simple and easy to use. It was thought to be used with clicks and touches (clicks and touches differing in the amount of time of the action, meaning that a touch is detected until the user releases the finger off the screen). The user could also easily swap the formation, by clicking the *Formation* button or could quickly stop the system by

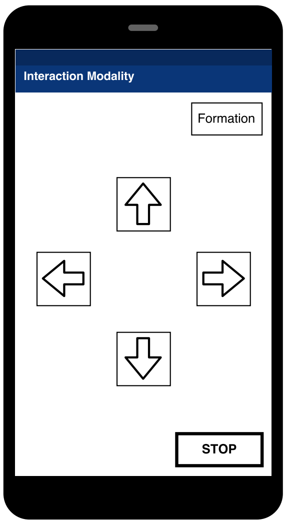


Figure 4.1: First iteration on an IM based on arrows.

click the *STOP* button.

Another two types of interactions were thought and sketched, one that used the motion sensors of the smartphone to control the motion of the robots, and the other one that relied on swipes and drags on the screen, based on the position of the finger in each moment in time.

The sketch for the motion sensors mode can be seen in figure 4.2. This approach would consider the tilt and rotation of the device to estimate the position of a ball inside a square box with the desired motion being derived directly from the position vector of the ball. This way, the ball would reflect the motion inflicted on the robot.

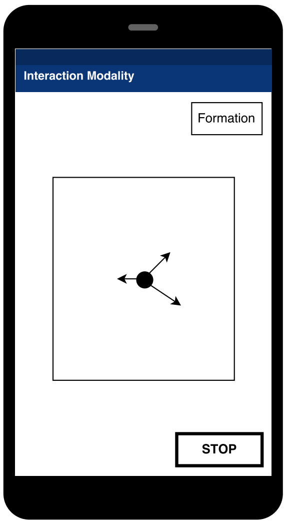


Figure 4.2: First iteration on an IM based on motion sensors.

The other IM was seen as an interpreter of lines drawn on the screen. This way, the start and ending positions would be different and the application would then be able to determine the displacement vector and reflect that into velocity commands. A possible representation is seen in figure 4.3.

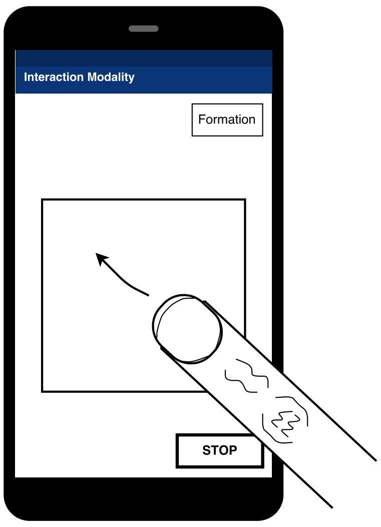


Figure 4.3: First iteration on an IM based on touch gestures.

The interactions all allowed the user to change the formation and stop the normal execution of the session in the same manner, as was described before, by clicking the appropriate buttons.

4.2.2 Second Iteration

Besides being good starting points, the first iteration modes did not suffice and presented many flaws that were not desirable for the final product. This fact led to a second iteration having as guiding lines the results of the first iteration, in which the corresponding modes were modified and redesigned to match the flaws encountered and in which a new kind of interaction arose.

The mode based on arrows stayed almost intact as far as navigation goes, but one thing that was changed was the number of buttons available on the screen. In this new version, both the *Formation* and the *STOP* buttons were removed from the screen so that the user could stay confident about the navigation and would not have to worry about the possible clicking of an unwanted button. With this approach, the navigation was privileged and the formation could be changed by accessing the Preferences Activity, on the top right part of the screen, by clicking the three-dotted button. The corresponding sketch can be seen in figure 4.4.

It is clear that the previous buttons disappeared, giving the user a more comfortable navigation experience. A *Connection* button was added which was thought as a safety mechanism that would not allow the user to start operating the system without clicking it.

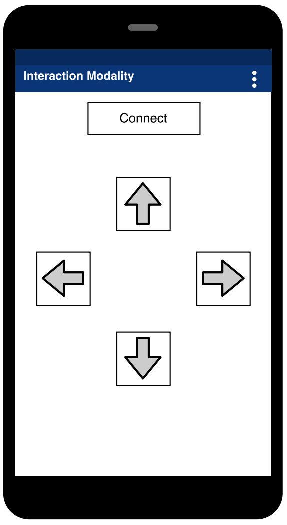


Figure 4.4: Second iteration on an IM based on arrows.

The second iteration on the motion sensors mode reflected a major change in terms of the bounding box around the ball in the center. The basic principles stayed the same, with the ball's movement being dependent on the tilt and rotation of the device with respect to the correct axes, but now the ball was thought to be moving inside a circular box. The ball's position on the screen was calculated with respect to the rotation degrees and the direction of the movement reflected the movement of the robot being controlled. The second version of this mode can be seen in figure 4.5. The same rules regarding the buttons were carried out on this design, having only the *Connect* button.

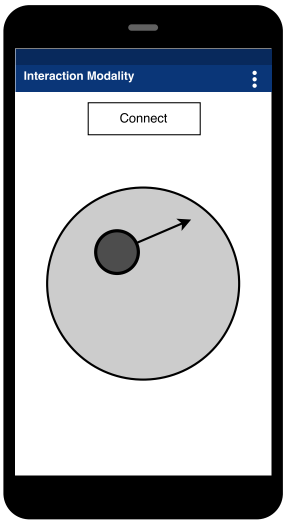


Figure 4.5: Second iteration on an IM based on motion sensors.

The first iteration on touch gestures was not viable and was the inspiration for two new kinds of interfaces. The first one is seen in figure 4.6, reflecting what is widely known as a Joystick. In gaming industry, this kind of interaction is well known for quite some time and has remained one the fans favorites because of how intuitive and easy to control it is. This was the main idea behind this second iteration. With this approach, the user would only need to click and drag the handle (the circular box in the middle) towards a determined location within the other circular box in order to control the movement of the robot. The other interface sketched was seen as a replacement of the traditional electric cars controllers, in which two slider handlers allowed the user to control both the linear speed and the rotation of the robot, by sliding the handlers accordingly. This can be seen in figure 4.7

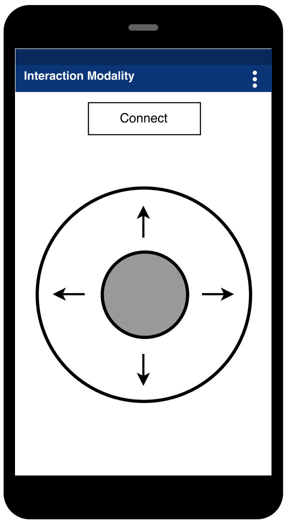


Figure 4.6: Second iteration on an IM based on touch gestures (Joystick).

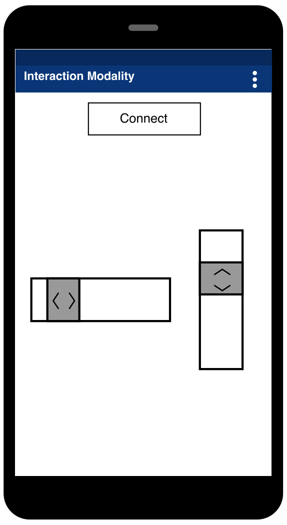


Figure 4.7: Second iteration on an IM based on touch gestures (Sliders).

Even though this second iteration resulted in better working models of the interfaces to be incorporated in the application, there were still some issues regarding usability and the way the user had to perceive the screens in order to interact with the robots. Due to the dynamics of the screen, the second touch gesture approach (the one that relied on sliders, in figure 4.7) was abandoned, because it did not feel as natural and easy to understand as the other ones.

Some of the adjustments, such as relieving the screen from having several support buttons came out to be bad for interaction because now the user had to perform a series of steps, including changing screen to the Preferences Activity, in order to change formation. Also, in this design, the stoppage of the system would be done by clicking the same button used to start the connection. This seemed to be a bad approach and did not feel right for the user, again because it populated the screen with a button that even though had two purposes and was fixed in the same position, could be the source of accidental clicks.

4.2.3 Third Iteration

The third iteration on the IMs considered the decisions taken on the previous iteration regarding the types of interactions to implement, aiming now to solve some of the issues presented by the previous versions.

The final three modes would then be: one in which navigation would be based on arrows, being a simple and intuitive way to control motion; another one based on the data acquired from the gyroscope, accelerometer and magnetometer, using that data to control the robots; finally one that would behave like a joystick, based on clicks and drags done by the user's finger on the screen.

