Bachelor Degree Project

UNIVERSITY OF SKÖVDE

AUTONOMOUS MEDICAL ROBOT

Bachelor's Degree Project in Production Engineering
G2E, 30 credits
Spring Term 2020

Ana Almer Casino
Miguel Ángel Sempere Vicente

Supervisor: Stefan Ericson
Examiner: Göran Adamson

Abstract

Lack of healthcare assistance is one of the issues that European countries such as Sweden face due to the increase of the ageing phenomenon which leads to a higher demand for personnel in hospitals, basic attendance, and housing. Therefore, a tool is clearly required to fulfil these social needs. The thesis focuses on the elderly but especially on those with reduced mobility and develops a wheeled domestic mobile robot whose objective is to deliver their pills at the right time of the day. The main goal is the implementation of automatic drug delivery. The project presents a selection of the most suitable wheel-based robot, and a study and evaluation of different techniques and algorithms used for indoor positioning, navigation, facial recognition, and a communication interface between the patient and the robot.

The robot used for the development of this thesis was the TurtleBot3 Burger and was evaluated upon a map created using the Hector SLAM method. Results showed that the Bluetooth technology (iBeacon), as well as the trilateration algorithm, are suitable choices for detecting a human in an indoor environment; a successful average drift error of 0.83 metres was obtained for indoor localization, and further results showed the facial recognition system achieved an accuracy of around 90%. It is concluded that the robot is capable of reaching and identifying the patient in an indoor environment, and so this project can be considered as the first step of implementation of a totally autonomous medical robot ready for domestic use.

Certification

This thesis has been submitted by Ana Almer Casino and Miguel Ángel Sempere to the University of Skövde as a requirement for the degree of Bachelor of Science in Production Engineering.

The undersigned certify that all the material in this thesis that is not our own has been properly acknowledged using accepted referencing practices and furthermore, that the thesis includes no material for which we have previously received academic credit.

[illegible] [illegible]

Miguel Ángel Sempere *Ana Almer Casino*

Skövde 2020-06-01
School of Engineering Science

Acknowledgements

First and foremost, we would like to thank our supervisor Stefan Ericson for his guidance and for sharing our joy and motivation during this ambitious project.

Besides, we wish to express our sincere gratitude to the University of Skövde for providing us with all the necessary material used for the development of the project.

Finally, we cannot forget our families. Their unconditional support made this Erasmus experience possible and, thank to them, we have become the engineers we are today.

List of Figures

List of Tables

Nomenclature

LiDAR	Light Detection and Ranging.
ROS	Robotic Operating System.
SLAM	Simultaneous Localization and Mapping.
WMR	Wheeled Mobile Robots.
IPS	Indoor Positioning System.
IR	Infrared Radiation.
LASER	Light Amplification by Stimulated Emission of Radiation.
TOF	Time of Flight.
MCU	Micro Controller.
IDE	Integrated Development Environment.
SBC	Simple Board Computer.
RF	Radio Frequency.
RFID	Radio Frequency Identification.
UWB	Ultra-Wideband.
TOA	Time of Arrival.
AOA	Angle of Arrival.
TDOA	Time Difference of Arrival.
MEMS	Micro-Electro-Mechanical Systems.
IMU	Inertial Measurement Unit.
CVG	Coriolis Vibratory Gyroscopes.
IrDA	Infrared Data Association.
IoT	Internet of Things.
ROS	Robot Operating System.
MATLAB	MATrix LABoratory.
API	Application Programming Interface.
OpenCV	Open Source Computer Vision.
LDS	Laser Distance Sensor.

Table of Contents

1. Introduction

Over the years, robots and artificial intelligence have been introduced into the daily life of most of the world-wide population. Researchers keep finding new ways in which they can improve and integrate their technology leading to new discoveries that push us towards a future where the majority of tasks are done by robots and not by humans.

The use of automation in the industrial world has a great impact on factories due to the replacement of unskilled labourers with new machinery [1]. A new concept of technology appears when robotic experts start to focus on the medical field. Many believe that autonomous robots could be of great help for people with high-skill careers, such as doctors. Duties or even operations could be performed by robots and be able to provide better diagnostics, safer surgery, shorter waiting times, and reduced infection rates for everyone.

1.1 Problem statement

Ageing in Europe is a phenomenon defined by an increase in life expectancy and a decrease in the fertility rate across Europe. In Sweden, about one in five people is 65 or older [2] and it is expected to increase in the coming years.

The elderly in Sweden have the right to receive care in their own homes although special housing service is available. However, most elderly people prefer to stay in their own house for as long as possible [3]. This nursing home service for old people is one reason that Sweden is considered to have one of the best healthcare systems in the world, but due to ageing population, people need to wait for many days in order to be able to see a doctor or a specialist [4] as there are not enough personnel available to cover the job in hospitals, primary healthcare, or houses.

The increasing need for personnel is focused especially on people with reduced mobility, who require help from health care assistants for longer periods of time, generating longer queues. There is therefore a need for a tool to compensate for the lack of personnel and the wish for the elderly to prolong their independent lives.