The issues were considered and solutions presented:

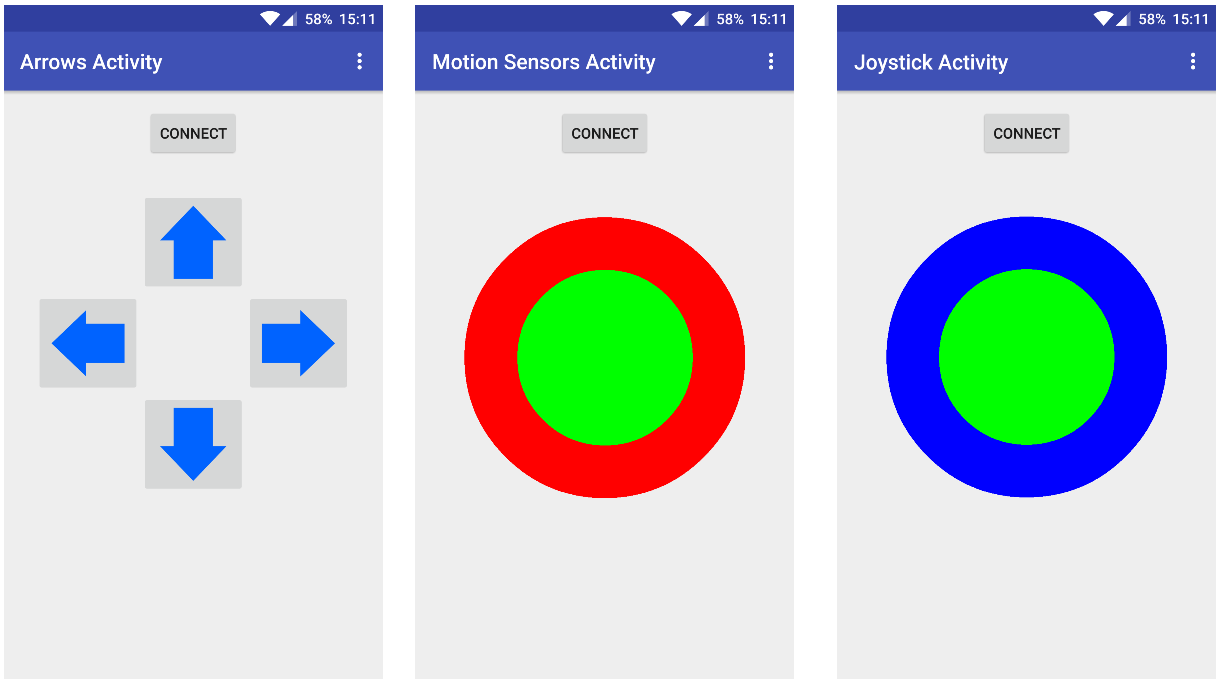
- **Connect button** - a button to establish the connection is considered vital because the user has the ability and control over when to start the interaction. The point here is for the button to disappear after the connection is establish and thus free the screen from accidental clicks.

- **Stop button** - the decision here was to rely on the physical buttons of smartphones, more specifically, the volume keys. With this approach, the user can easily reach for those buttons and efficiently stop the connection and the motion of the robot (by clicking any of the two volume keys).

- **Changing the formation** - changing the formation required the user to access the application preferences, without being able to do it in a fast way. This posed to be a problem because it meant stopping the navigation of the system for a relatively long period. To overcome this situation, a new feature was implemented, fully exploring the smartphones abilities: a double tap on the screen was mapped to change the current formation, thus allowing the user to easily swap formation without compromising the navigation.

- **Navigation between interfaces** - navigation between different interfaces was still not intuitive enough and also required the user to access the preferences. A new, easier way had to be implemented.

After these three iterations, the types of interactions were defined and the interfaces started to take form. After this step, the interfaces were already functional in terms of robot navigation and looked like the representations seen in figures 4.8a, 4.8b and 4.8c.



(a) (b) (c)

Figure 4.8: Results of the third iteration: (a) Arrows Activity. (b) Motion Sensors Activity. (c) Joystick Activity.

After this third iteration, many of the problems had been solved, even though the visual design was not good and there were still some issues regarding crucial parts of the interactions, such as changing between the different interfaces and adjusting the visual feedback presented to the user.

The following steps of the development were deeply related to solving the detected problems and enhancing the visual aspect of the application, aiming at producing quality usability and making the interface as much user-friendly as possible. After a series of tests and informal validation with different users, the feedback received was taken into consideration to create the final solution, which will be discussed in the next sections, focusing on the details of each of the interfaces.

4.3 Arrows Mode

The final version of the *Arrows Mode* took into consideration every bit of advice and feedback that was receiving from people and the visual design was dedicated to make the experience feel just right for the user. The final result can be seen in figure 4.9.

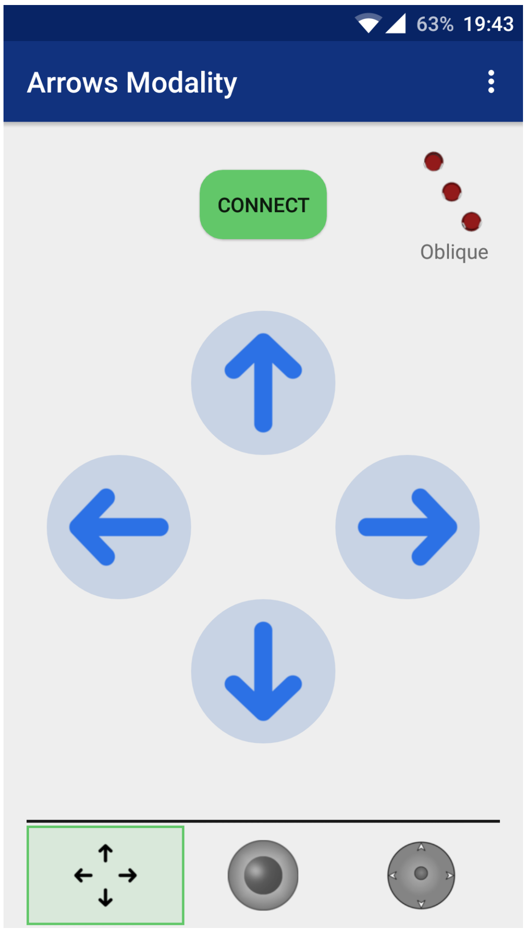


Figure 4.9: Final result of the IM based on Arrows.

In this kind of interaction, the way things are processed is not really complex. In fact, what happens is that there is a *OnTouchListener* on each arrow, waiting for the user to touch one or two arrows simultaneously. When that happens, the callback function gets called and the process is to check which arrows have been pressed, then apply a constant value of velocity to the corresponding linear and angular components.

By the time the function reaches the end, the values of linear, angular velocity and current formation type on the *RosConnection* object are set accordingly. After that, this communication object takes care of the command and sends it, as described in section **Communication Details** (page 22).

If the formation type is changed, then the corresponding formation type variable of the *RosConnection* object is updated so that when a new command is sent, the information is up-to-date.

From figure 4.9, it can be seen that the environment seems a little different and that entities other than the *Arrow* buttons are present. Even though that is true, the design was projected so that the risk of accidental clicks was minimized. That was done by registering other types of touch-based gestures. The same points apply to the other two IMs that will be feature next. These are the topics to be considered in this matter:

- **Connect Button** - the green *Connect* button allows the user to establish the connection with the robot. Once a connection is accepted, the button will disappear from the screen.

- **Disconnect/Stop Button** - to disconnect from the robot or stop its motion without being able to move it accidentally by clicking the arrows, the physical lateral volume keys can be pressed. What happens is that the application ensures that the robot receives commands to keep its velocity at zero and after that, closes the connection. When this occurs, the *Connect* button appears again on the screen, at its initial position and if the user wants to keep navigating the system, just clicks the *Connect* button again.

- **Changing the formation** - it can be seen that the current formation is shown on the top right corner of the screen, with an indicative text being supported by an image that suggests the current state of the geometric pattern of the group of robots. To change the formation without having to leave the IM, the user can simply double tap the screen somewhere other than the arrows area. This means that for the formation change to be accepted, the user has to quickly click two consecutive times around the same area, in which case a message will be briefly presented and both the text and image of the formation are changed to match the Column formation type. Because there are only two types of formations, if the user repeats the same action, the formation will be back to Oblique.

- **Navigation between interfaces** - as it can be seen in the lower part of the image represented, there is a new group of elements on the screen. These elements correspond to the lower screen navigation bar, which allows the user to quickly change the IM without having to rely on the Preferences Activity. The current IM, in this case, the *Arrows Mode*, is highlighted in green for the user to know which one is selected. To avoid unpleasant accidental navigation, another kind of click was registered in this lower bar: a Long Click. Differing from normal clicks, this kind of touch gesture takes a little longer that the normal click to be accepted, thus enabling a different kind of interaction. By doing this, the problem of accidentally clicking this area is solved. To prevent other incidents, when the user changes IM, the *Connect* button on the recently launched IM will appear and will only allow navigation of the robots after the user clicks on it.

4.4 Motion Sensors Mode

The final version of the *Motion Sensors Mode* also reflected the feedback obtained from several different people. The result can be seen in figure 4.10.

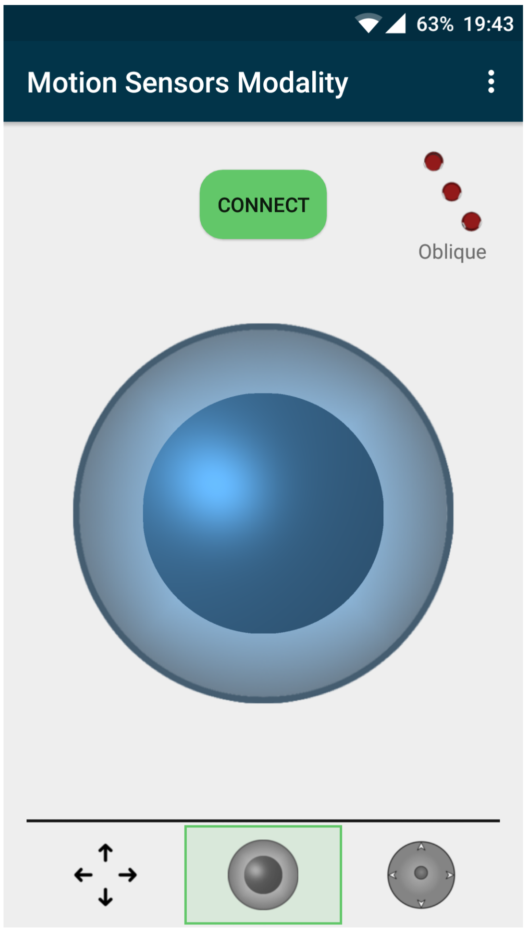


Figure 4.10: Final result of the IM based on Motion Sensors.

In this version, one of the key aspects of improvement was the visual looks of the interface. This time, a circular bounding box is presented in more pleasing tones and the ball inside it gives the sensation of being a 3D item. The metaphor behind this interaction is actually a 3D round object that moves according to the inclination of the surface in which it is inserted. And keeping in with this idea, by leaning and tilting the device, that is what happens to the virtual object: it appears to be moving towards the limits of the circular box. And the movement that the ball appears to be doing reflects how the actual robot is moving in real life (regarding the direction and velocity of the movement). The further away from the center of the circular box the ball is, the faster the robot is moving.

To implement this IM and having to rely on the sensors that smartphones incorporate, the challenge was firstly to find a way to obtain viable information from the raw data coming from the sensors; secondly was to fuse the data from the three different sensors; and lastly was to deal with the drifts problems. Luckily, and because the community behind Android is huge, there have been quite a few projects that relate to this and after some research, one important and well planned, tiny library was found that treats the data and enables the programmers to get viable data very easily [39]. This library was created by Paul Lawitzki and further refined by Jose Collas [40].

After having the viable data from the sensors, there is still the need to match that data with proper velocity commands. Every time a new value is read from the sensors, the callback function is called to prepare the data. Basically, the rotation degrees, α and β, along the two relevant axes, **x** and **y**, respectively (Fig. 4.11), are acquired and divided by constants, $SPEED\_CONST\_A$ and $SPEED\_CONST\_L$, and then these values are mapped into velocity components, as the following:

$$\vec{v} = \begin{bmatrix} v_a \\ v_l \end{bmatrix} = \begin{bmatrix} \frac{\beta}{SPEED\_CONST\_A} \\ \frac{\alpha}{SPEED\_CONST\_L} \end{bmatrix} \tag{4.1}$$

The position of the ball, $\vec{p}$, inside the bounding box can then be set to:

$$\vec{p} = \frac{\vec{v}}{\|\vec{v}\|} \times \Delta r + boxCenterPosition \tag{4.2}$$

Where Δr is the difference between the outer and inner ball's radius.

In the same manner as with the Arrows Mode, when the callback reaches the end, the *RosConnection* object already has the most updated values of velocity and is ready to send them as a command to the robot.

As was said in the previous section, the same criteria apply to the buttons that appear on the screen in figure 4.10.

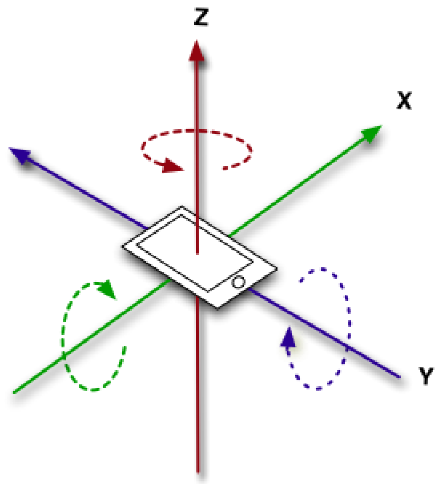


Figure 4.11: Representation of the smartphone's sensors axes.

4.5 Joystick Mode

This final version of the *Joystick Mode* tried to represent the usability provided by game controllers of this kind. The result can be seen in figure 4.12.

In this version there was a huge improvement in terms of the visual aspect of the joystick. It is composed of two distinct parts: the base, shown in green tones and possessing indicative arrows towards each direction; and the handler, with which the joystick will produce effect, presented in grayish tones at the center. To use this interface, the user has to click on the area delimited by the handler and then drag the handler, without dropping the click, towards a position that better reflects the intention of the navigation. This means, for instance, that if the user wants the robot to move straight forward, then has to move the handler straight towards the up arrow on the screen. To give the idea that the user is actually moving something, the handler will appear to be moving according to the movement of the user's finger on the screen.

The data that is used to produce viable commands is thus based on the position of the finger and with respect to the center of the joystick. So, the callback function will calculate the displacement vector, $\vec{d}$, between the current finger's position and the center of the joystick base, mapping that into two velocity components, $\vec{v} = (v_a, v_l)$, where v_a is the angular component and v_l the linear component as the following:

$$\begin{bmatrix} v_a \\ v_l \end{bmatrix} = \begin{bmatrix} d_x \times VEL\_CONST\_X \\ d_y \times VEL\_CONST\_Y \end{bmatrix} \tag{4.3}$$

Where VEL_CONST_X and VEL_CONST_Y are velocity constants that map this

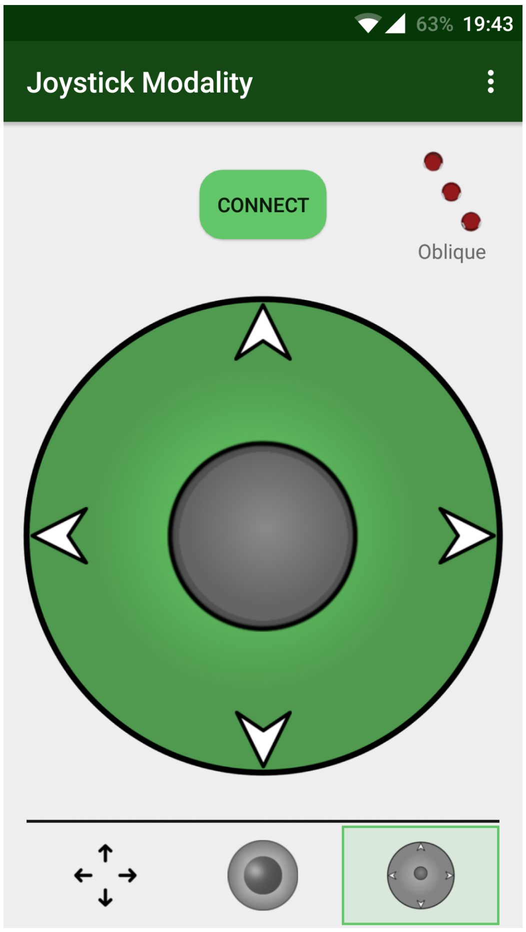


Figure 4.12: Final result of the IM based on Joystick.

displacement in pixels to viable velocity values, which are then set to the *RosConnection* object.

As said before, this mode relies on the same principles that were presented when the *Arrows Mode* was introduced, regarding the new buttons.

4.6 Other important points to be considered

Now that all the final IMs have been explained, there are some points that need clarification, still regarding the types of interactions designed.

To further enhance the interactions, some mechanisms of feedback were provided in the application. For instance, when the user clicks an arrow in the *Arrows Mode*, the device produces a vibration to let the user know that an arrow has actually been clicked. The same thing happens when the user changes the IM by interacting with the lower screen navigation bar. When the long click is accepted, a vibration is felt as well. These feedbacks can also be turned off through the Preferences Activity, if the user so desires.

Another way of giving feedback to the user is by presenting temporary messages after a variety of actions are performed, for example, after a connection is successful.

One other important feature that this system has is the ability to provide feedback if the robots are moving towards obstacles. This feedback is given by producing a vibration each time the robot is stopped because an obstacle has been detected and the user tries to keep moving it in a direction that directly collides with the obstacle. This way, if the user does not perceive this problem at first glance, the vibration will certainly provide some additional information and prove that something is not okay with that type of motion. Because promoting the user needs was one of the guiding lines throughout this process, it is also possible to rely on a visual feedback related to the distance to obstacles, by accessing the Preferences and activating the distance feedback bar, which will position a small bar onto the screen that changes color according to the distance to detected obstacles: green means no obstacles detected, yellow means an obstacle has been detected and is within the safety line, red means the critical line has been reached. Figure 4.13 illustrates how the information used for these feedbacks is obtained.