Therefore, the project focuses on the development of a wheeled domestic mobile robot that automatically delivers the patient's drugs at the right time of the day. The patient would be provided with a fixed storage station where the tablets are refilled by the assistant only when the hospital receives a signal informing that the system is starting to run out of them. The robot would be able to locate the patient, and whenever it is time to deliver the pills, it would go to the storage spot, pick up the pills, and take them to the patient. Then, the robot would return to its charging station ready to deliver the next dose. If the patient knows beforehand that he/she is not going to be at home, the pills will have to be personally taken from the storage and a message in a bracelet would show up remotely whenever it is time to take them.

The robot would be able to reach different heights and adapt to different scenarios such as the kitchen table, bed, bathroom…. As an additional feature, the robot would have a visible and accessible button to allow patients to call the hospital directly and so be able to ask certain questions or advise them about any issue they could possibly have.

1.2 Objectives

The objective is to successfully develop a robot that will be able to reach the patient by travelling around the typical home environment without colliding with any possible human/pet moving nearby or dodging around. The robot must be able to also recognise the patient once it has reached the desired destination as well as communicate with the patient.

It is important to ensure precise results as the majority of people using the robot will be disabled and they will not be able to reach the robot. Therefore, the robot has to be able to perform the project's challenge of indoor navigation and indoor positioning as well as possible. To obtain the best possible results, an evaluation of different wheeled mobile robots will be completed in order to choose the most suitable robot, and a wide research into and evaluation of different methods will be done in order to implement the most appropriate ones. Regarding the robot's ability to recognise and communicate with the patient, an appropriate algorithm is going to be used and convenient interfaces will be explored.

The aspects that are going to be fulfilled for the project are:

- Select a suitable wheel-based robot for indoor use.
- Localization and navigation of the robot with no collisions (obstacle avoidance).
- Identify the patient's identity and indoor location.
- Communication with the robot through an interface.
- Evaluate robot performance.

1.3 Delimitations

The focus will be mainly on the indoor navigation and localization area, therefore the robot will not include the actual physical component that would be responsible for the drug delivery. Furthermore, a dorm environment is going to be used for the robot evaluation due to the little options available. This means that results obtained may not be expected when testing the robot on a different environment conditions as there are many factors which are not considered.

1.4 Sustainability

The most common definition of sustainable development was stated by the Brundtland Commission in 1987 [5], who introduced it as: "*Sustainable development is the development that meets the needs of the present without compromising the ability of future generations to meet their own needs*."

In other words, this implies that development must take into consideration the planet, in order to cause less damage to the environment and ensure the best scenario for future generations. Achieving sustainability means finding a balance between economic, environmental and social factors in equal harmony [6]. These three aspects will be analysed in the following paragraphs, as well as the impact of the present project on each one of them.

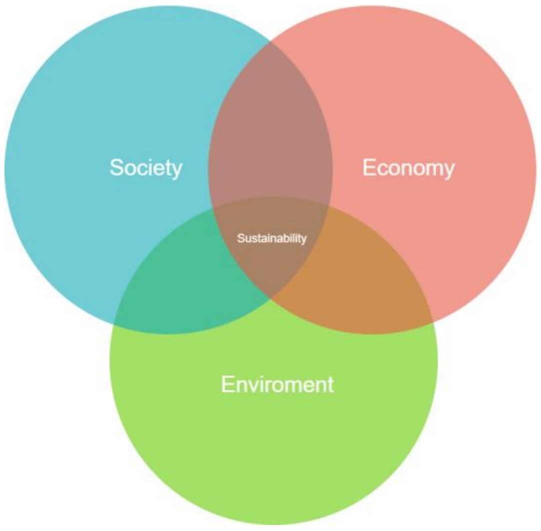


Figure 1. Sustainability Venn diagram

Social sustainability is achieved when the social relationships, systems and processes are focused on ensuring good quality and healthy future communities. This robot is destined for domestic use; although it may still not be socially accepted, especially in generations that grew up with no or little technology integrated into their daily lives, the interest in this type of robots keeps growing as people are noticing that they can turn into a significant help with daily repetitive tasks. They can be classified as social robots due to the possible human-machine communication and can also provide company, which can be of high benefit to certain groups of people. Medical robots can be beneficial and helpful to society and can change the way of taking care of people.

Economic sustainability requires optimal and responsible use of the existing resources in order to obtain an economic benefit. Research into different robots has to be carried out in order to choose the one that both ensures a satisfactory solution and fulfils the demand for a cheap and affordable product. The fact that the robot includes a versatile control board and the common but effective sensors for navigation, localization and vision makes it suitable for many different future applications. Medical robots will also reduce the economic impact due to less need for healthcare assistants to visit the patient's apartment, reducing both transport and personnel costs.