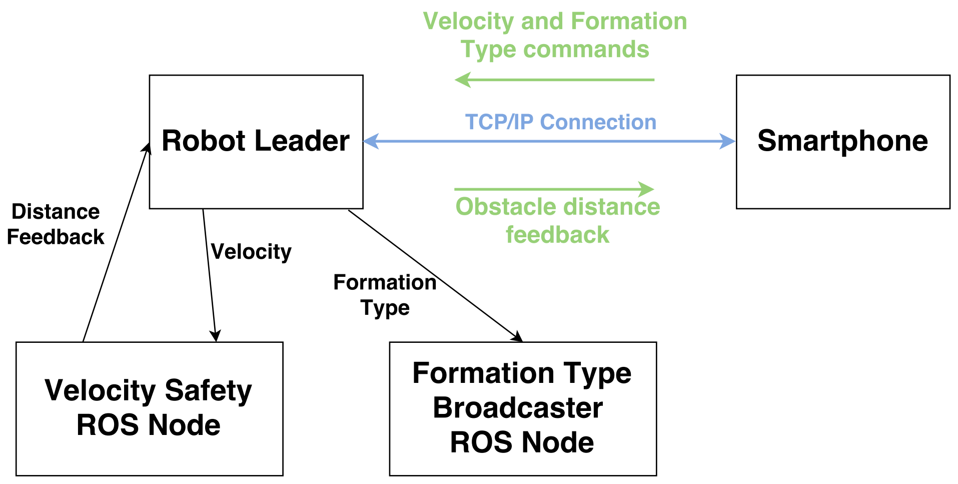


Figure 4.13: Representation of the origin of Obstacle Distance Feedback.

It is also important to refer that this kind of application involves presenting a lot of visual content to the user which in a mobile device, may pose some trouble when rendering is done, regarding memory. To prevent memory problems and to allow the user to change very quickly between IMs, a simple caching system was implemented using a Java structure called *Singleton* [41], which ensures that during execution, only one object of that type (Singleton) is instantiated. It is useful here to prevent copies of the same object. Within the Singleton, the source bitmaps that populate the screen are stored, thus guaranteeing that those bitmaps are not stored in memory each time an Activity is launched. This way, when rendering begins, firstly the cache is checked for possible matches for each bitmap and if a match occurs, there is no need to load it again. This reflects in a much smoother session for the user and presents better battery life and memory conditions to the smartphone.

4.7 Summary

This chapter follows the creation process behind designing new ways to interact with robots. After reviewing opinions and suggestions made by different users, an iterative process began and new sketches started to take form, regarding the different types of IM and interfaces.

With each iteration, some issues were solved while others arose, some of which proved to be challenging to solve. The final solutions gradually appeared and converged to three distinct IMs, which were implemented in the Android *RobotShepherding* application.

The final three IMs were then presented: Arrows Mode, Motion Sensors Mode and Joystick Mode. The Arrows Mode aimed at providing a simple solution, available to everyone, even unexperienced people in controlling robots. The Motion Sensors Mode, provides an interaction based on tilting and rotating the mobile device, keeping it as much intuitive as possible. The Joystick Mode, based on games joysticks that have been on the market for years, provides a smooth interaction with the robots. To favor usability, many feedback mechanisms were also implemented and incorporated in the application, including haptic feedback and visually pleasing animations that behave according to what the user desires to do with the robots. The application also allows the user to change formation with a very simple double tap mechanism.

5 Experiments, Analysis and Results

This chapter aims to present the reader with the experiments that were designed to evaluate and validate both the application, regarding usability and user experience, and the functioning of the system, in relation to its performance and reliability as a whole.

Firstly, a usability and user experience study of the application is shown, covering the process behind the experiments with several individuals. This section will focus on the objectives of each test and on the analysis of the results of each person involved in the study.

On a latter stage, an analysis of the system with respect to its performance is presented, providing some final thoughts on its behavior, with emphasis on the validation of the functioning of the formation control mechanisms and the collision safety layer.

It is important to notice that all the tests and results shown concern the usage of the application and the robots in real environments.

It is also relevant to know what type of robots was used. With respect to this, one of the most popular mobile robotic platforms within research community was used: Pioneer P3-DX [42] (Fig. 5.1).

This reliable, versatile, two-wheel robot comprises frontal sonar sensors and is used for many applications. In the scope of this work, up to three of these robots were used to perform tasks of teleoperation and formation control.

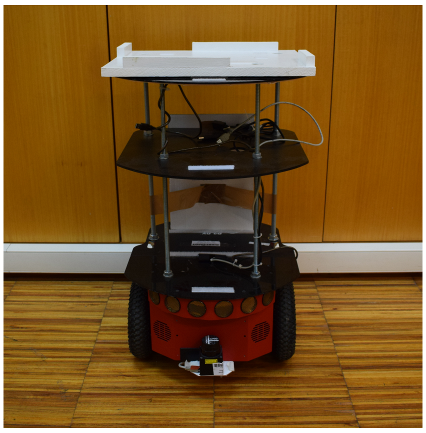


Figure 5.1: Robot Pioneer P3-DX, which is the type of robots used throughout this work.

5.1 Usability and User Experience

Even though the concepts are deeply related, they are distinct when considering the final objective of the each of them [43]:

- **Usability** - concerns the ability to perform something in an easy or intuitive way. In this particular case, the objective of these tests is to evaluate the performance of each user when navigating a robot with each type of IM, so that conclusions on the degree of usability of the IMs can be extracted in order to improve what in the eyes of the users needs to be improved.

- **User Experience** - user experience tries to evaluate how the user feels when interacting with the application. It is somehow related to the emotional part of the user and to the reactions that each particular detail of the application provokes in each user. For instance, one user may find one particular aspect of the application very pleasing while others react with frustration.

This slight difference is responsible for having cases in which, for example, a user performs really well in a certain task when using the application, even though he/she does not feel emotionally sympathetic with the way the application is built (or cases in which what happens is exactly the opposite).

In the specific case of this work, the *usability* and *user experience* tests try to understand how a user performs and feels when using the developed application with one single robot, in a controlled real environment, with the final objective of improving the neglected aspects.

Even though they evaluate different points, the two types of tests were performed simultaneously because users were presented with navigation challenges while using the application and its resources.

Regarding *usability*, a circuit was planned and established in the laboratory, as shown in Fig. 5.2. For the purpose of these tests, the users would start off in the initial position marked in Fig. 5.2 and would try to navigate the robot through the circuit until reaching the finishing line, with each one of the three different IMs. The times taken by each user to complete each of the three navigation tests were saved for further comparisons.

After performing these tests, always in the safest operation mode (lower velocity), the users were then allowed to experiment the other operation modes to verify the relationship between the operation modes and the global usability. After having had a solid experience

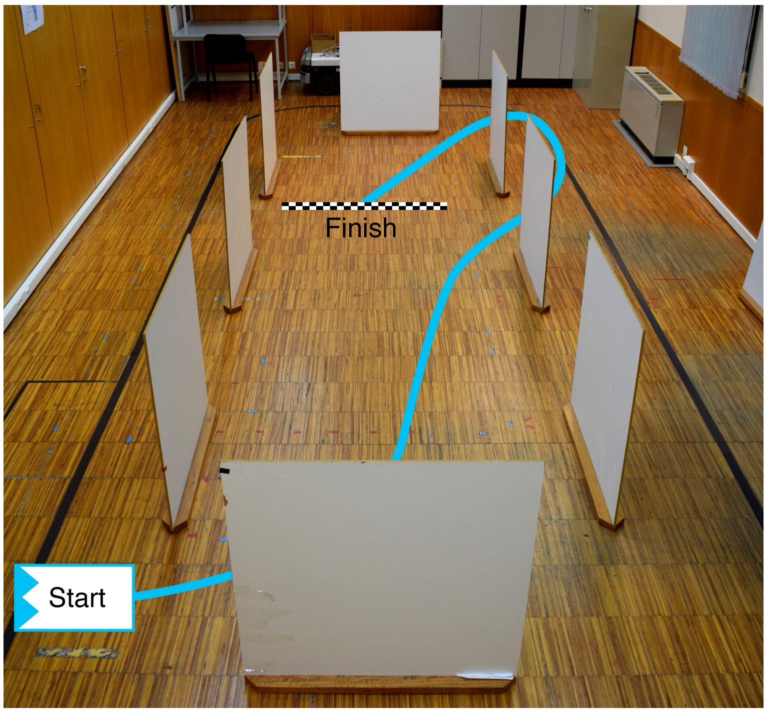


Figure 5.2: Circuit established for performing the *Usability* and *User Experience* tests.

with the application and the robots, each user was asked personally to vote for the preferred IM.

It is relevant to mention that before starting the tests, each user was given a small amount of time to get acquainted with the details of the application and freely move the robot using the different IMs.