Being environmentally sustainable means to ensure that the consumption of natural resources takes place at a sustainable rate. The key is to find a balance between resource consuming and natural resource regeneration, and to avoid unnecessary waste. The environmental aspect of the robot is of great importance amongst the population, so the amount of energy needed to function must be considered when choosing an ecologically sustainable and suitable robot. The fact that the use of the robot will result in a decrease in the number of healthcare assistants travelling from one house to another may generate interest and acceptance as people are increasingly concerned about climate change and CO2 emissions.

1.5 Overview

This thesis is divided into 8 chapters, a list of figures and nomenclatures, and the reference list. Chapter 2 describes the methodology used for this project. Chapter 3 includes the frame of reference, in which the main relevant concepts and tools for this project are described. Chapter 4 encompasses the literature review research done to motivate the choices taken for the thesis. Chapter 5 describes how the project will be undertaken, what it is going to be included, and the way it will be developed; Chapter 6 is where the tests used to evaluate the performance of the different features of the robot are described. Chapter 7 shows and analyses the results obtained from the previously specified tests, and in Chapter 8 the results obtained are summarised. The conclusions from the whole project are laid out in Chapter 9. Finally, Chapter 10 looks at the possible future work that could be done continuing the research and outcome obtained from this project.

2. Frame of Reference

The important concepts and definitions of the related existing theory need to be studied and explained in order to ensure the correct understanding and development of the project. Having a satisfactory tentative design that achieves the proposed objectives can only be reached by evaluating and selecting the best path to follow on the project.

2.1 Wheeled mobile robots

A mobile robot is an automated machine capable of autonomous navigation to any given location. They move around the space by using intelligent systems for path following along with the sensorial integration to capture information about the robot's surroundings. A mobile robot is made up of actuators and the sensors (external or internal), and they are classified as aerial, marine, or ground. Ground robots are the best option for indoors, especially those which are wheel-based. Wheeled mobile robots (WMR) are easy to design and implement, practical for robots that require speed [7], and its stability allows the robot to be suitable for carrying objects. From the viewpoint of control, less control effort is required, owing to their simple mechanisms and reduced stability problems.

2.1.1 Robot structure

It is important to have a clear idea of the application and environment of the robot to implement one type of robot structure or another. Both aspects determine the project and impose restrictions [8], such as the dimensions of the robot. Moreover, the structure shape influences the robot's performance. Robots with a square shape are more likely to get trapped by an obstacle than those with a cylindrical shape, and would have more difficulty manoeuvering through a narrow space [9].

2.1.2 Wheel types

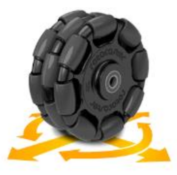
It is important to acknowledge the effect that each wheel type has on the kinematics of the robot. Narrow wheels are commonly used for applications in which high speed and low energy consumption is needed, whilst wide wheels are better for applications that require grip [10]. Together with the choice of wheel geometry, the choice of wheel types must be considered when determining the most suitable mobile robot. In this section, the four main different types of robot wheels will be explored.

a) Standard / Fixed Robot [11]

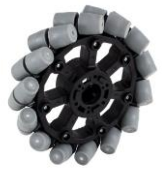
b) Centred Orientable Wheel [7]

c) Off-centred Orientable Wheel [7]

d) Ball Wheel [12]



e) Omni Wheel [13]



f) Mecanum Wheel [14]

Figure 2. Wheel types

Standard wheel. With two degrees of freedom and a traverse front or reverse movement. The centre of the wheel is fixed to the robot chassis, and the angle between both the chassis and the wheel plane is constant. This type of wheel is the most used in WMR's as they are attached to motors in order to drive and guide the robot [7].

Orientable wheel. It could be either a centred orientable wheel or an off-centred orientable wheel. Centred orientable wheels allow a plane motion of the wheel with respect to the vertical axis which passes through the centre of the wheel. On the other hand, off-centred orientable wheels (also known as castor wheels), are orientable with respect to the frame. They avoid lateral slipping if the wheel is not properly headed [15].

Ball wheel. It is an omnidirectional wheel composed of a spherical metal inside a holder, which means that it has a 360^o freedom and so it is suitable for balancing a robot. They are also referred to as "Castor ball wheels" but with the disadvantage that they are not ideal for uneven and dirty surfaces [7]. They also use more power than any other wheel as they have higher traction.

Omni wheel. It is the best option for a robot that requires multi-directional movement. They differ from the rest of the wheels by having passive rollers attached around the centre wheel. They have very low traction [7] as they move in any direction. The axis of smaller wheels is perpendicular to the bigger one. This means that the robot will be able to move parallel to the wheel's axis. Omni wheels can be used to both guide and drive a robot. There is another type of Omni wheel called the Mecanum wheel. The main characteristic of this one is that the roller is attached 45^o around the bigger wheel instead of 90^o.

2.1.3 Wheel configuration

There are many mobile robots with different wheel placements. It is important to have a clear idea of the purpose of the robot to implement the most suitable wheel configuration according to the environment used and tasks to be completed, as well as the initial and operational costs of the robot. This will then lead to a suitable choice for the robot's design. In this section, the robot's wheel configuration will be analysed according to the number of wheels.

One wheel. Robots with one single wheel are the most unstable and have no dynamic control needed to provide balance to the robot. This includes spherical robots, that may be considered as single-wheeled robots. Additional mechanisms implemented to balance the body are required, and control is challenging. This is the reason why single-wheel robots are not the most popular option in practical applications. An example of a robot with one single wheel can be found in Figure 3.

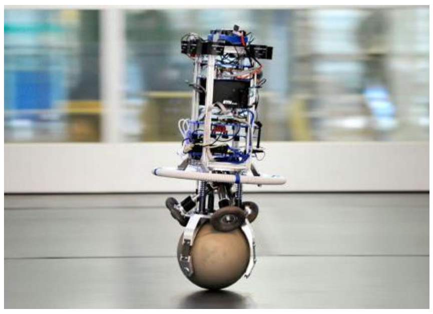
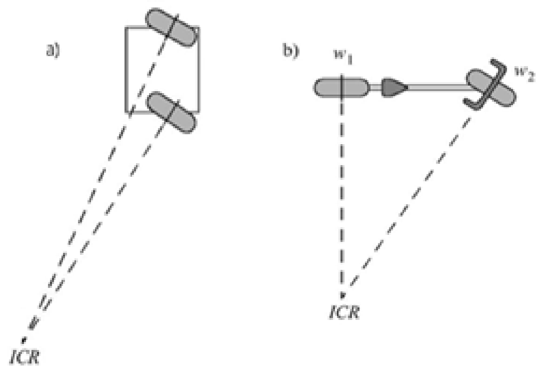


Figure 3. Single-wheeled balancing robot [16]

Two wheels. Two main types of robots use two wheels. Figure 4b shows a bicycle-type robot. The front wheel is commonly the steerable wheel and the rear wheel is driven [17]. This configuration has the advantage of allowing the robot to have a small width, and that no balancing mechanism is required as dynamic stability increases proportionally to the robot's velocity. On the other hand, this configuration is not commonly used due to its inability to keep its posture when standing still.

Moreover, Figure 4b shows an inverted-pendulum-type robot. It consists of a differential drive robot. This configuration allows the robot to have a smaller size than the ones that use three or more wheels and provides static stability when the centre of gravity matches the wheel axle. However, control effort is always required for dynamic balancing.



Bicycle-type robot

Inverted-pendulum-type robot

Figure 4. Two-wheeled configuration

Three wheels. Robots with three wheels are the most used due to their simple structure and stability. As Figure 5 shows, six design examples are popularly used. However, many different designs depending on the choice of the individual can be encountered.

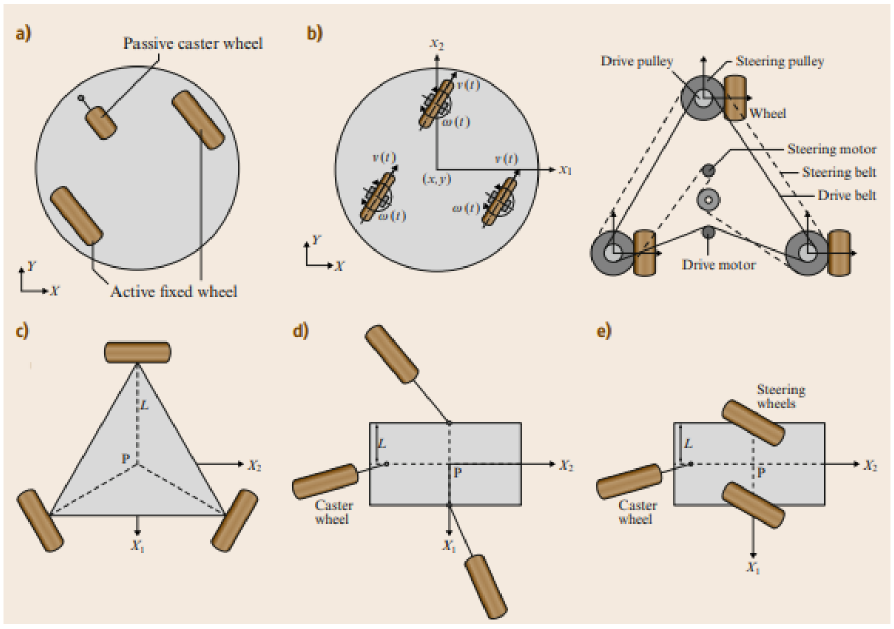


Figure 5. (a) Two-wheel differential drive, (b) Synchronous drive, (c) Omnimobile robot with Swedish wheels, (d) Omnimobile robot with active caster wheels, (e) Omnidirectional robot with active steerable wheels [17]

The two-wheel differential drive is a commonly used configuration. It is composed of two active fixed wheels and one passive caster wheel. The main advantage is the simple mechanical structure, simple kinematic model, and low fabrication cost [17]. However, only bidirectional movement is available.