To evaluate *user experience*, two methods were considered: the first, informal and purely indicative, consisted on keeping attention to the reactions of the users when performing the previously referred tests, to take note of the most frustrating general aspects in each type of IM; the second, a formal, direct reflection of the users' feelings, consisted on a questionnaire, designed to analyze every relevant aspect of the experience (from the visual design to the specific feedback of each mode). This questionnaire also presented the user with several questions about the interaction and usability of the application, once again proving the relation between the two concepts.

To summarize, the questions were to be answered in a scale from 1 to 5 (with 5 being the best classification), and the following points were considered in the questionnaire:

- For each IM screen, the user was asked to evaluate the *Visual Aspect and Content Distribution*;
- For each IM screen, the user was asked to evaluate the *maneuverability of the robot*, in other words, to evaluate how controlling the robot was;

- For each IM screen, the user was asked to evaluate the *feedback* with relation to the haptic feedback in the *Arrows Mode* and the feedback of the robot's movement in the *Motion Sensors Mode* and the *Joystick Mode*;

- The user was then asked about general aspects of the application, such as: the overall *design*, both visual and in terms of the screens available, to check how pleasing and intuitive it felt to them; the *navigation between IMs*; the way the application allows to *change formation* (even though the actual change of formation was not incorporated in these tests); the usage of the *STOP button* and the *Preferences*; the total *time until the start of interaction*; the *differences between* each *operation mode* (the velocity); and finally about the overall *user experience.*

5.2 Results

For the *usability* and *user experience* study, a sample of 21 individuals was selected to perform the tests. This sample (over 20 individuals) seemed to suit the amount of people necessary to evaluate and validate the application with a good degree of reliability.

After reviewing the answers to the questionnaire as well as the collected times for each IM with regards to each user, the results were comprised into graphics, to better show the evaluations and the tendencies of the experiences of the users.

Regarding the *Visual Aspect and Content Distribution*, it can be said that in general, people seemed to like the way the application shows the content to users. In fact, as shown in Fig. 5.3, which represents the evaluations by individual, for each IM, the public seems to find the visual part of the application to be pleasant. To further clarify this information, Fig. 5.4 represents the average evaluation of each IM's *Visual Aspect and Content Distribution*, according to the classification provided by the users.

It is possible to see that, even though the average evaluation is quite high, people tend to like the *Arrows Mode* a little more, when it comes to the visual part.

One of the key parts to evaluate in this specific scenario, when it comes to *usability* is the *Maneuverability* of the robots that each IM can offer. This is the reason why this type of evaluation had to be in the questionnaire. The data collected and comprised from the answers can be seen in Fig. 5.5, which establishes a comparison of the maneuverability of the robots using each type of IM, by individual.

In this question, it is clear that there are more divergences than in the previous case.

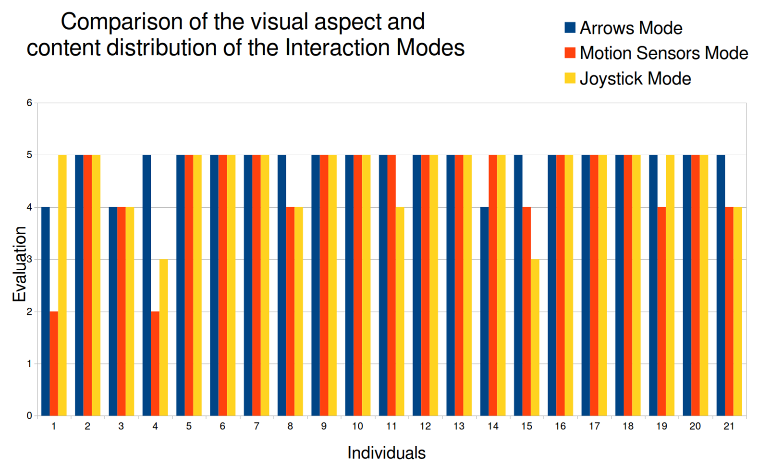


Figure 5.3: Evaluations of the Visual Aspect and Content Distribution by individual.

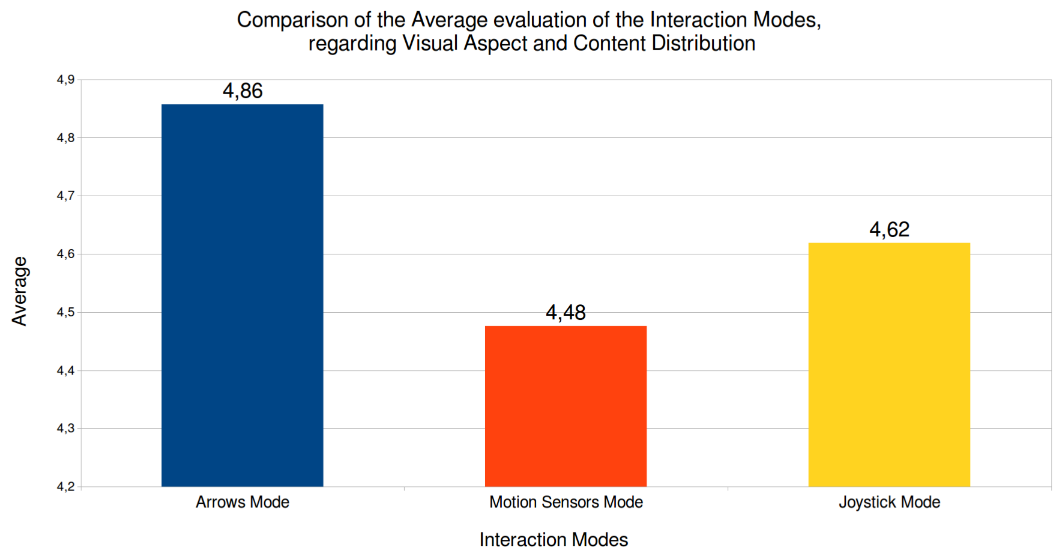


Figure 5.4: Average evaluation of each IM's Visual Aspect and Content Distribution.

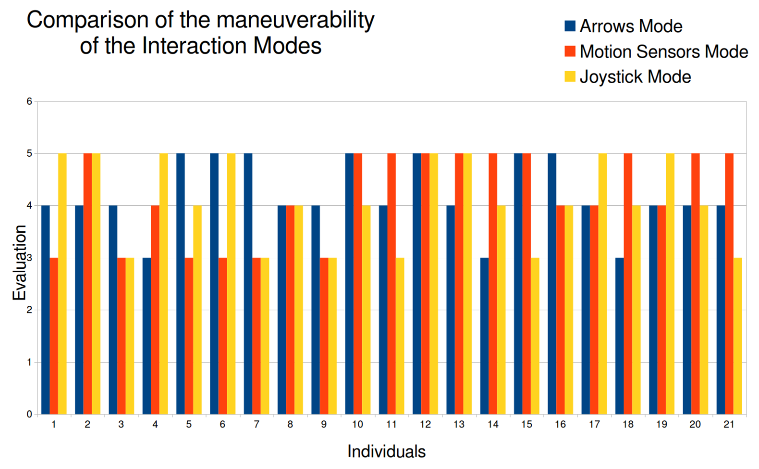


Figure 5.5: Evaluations of the Maneuverability of the robots, using each of the IMs, by individual.

These divergences are observed by the peeks in the graphic, which are not distributed in a uniform way: people do not all agree when it comes to maneuverability. Fig. 5.6 shows the average evaluation of the IM's maneuverability with regards to the robots.

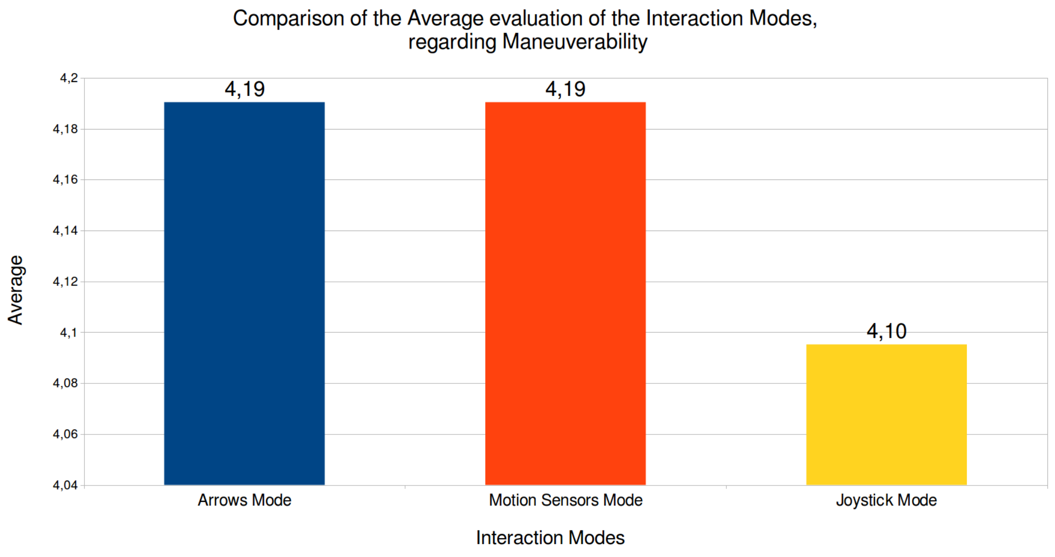


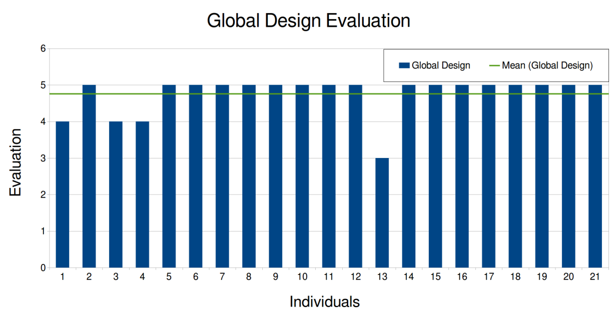
Figure 5.6: Average evaluation of each IM's Maneuverability of the robots.