For synchronous drive, wheel motions are actuated synchronously so they will have the same orientation. This robot will be able to achieve omnidirectional movement by the use of only two actuators, but the mechanical structure will be complicated.

The omnimobile robot with Swedish wheels is classified as a three-wheeled holonomic omnidirectional robot. This configuration is simple to construct. Although a simple structure is used for the mechanical structure of actuators [17], Swedish wheels produce vibrations due to the discontinuous contact with the surface. This means that extra work is required to design a solution to this problem mechanically.

Omnimobile robots with active caster wheels include two active caster wheels or more that use conventional tyres. This means that the previous problem which described Swedish wheels and the vibrations produced can be solved. The main drawback of this configuration concerning this project is that if the robot suddenly switches to the opposite direction, a quick change of wheel orientation will take place and therefore it may result in high steering velocities [17].

For an omnidirectional robot with steerable wheels, at least two-centred orientable wheels are needed to achieve this configuration. It is considered a non-holonomic and omnidirectional robot as centred orientable wheels must maintain their orientation according to the direction in which the robot is moving. This type of wheel requires both the driving motor and axis to be attached which leads to possible wiring problems.

Four wheels. Using a four-wheel configuration allows easy control of the robot, good balance and mobility [10]. The car-like structure is the most used amongst the large variety of four-wheel robots. The front two wheels are synchronously steered to keep the same centre of rotation. The greatest advantage of this configuration is that it is stable when travelling even at high speed, but it also requires a complicated steering mechanism to work. Besides, any four-wheel vehicle needs suspension for the front and rear wheels. Those that have two drive wheels may present different configurations. The most used configuration is the rear-wheel-drive, although rear suspension has many constraints and the development of independent suspension has been difficult.

2.2 Development boards

A microcontroller (MCU) is an integrated circuit that can execute a set of instructions allocated to its memory. These instructions must be written using the appropriate programming language through an IDE (integrated development environment). Inside a microcontroller, the three main parts of a computer can be found: the central processing unit (CPU), the memory and the input/output peripherals. Different circuit boards include all the necessary elements to connect the peripherals to the inputs and outputs of the microcontroller [18]. These boards are named as microcontroller boards and they cannot be considered as a computer or as a simple board computer (SBC) due to several attributes that differentiate both. A basic desktop computer or a typical SBC is much more powerful than a simple MCU board. Generally, a computer can run multiple applications at the same time whilst most MCU boards can only run one program at a time. The fact of having an operating system (OS) and the I/O capabilities of the SBCs (HDMI, Ethernet or USB ports) are also differential factors when trying to draw a line separating single board computers and microcontroller boards. The MCUs run the code written on itself and nothing else, they are reliable, very economical and have low-power consumption. By attaching the external hardware to the I/O pins of the board, lots of different sensors and devices can be controlled, the reason why they are often used as a learning tool to take the first steps in electronics.

However, plenty of robotics projects have been already developed using both SBC and MCU boards. Three of the main development boards available on the market will be be explained below, with information about their strengths and weaknesses.

2.2.1 Raspberry Pi 3 Model B/B+

The Raspberry Pi is an SBC developed in the United Kingdom in order to stimulate informatics teaching in schools. It became more popular than expected, being used even in robotics applications. This small computer runs Linux and it is usually used to improve programming skills, nevertheless, the possibilities are almost unlimited. The Raspberry Pi community is huge and the vast quantity of theses and projects from different kinds of technology areas using this board is a significant help. The average price of the Raspberry Pi oscillates around 35€ depending on the model.

Figure 6. Raspberry Pi 3 Model B [19]

2.2.2 Arduino

Arduino is an open-source electronics platform used in multiple projects. The main advantage of working with this platform is the easy-to-use hardware (MCU) and software (IDE). The programming language used is a simplified version of C++ and the Arduino IDE is based on Processing, which makes the programming part much easier and easy to understand [20]. The Arduino Uno is one of the most popular circuit boards of the Arduino family, most of them based on ATMEL microcontrollers. The number of projects available on the internet using Arduino boards is massive and the cost of getting one starter kit including the board, jumper wires, sensors, LCD display, resistors, LEDs… is extremely cheap (30€). The price of starting to work with Arduino along with the easy programming and the quantity of free available libraries for fast integration of every type of peripherals encourage the "learn by doing" Arduino way of thinking. However, the power of the board is quite limited, and it may be insufficient for developing a big project.

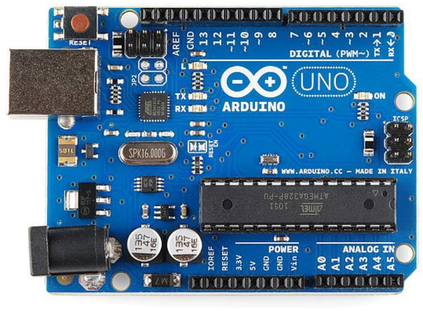


Figure 7. Arduino Uno [21]

2.2.3 ASUS Tinker board

The Tinker Board is an SBC launched by ASUS and it is used in different electronic projects and some mobile robots available on the market. Online benchmarks claim that the Tinker Board is two times faster than the Raspberry Pi 3 Model B and, in fact, ASUS' board physical layout is designed to be compatible with it, which means that both of them have identical pin distribution and look very similar [22]. The price of the board is more expensive than its competitors (80€).