Although the results are quite good, as the average maneuverability is above 4 in all three cases, the *Joystick Mode* appears to be the most challenging mode to use when controlling robots. After reviewing the data of the actual navigation of the robots and the initial visual impact provoked by the application on the user, there is still the need to analyze the remaining information from the questionnaire. In that instance, Fig. 5.7 shows the representations of each of the crucial aspects that were studied and observed.

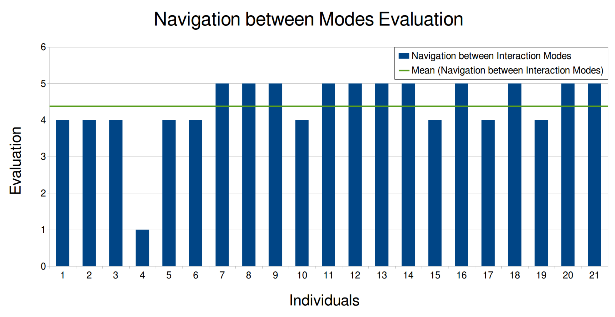
A green line represented in each of the graphics represents the average value that relates with the users classifications. Once again, although there are some divergences, people tend to be comfortable and feel good about using the application, reflected by the positive responses in the questionnaire. Every single one of the tested aspects of the application has an average classification above what is accepted as good, but this does not mean that everything within the application was good.

As a matter of fact, many suggestions and opinions were given regarding many issues, and many of these critics helped to improve the details that were not yet considered good, as will be covered in a latter section.

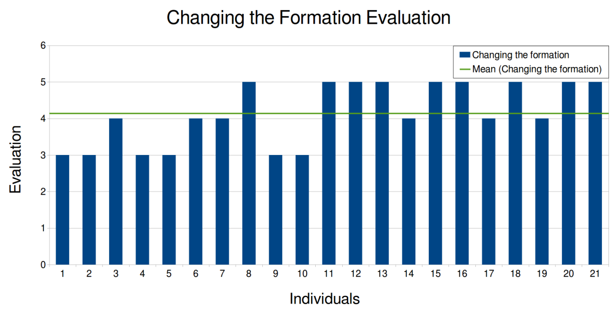
The other really important part of the tests was the performance in the circuit tasks. The mean to evaluate these performances is the collection of times that were taken for each navigation task. For this part, what seemed to be important to evaluate was the average time that an unexperienced user took to complete each task with each IM. These results can be seen in Table 5.1, which also incorporates the maximum and minimum times taken for each type of interaction.



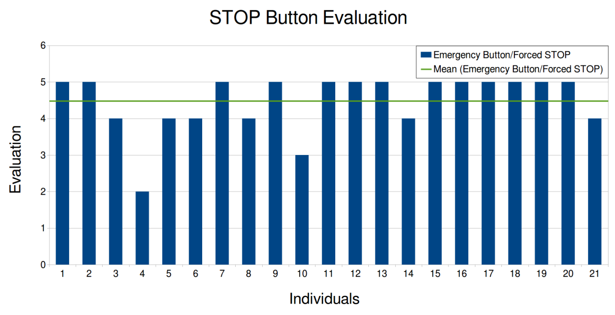
(a) Global design appreciation.



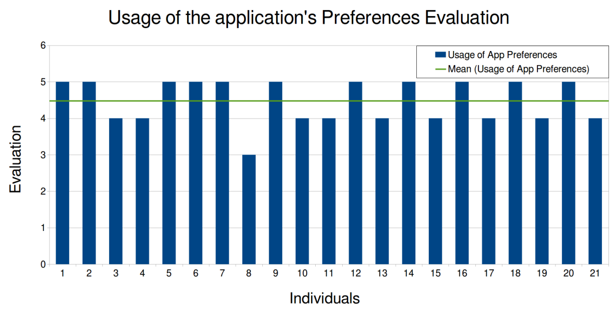
(b) Navigation between different IMs appreciation.



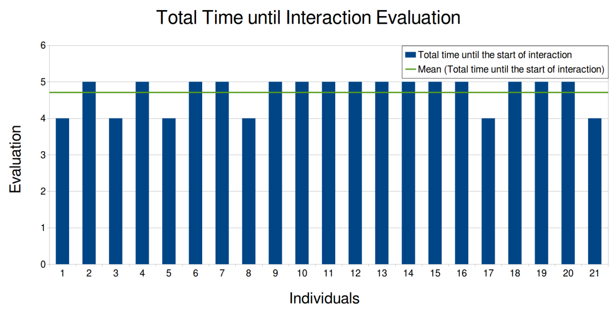
(c) Formation change appreciation.



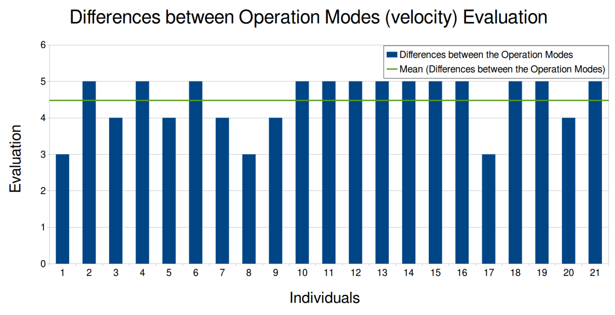
(d) Usage of the STOP button appreciation.



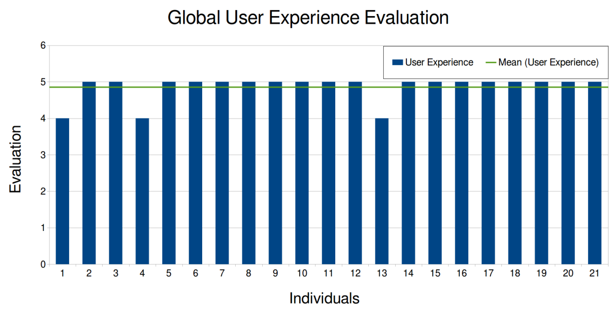
(e) Preferences usagen appreciation.



(f) Time to begin interaction appreciation.



(g) Differences betwee the Operation Modes (velocity) appreciation.



(h) Global User Experience appreciation.

Figure 5.7: Representation of the data taken from the answers to the questionnaire. Each of the graphics evaluates a different aspect of the application.

From Table 5.1, it is fair to say that the average times to complete the circuit for each

Table 5.1: Relevant times taken from the data collected when users performed the circuit tasks with each of the three different IMs. The times are indicated in minutes (mm:ss,...).

Interaction Mode	**Arrows Mode**	**Motion Sensors Mode**	**Joystick Mode**
Maximum	01:47,17	01:59,18	01:31,47
Minimum	00:45,62	00:47,16	00:45,40
Average	00:58,33	01:02,52	00:59,89

type of IM are not that far apart. However, it can still be seen that the *Motion Sensors Mode* tends to be the one that takes the most time to navigate the robot from the starting point to the finishing line.

5.3 Analysis

5.3.1 Usability and User Experience

After reviewing in detail the responses of the questionnaire, it is fair to say that most of the users felt good when interacting with the robot through the smartphone. The average scores for almost all categories are high and the main aspects of the application have been accepted and validated.

With these results, it is natural to assume that people probably prefer the *Arrows Mode* or even the *Joystick Mode* when compared to the *Motion Sensors Mode*. This assumption is, in fact, entirely wrong. Fig. 5.8 concerns a representation of the preferred IMs, collected after the individuals finished the tests. By analyzing this figure, it is clear that actually, the *Motion Sensors Mode* is the one people prefer to use. Another thing that can be seen is that despite some percentage differences, users have many different opinions and personal tastes, contributing to a diversification in terms of the preferred way of interacting with robots. One interesting point to consider is that the *Motion Sensors Mode* was the one with the lower average classification in terms of Visual Aspect and Content Distribution, even though it is the preferred one.