Figure 8. Asus Tinker Board [23]

2.3 Sensors

An exteroceptive sensor is a module that detects changes in the external environment, communicating with other devices and sending them the raw information of the measured magnitude. Sensors are an emulation of the human senses. They allow the robot to "see", "feel", "hear"... with these electronic devices, different kinds of magnitudes that humans are unable to perceive can be measured, corresponding more or less to the real value of the sensed object depending on the quality of the measurement and the possible interferences that can be corrected using filtering algorithms. Proprioceptive sensors provide information about the internal values of the system.

The sensors equipped on the robot will determine its capabilities. The main sensors that most mobile robots use to obtain information about its environment are distance sensors, wheel encoders, image sensors and orientation sensors.

2.3.1 Wheel encoders

These types of proprioceptive sensors are used to measure the internal values and state of a mobile robot. The wheel encoders are a type of wheel/motor sensor and they convert the motion into an electrical signal that can be interpreted by other devices to obtain a position, number of revolutions, speed or the rotation sense. The encoders can be either absolute, which provide a digital code of N bits that represents the absolute value of the position of its axis, or incremental, which provides a pulse signal as the axis goes forward or backward representing the position variation. Absolute encoders do not have important advantages over the incremental ones apart from the direct position measure, with no need to count the pulses to obtain it. They present considerable drawbacks; for high-resolution encoders, a large number of bits are required (N from 8 to 16) and the manufacturing complexity increases as well as the price. Optical incremental encoders have become the most popular device for angular speed and position measuring. The resolution is measured in CPR (cycles per revolution) and this value can range from 2000 CPR to 10000 CPR depending on the application [24].

Figure 9. Incremental encoder (Omron E6A2) [25]

2.3.2 Distance sensors

Ultrasonic sensors provide distance measurements with no contact. Depending on the model, the measurement range can cover up to 5 metres with precision from 0,5 mm to 25 mm. The sensor sends a high-frequency pulse and waits for the echo, the distance value can be obtained either using the TOF (time of flight) by measuring the time it takes the sound wave to hit the object and come back or using the Doppler effect by measuring the frequency difference between the received and the emitted waves. Ultrasonic sensors are common on indoor use and cheap applications, but some of them are designed to operate under aggressive environments outdoors and therefore they have IP protection (e.g., water, dust).

Figure 10. Ultrasonic distance sensor (Parallax PING) [26]

The principle of IR light and LASER measurements is very similar to the one used in ultrasonic sensors. The sensor sends a light beam instead of a sound wave and waits for it to be reflected. The IR light sensors are cheaper, but the LASER ones are more accurate. The different methods used to obtain the distance are TOF, triangulation and phase comparison. Triangulation method works well for short distances (0,5-1m), whilst phase comparison allows high resolutions (0,1mm) or long-distance measurements (30-100m) [27].

Figure 11. Infrared distance sensor (SHARP) [28]

Based on the LASER measurements, 2D and even 3D rangefinders can be assembled. These LASER scanners are usually called LiDAR (light detection and ranging). LiDAR is an optical technology that allows determining the distance to an object from a laser transmitter using a pulsed laser beam. The main improvement of LiDAR sensors, in comparison to other distance measurement methods, is that it sweeps and scans the environment around itself with a wide scanning angle (usually 270° or 360°). This means that the LiDAR can calculate the distance from the sensor to the surrounding objects almost instantly and a map of the environment can be easily figured [29].

Figure 12. 2D LiDAR (SLAMTEC A3M1) [30]

The LiDAR sensor is used on most self-driving cars due to its good accuracy and longer range compared to conventional ultrasonic sensors. However, LiDAR's have some limitations that could cause trouble when moving through a map obtained with this sensor and so navigation should be done along with the information of other "obstacle avoidance" sensors. Different types of objects could not be detected. Transparent materials will not reflect the laser beam, letting it through. Some black surfaces may absorb part of the light received and create problems in the measurement. Also, if a mirror is not placed perpendicular to the laser beam, the laser will be deflected away from the LiDAR [31].

2.3.3 Image sensors

Vision is the most powerful human sense. An image sensor gathers the necessary information to compose an image. A camera has a sensor image that captures a view of a 3D real-world scene and projects it into a 2D image. Nevertheless, a 3D scene cannot be reconstructed with only one image; two or more cameras (stereo pair) are needed to capture the scene from different points of view. In order to obtain a 2D image a sensor composed with multiple photoelectric detectors which capture light points is used. There are two different technologies for image sensors:

- CCD (charged-couple device): the photoelectric detectors generate and store electrical charge when receiving light. The higher the intensity of received light, the more charge they store. Depending on the camera frequency, the charge of every detector is transferred in series to a unique output signal.