Some of these results seem to be contradictory but a little deeper observation proves that that is not the case. During the tests, users seemed to find that the *Arrows Mode* was the most intuitive one and the easier to control the robot. This is related to the fact that navigation through arrows is something that many people are used to do, whether it is in games or other situations. On the other hand, people seemed to like the *Motion Sensors Mode* because of the liberty and fluency of movement it allowed, despite having a larger learning curve. And one of the main reasons behind the lower score in the Visual Aspect

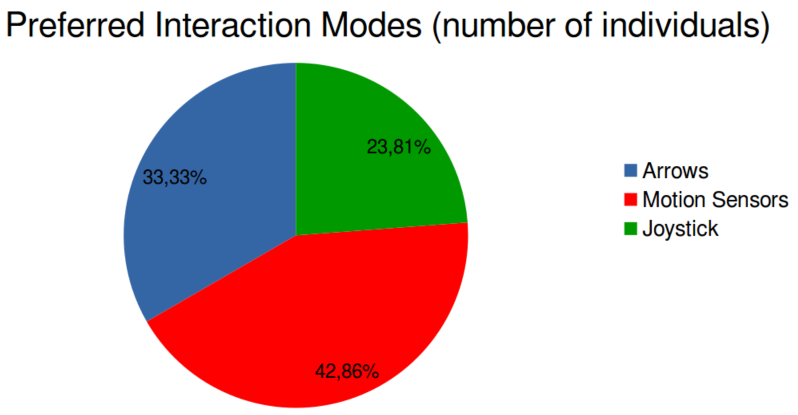


Figure 5.8: Representation of the preferred IMs.

and Content Distribution for this specific IM is that when people use the *Motion Sensors Mode*, they are not really paying attention to what is happening in the smartphone's screen, but rather looking to the robot to have a direct visual feedback.

Regarding the *Joystick Mode*, some people struggled to use it because of some issues related to the rotation of the robot. The fact that the Joystick did not perform as people would have hoped was probably one key factor for having less people preferring this IM. With the feedback collected, a series of fixes to this particular IM were done and it is now much more maneuverable and easier to use. To provide people with better options, it was also decided to incorporate the possibility of inverting the rotation when navigating backwards, as some people prefer it this way.

Other minor corrections were performed based on the suggestions received and it is fair to say that the system improved a lot due to these crucial tests.

5.3.2 Formation Control and Safety Layer

The functioning of the whole system, including the formation control was not the aim of the usability and user experience study. However, of the key features of the application is allowing the user to change the geometric pattern of a group of robots in formation control. Despite the dynamics of the formation control mechanisms being already validated by Daniel Marcelino [3], the system was tested in a more informal way, to see if everything worked as it was projected to work.

To do so, three Pioneer P3-DX robots (Fig. 5.1) were used. The one considered the leader of the formation ran the developed nodes *Android Communication ROS Node*, the *Velocity Safety ROS Node* and the *Formation Type Broadcaster ROS Node*. Each one of the two followers ran the nodes *Formation Type Receiver ROS Node* and a node adapted from the work of Daniel Marcelino which implemented the formation control and allowed to change between two different geometric patterns. After setting up the system, a navigation was performed, changing several times the formation type through the application and observing the reactions from the two followers. Each time the formation was changed, the followers reacted with a change of pattern, confirming the validity of this part of the system. The safety layer was also something tested throughout the navigation, even though it was already being used in the usability and user experience tests. These tests allowed to tune the distance thresholds of the safety layer, as each user has different perspectives of how to plan and coordinate the movement of the robot, specially when navigating through curves.

These final navigation tests with the formation control in place confirmed that the system is working with a significant satisfactory level.

5.4 Summary

This chapter focuses its attention on the *Usability and User Experience* study performed for the purpose of this work, in order to validate and evaluate the application and the navigation of a robot. The objective is to then show the results of the tests performed with 21 individuals that did not have any prior experience with the application and which contributed to evaluate certain aspects of the application and provided suggestions that helped solve some issues.

An analysis of this study is done just before the informal *Formation Control and Safety Layer* tests are presented and analyzed.

6 Conclusion and Future Work

6.1 Conclusion

Interaction between humans and robots has since the origins of robotics been a topic for discussion and is still one of the most researched areas within the community. This thesis takes on a special case in which a group of robots moves as a whole, through formation control dynamics, using a leader-follower approach. In this type of formation control, the leader robot is responsible for the only direct contact with the operator, while the other robots follow the movement of the leader in a certain geometric pattern.

The work developed aimed then at providing new ways of interaction between the user and the leader robot and thus with the fleet of robots. The developed system is composed of various parts that work together to maintain its robustness. On one side, the smartphone is responsible for the teleoperation and the structural integrity of the movement of the robots. On the other side, the ROS nodes make it possible for the robots to receive the data that directly reflects the user's intentions and provide the basis for feedback mechanisms to the user.

The final application provided three new modes of interaction, each with its specific details and exploring different possibilities: *Arrows Mode*, *Motion Sensors Mode* and *Joystick Mode*. The process of creating and designing new IMs is iterative and interactive and in this particular scenario, reflected many of the suggestions and opinions that several different users had to offer.

After the creation of stable versions of the final three IMs, a study of *Usability and User Experience* was carried out to evaluate and validate both the application's design and the most relevant aspects of each IM.

These tests were important to solve many of the issues the application had, as previously mentioned, but also to prove the viability of this work, as it is now clear that people extend their personal tastes even to the way they prefer to interact with robots. By providing different kinds of IMs to the users, it became clear that the notion of the best type of interaction depends on the user feelings regarding these interactions and the emotions they are able to establish with them. This is proven by the fact that all three IMs were chosen several times as the preferred ones, with results showing that there is no perfect IM to match everyone's needs.

One other key point that these tests seem to show is that there is also no direct relation between preferring an IM and performing better in terms of circuit times. Once again, it suggests that the personal preferences are much more related to the emotional part of the individual.

The formation control was also tested in a real environment, to ensure that a user running the necessary software in the robots is able to change formation through the smartphone. This fact was proven with several navigation tests with a group of three robots. The safety layer to avoid collisions was also tested in several different conditions.

6.2 Future Work

HMI is always evolving and there is always room for improvement. In the scope of the work, there are still issues to improve with regards to the IMs. It would be good to further extend the number of IMs to include a diverse set of interactions. For instance, exploring the functionalities and possibilities of acoustic-based interactions could really be an improvement point to the application. In the same way, fully exploiting the possibilities offered by the smartphones' cameras could also be beneficial. Cameras can be used to gather gestures and thus offer new ways of interaction.

Another good improvement with respect to smartphone's capabilities, specially, its portability, would be to allow a user to fully plan, save and distribute navigation paths through the application. This way, the application would not only allow direct navigation, but also enable motion planning and broaden the interactions' scope.

With respect to formation control and to solve the original problem that followers lose sight of the other robots quite easily, it would be good to boost the application's interaction with the followers. By this, it is meant that even though the user would still control the motion of the leader and the followers would react to that movement in the same manner, it would be possible to sporadically select and control one specific robot of the formation, for instance, if for some reason it lost sight of other robots or its position was not accurate to perform a certain task.

References

[1] Rui Paulo Pinto da Rocha. *Building volumetric maps with cooperative mobile robots and useful information sharing: a distributed control approach based on entropy*. PhD thesis, University of Porto, 2006.

[2] Sérgio Paulo Carvalho Monteiro. *Attractor Dynamics Approach to Formation*. PhD thesis, University of Minho, 2007.

[3] Daniel José Marcelino. Navegação de Robôs Móveis em Formação. Master's thesis, Universidade de Coimbra, 2016.

[4] Sérgio Monteiro and Estela Bicho. Attractor dynamics approach to Formation Control: theory and application. *Autonomous Robots*, 29(3):331–355, 2010.

[5] Gunnar Johannsen. Human-Machine Interaction. *Control Systems, Robotics and Automation–Volume XXI: Elements of Automation*, page 132, 2009.

[6] Manisha B Bansode and JK Singh. Android Mobile Phone Controlled Wi-Fi Robot. *International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)*, June 2015.

[7] Jussi Suomela and Aarne Halme. Cognitive Human-Machine Interface of workpartner robot. In *Intelligent Autonomous Vehicles 2001 Conference (IAV2001), Sapporo, Japan*, 2001.

[8] C. Parga, X. Li, and W. Yu. Smartphone-Based Human-Machine Interface with Application to Remote Control of Robot Arm. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pages 2316–2321, Oct 2013.

[9] James Cannan and Huosheng Hu. Human-Machine Interaction (HMI): A survey. *University of Essex*, 2011.

[10] M. Tornow, A. Al-Hamadi, and V. Borrmann. Gestic-Based Human-Machine Interface for Robot Control. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pages 2706–2711, Oct 2013.

[11] Hans-Joachim Boehme, Anja Brakensiek, Ulf-Dietrich Braumann, Markus Krabbes, and Horst-Michael Gross. *Neural architecture for gesture-based Human-Machine Interaction*, pages 219–232. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[12] Christopher Parlitz, Winfried Baum, Ulrich Reiser, and Martin Hägele. Intuitive Human-Machine Interaction and implementation on a household robot companion. In *Symposium on Human Interface and the Management of Information*, pages 922–929. Springer, 2007.