- CMOS (complementary metal-oxide-semiconductor): each detector directly converts the light intensity into electrical voltage. The higher the intensity of received light, the highest output voltage.

Both CCD and CMOS cameras are passive sensors, which means they measure the ambient energy entering the sensor [32]. A colour camera can use three image sensors for the basic colours: red, green and blue (RGB). However, these 3-sensor RGB cameras are expensive and most of the conventional colour cameras use only one image sensor with a colour matrix filter (Bayer filter) placed above it. The quality of the image obtained is lower, but the camera is simpler and cheaper [33].

Figure 13. RGB camera module (OV4682) [34]

2.3.4 Orientation sensors

These kinds of sensors are mainly based on MEMS (Micro-Electro-Mechanical Systems): microscopic devices with mobile parts. An IMU (Inertial Measurement Unit) are electronic devices that combine different orientation sensors in a single circuit. The three following sensors described are used for the robot control and balance and are usually the ones integrated into the IMU circuits.

Accelerometer

Measures linear accelerations (*m/s^2*) using Newton's first and second laws. An accelerometer is affected by Earth gravity (*$9.81m/s^2$*) on the planet surface. If no other forces are acting on the sensor, this fact can be used to measure the orientation of the sensor analysing the *g* vector components on each axis of the device. The advantages of the accelerometer are that the measures of acceleration are absolute, and the orientation values obtained from the *g* vector present no integration errors. On the other hand, they are very sensitive to vibrations and therefore the measurements present a lot of high-frequency noise. It is not adequate to determine the speed or position from the acceleration value due to the error generated when integrating. Integration is based on adding values, which cause notorious errors in the long term when accumulating the small measurement errors or noise [27].

Gyroscope

The gyro is a mechanical device that measures changes in the orientation of the sensor, not absolute values of orientation. The MEMS gyros are based on the Coriolis effect and are named as Coriolis vibratory gyroscopes (CVG). CVGs capture angular speeds (*rad/s*) and, in order to obtain rotation angles, integration is required. As previously stated, integration leads to measurement errors in the angles of rotation. Nevertheless, gyros present many different advantages; they are fast and offer many measures per second, they answer well to abrupt changes, are immune to noise and quite accurate on short-time measurements.

Magnetometer

A compass is a magnetometer that measures the magnetic field of the Earth on a surface. The orientation referred to the magnetic north of the Earth can be calculated from the magnetic field measures using trigonometry. Most MEMS magnetometers use Hall sensors and provide noisier measures if compared with the gyros or the accelerometers since compasses are very sensitive to electromagnetic and magnetic fields and nearby metallic objects. It is necessary to perform an initial calibration [27].

2.4 Localization

Localization systems can estimate the position of an object from a specific reference point. In the past decade, robot localization has received a major interest and significant advances have been made. There are many methods to determine a robot´s position, however, it still presents a variety of difficulties depending on the environment. There has been an increase in the demand for accuracy on indoor positioning, and this has led to there being more interest in this research area. This means that many solutions have been proposed, most of them using existing technologies to focus on the problem of the position determining [35]. The following section will focus on indoor localization and will explore different technology classifications.

2.4.1 Indoor positioning

Indoor Positioning Systems (IPS) make use of sensors and communication technologies to locate objects in indoor environments [36]. An estimation of the target object location is obtained from the data collected by a set of sensors or sensing devices. This system can report the estimation either as a symbolic reference (e.g. Bedroom) or as a coordinate-based reference and can be implemented in many areas. IPSs are ideal for private home applications, aiding the disabled and the elderly in their daily tasks, providing item detection and tracking, and medical monitoring.

Indoor positioning technologies

Choosing the appropriate indoor positioning technology is crucial to ensure valid results and a correct balance between performance and complexity of IPSs. Throughout the past years, indoor positioning technologies have been classified in many ways due to the exhaustive investigation that was taking place.

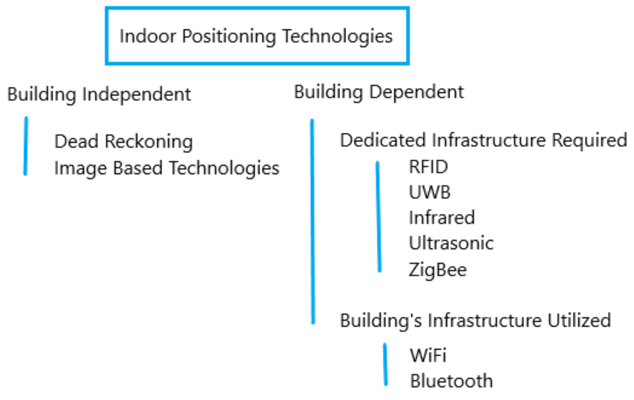


Figure 14. Classification of indoor positioning technologies

As Figure 14 shows, technologies are classified into two main classes: building dependent and building independent. This means that those technologies which are building dependent, depending on the infrastructure and characteristics of the building the system will be implemented in and can be further classified as indoor technologies that need a specific infrastructure or as indoor technologies that use the building's existing infrastructure. On the other hand, in building independent technologies, special hardware in a building is not required. For example, in dead reckoning, an object can determine its position by using its internal data such as past position, speed, and direction in which it moves [37]. Further detail of relevant technologies for this project is given in the following.