[13] L. R. Stephygraph, N. Arunkumar, and V. Venkatraman. Wireless mobile robot control through Human-Machine Interface using brain signals. In *2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)*, pages 596–603, May 2015.

[14] Jan Nadvornik and Pavel Smutny. Remote control robot using Android mobile device. In *Control Conference (ICCC), 2014 15th International Carpathian*, pages 373–378. IEEE, 2014.

[15] F. Zhang, X. Wang, Y. Yang, Y. Fu, and S. Wang. A Human-Machine Interface software based on Android system for hand rehabilitation robot. In *2015 IEEE International Conference on Information and Automation*, pages 625–630, Aug 2015.

[16] Bruno Henrique Andrade Cruz, Josué Fernandes Agnese, Brunno José Fagundes, Marcelo Teixeira Bastos, Rolf Fred Molz, and Jacques Nelson Corleta Schreiber. Desenvolvimento de uma aplicação embarcada em celular visando controle de robô via Wi-Fi. *Revista Brasileira de Computação Aplicada*, 3(1):43–52, 2011.

[17] Stephan Göbel, Ruben Jubeh, Simon-Lennert Raesch, and Albert Zündorf. Using the Android platform to control robots. In *Proceedings book of the 2th International Conference on Robotics in Education (RiE 2011),(Vienna, Austria)*, pages 135–142, 2011.

[18] C. Bisdikian. An overview of the Bluetooth Wireless technology. *IEEE Communications Magazine*, 39(12):86–94, Dec 2001.

[19] M Jidhu Mohan, Prashant Kumar Tripathi, and KV Gangadharan. Smartphone Motion Sensor Controlled Wireless Mobile Robot. *Eighth Control Instrumentation System Conference (An International Conference) CISCON-2011*, 2011.

[20] Gyula Mester. Wireless sensor-based control of mobile robots motion. In *Intelligent Systems and Informatics, 2009. SISY'09. 7th International Symposium on*, pages 81–84. IEEE, 2009.

[21] André Guilherme Nogueira Coelho dos Santos. Autonomous Mobile Robot Navigation using Smartphones. Master's thesis, Instituto Superior Técnico, 2008.

[22] E. Ferro and F. Potorti. Bluetooth and Wi-Fi Wireless protocols: a survey and a comparison. *IEEE Wireless Communications*, 12(1):12–26, Feb 2005.

[23] J. S. Lee, Y. W. Su, and C. C. Shen. A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi. In *IECON 2007 - 33rd Annual Conference of the IEEE Industrial Electronics Society*, pages 46–51, Nov 2007.

[24] Xiao Lu, Wenjun Liu, Haixia Wang, and Qia Sun. Robot control design based on smartphone. In *Control and Decision Conference (CCDC), 2013 25th Chinese*, pages 2820–2823. IEEE, 2013.

[25] Wikipedia. Android (operating system), 2017. [Online: `https://en.wikipedia.org/wiki/Android_(operating_system)`. Accessed 30-June-2017].

[26] Android. Introduction to android, 2017. [Online: `https://developer.android.com/guide/index.html`. Accessed 10-February-2017].

[27] Android. Kotlin and android, 2017. [Online: `https://developer.android.com/kotlin/index.html`. Accessed 15-June-2017].

[28] Wikipedia. Java virtual machine, 2017. [Online: `https://en.wikipedia.org/wiki/Java_virtual_machine`. Accessed 30-June-2017].

[29] Android. Sdk tools release notes, 2017. [Online: `https://developer.android.com/studio/releases/sdk-tools.html`. Accessed 15-July-2017].

[30] Android. Meet android studio, 2017. [Online: `https://developer.android.com/studio/intro/index.html`. Accessed 10-July-2017].

[31] Jet Brains. Intellij idea, 2017. [Online: `https://www.jetbrains.com/idea/`. Accessed 10-July-2017].

[32] ROS. Is ROS For Me?, 2017. [Online: `http://www.ros.org/is-ros-for-me/`. Accessed 18-February-2017].

[33] Wikipedia. Robot operating system, 2017. [Online: `https://en.wikipedia.org/wiki/Robot_Operating_System`. Accessed 1-March-2017].

[34] ROS. Nodes, 2017. [Online: `http://wiki.ros.org/Nodes`. Accessed 20-February-2017].

[35] ROS. Topics, 2017. [Online: `http://wiki.ros.org/Topics`. Accessed 20-February-2017].

[36] ROS. Messages, 2017. [Online: `http://wiki.ros.org/Messages`. Accessed 20-February-2017].

[37] Java. Interface runnable, 2017. [Online: `https://docs.oracle.com/javase/7/docs/api/java/lang/Runnable.html`. Accessed 20-June-2017].

[38] Public Library of Science. Nintendo wii with a new mission: Wiimote as an interface bridging mind and body. ScienceDaily, 2008. [Online: `www.sciencedaily.com/releases/2008/03/080304200905.htm`. Accessed 9-July-2017].

[39] Paul Lawitzki. Android sensor fusion tutorial. Code Project, 2014. [Online: `https://www.codeproject.com/Articles/729759/Android-Sensor-Fusion-Tutorial`. Accessed 15-June-2017].

[40] Jose Collas. Android Sensor Fusion. GitHub, 2014. [Online: `https://github.com/goatstone/AndroidSensorFusion`. Accessed 15-June-2017].

[41] David Geary. Simply singleton. JavaWorld, 2003. [Online: `http://www.javaworld.com/article/2073352/core-java/simply-singleton.html`. Accessed 2-June-2017].

[42] OMRON adept. Pioneer P3-DX, 2017. [Online: `http://www.mobilerobots.com/ResearchRobots/PioneerP3DX.aspx`. Accessed 16-July-2017].

[43] Korbinian Lipp. User experience beyond usability. *User Behavior*, page 13, 2012.

Appendices

A Usability and User Experience Results

The following tables present the results of the questionnaire's answers that were the basis for the results presented in this document.

The classification is in a scale from 1 to 5, where 1 is lowest evaluation and 5 the highest.

Table A.1: Results of the questionnaire's questions regarding the details of the Arrows Mode of Interaction.

Individual #	**Visual Aspect Content Distribution**	**Maneuverability**	**Feedback**
1	4	4	2
2	5	4	5
3	4	4	4
4	5	3	4
5	5	5	4
6	5	5	3
7	5	5	5
8	5	4	4
9	5	4	1
10	5	5	5
11	5	4	3
12	5	5	5
13	5	4	5
14	4	3	5
15	5	5	5
16	5	5	5
17	5	4	3
18	5	3	5
19	5	4	4
20	5	4	5
21	5	4	5

Table A.2: Results of the questionnaire's questions regarding the details of the Motion Sensors Mode of Interaction.

Individual #	**Visual Aspect Content Distribution**	**Maneuverability**	**Feedback**
1	2	3	2
2	5	5	5
3	4	3	3
4	2	4	3
5	5	3	3
6	5	3	3
7	5	3	4
8	4	4	4
9	5	3	5
10	5	5	5
11	5	5	5
12	5	5	5
13	5	5	3
14	5	5	5
15	4	5	4
16	5	4	5
17	5	4	3
18	5	5	5
19	4	4	4
20	5	5	5
21	4	5	5

Table A.3: Results of the questionnaire's questions regarding the details of the Joystick Mode of Interaction.

Individual #	**Visual Aspect Content Distribution**	**Maneuverability**	**Feedback**
1	5	5	5
2	5	5	5
3	4	3	3
4	3	5	4
5	5	4	4
6	5	5	4
7	5	3	2
8	4	4	3
9	5	3	4
10	5	4	5
11	4	3	4
12	5	5	5
13	5	5	3
14	5	4	5
15	3	3	2
16	5	4	4
17	5	5	3
18	5	4	3
19	5	5	5
20	5	4	4
21	4	3	4

Table A.4: Results of the questionnaire's questions regarding the general aspects of the application.

Individual #	Global Design	Navigation between IMs	Changing formation	STOP Button	Usage of App Preferences	Total time until starting interaction	Differences between OMs	UX
1	4	4	3	5	5	4	3	4
2	5	4	3	5	5	5	5	5
3	4	4	4	4	4	4	4	5
4	4	1	3	2	4	5	5	4
5	5	4	3	4	5	4	4	5
6	5	4	4	4	5	5	5	5
7	5	5	4	5	5	5	4	5
8	5	5	5	4	3	4	3	5
9	5	5	3	5	5	5	4	5
10	5	4	3	3	4	5	5	5
11	5	5	5	5	4	5	5	5
12	5	5	5	5	5	5	5	5
13	3	5	5	5	4	5	5	4
14	5	5	4	4	5	5	5	5
15	5	4	5	5	4	5	5	5
16	5	5	5	5	5	5	5	5
17	5	4	4	5	4	4	3	5
18	5	5	5	5	5	5	5	5
19	5	4	4	5	4	5	5	5
20	5	5	5	5	5	5	4	5
21	5	5	5	4	4	4	5	5