1) **Radio Frequency Identification (RFID).** The RFID technology is widely used in large systems to both locate and keep track of objects automatically. It is a wireless communication method between a tag and a reader that uses radio waves. RFID tags emit radio signals which are received and read by RFID readers and the other way around [35]. The objects that are needed to be located are attached to these RFID tags which consists of a microchip to store data, and a radio antenna. Furthermore, RFID readers contain different components in order to connect to a server [38].

2) **Ultra-Wideband (UWB).** This technology englobes any RF that has a bandwidth greater than 500 MHz or than 25% of the main carrier frequency[1]. UWB utilize a large bandwidth from the RF spectrum to communicate. This means that it is capable of sending more information than other technologies such as Bluetooth and that transmitters need only a small amount of energy to transmit [39]. In positioning, Time of Arrival (TOA) and Time Difference of Arrival (TDOA) are used to measure the distance between the object and the reference point.

3) **Ultrasonic.** Ultrasound waves have a high frequency and are commonly used to detect objects or measure distances, as they do not interfere with electromagnetic waves and have a short-range [38]. Ultrasound pulses are emitted to receivers and so measure distances or ranges between devices by using TOA.

4) **Infrared (IR).** Infrared wireless technology is used in devices that transfer data through IR radiation which means that operates with short and medium ranges. Communication interception is difficult since line-of-sight[2] is required and it has a short transmission range. Although other systems operate in diffuse mode and are able to function when the source and destination are not directly visible to each other [40].

5) **Zigbee.** Zigbee is a wireless technology focused on addressing the need for low cost and low power IoT networks [41]. It is composed of a small Zigbee node consisting of a multichannel two-way radio and microcontroller with a low cost and complexity. This ZigBee technology can be implemented for indoor positioning by communicating and coordinating with other nodes nearby. There are two different types of devices used for Zigbee nodes: Reduced Function Device (RFD) and Full Function Device (FFD) [38].

[1] Stated by the FCC (Federal Communications Commission).
[2] Straight line along which an observer has unobstructed vision.

6) **Wireless Local Area Network (WLAN).** It is a wireless technology that exchanges information between two or more devices by using high-frequency radio waves and often encompasses access to the Internet. It maintains a network connection as long as the user does not exit the coverage area, usually a house or an office [42]. Available routers in the area in which the system operates in are needed to be listed as if Wi-Fi is going to be used for indoor positioning, it will depend on it [35]. Received Signal Strength (RSS) method is the popular choice for WLAN positioning whilst TOA, TDOA, and AOA methods are more time delay and angular measurement complex.

7) **Bluetooth.** It is a short-range wireless technology used for communication. It uses the principles of device "inquiry" and "inquiry scan". Little power is needed for scan devices to send information to the inquiring devices in order to connect both [43].

8) **Pedestrian Dead Reckoning (PDR).** It is a relative navigation technique which initially uses a known position, and successive position displacements are added up [37]. A higher updating speed means that linearly growing position errors can be contained within predefined bounds. In other words, this technique works by estimating the speed and direction in which the robot moves, and it has been applied to many navigation problems.

Indoor positioning techniques

In order to counterbalance some limitations of single positioning technologies, it is convenient to combine for the same application, one or more techniques [44]. Indoor positioning techniques are classified into four classes[3] detailed in this section.

1) **Triangulation**. The triangulation location technique is a method that calculates a position based on the distance between two reference points and the angle between these points to a specific object. It uses geometric properties of triangles to find the location of the object [45]. Triangulation comprises both angulation and lateration. Lateration uses the distance from multiple reference points to measure the distance to the object [44]. For a two-dimension measurement, the distance from 3 different points is required, whilst for a three-dimension measurement, 4 points are needed. There are two ways to measure the distance between the reference points mentioned above; time of flight and attenuation [35]. Time of flight consists of measuring the amount of time that the emitted signal takes to travel from the object to the reference point at a known velocity. Attenuation consists in measuring the gradual reduction of the intensity in the signal while transmitting. With the use of a given function that correlates both attenuation and distance, the strength of emission is measured when emission reaches the point of reference to estimate the distance in between.
   Angulation determines the position and distance of an object by the use of angles relative to different reference points. For a two-dimension angulation, two angles and a length are required, whilst for a three-dimension angulation, one azimuth measurement is also needed.

[3] Gu et al., 2009.

2) **Proximity**. This technique detects the presence of an object with respect to a known position. The object's presence sensing can be divided into three different categories. The first approach detects physical contact by using sensors that collect pressure and touch data, and capacitive field detectors [44]. The second approach monitors wireless cellular access points when the system moves inside the range of at least one of them. Finally, the last approach observes automatic ID systems[4]. The device may scan the label, interrogate the tag, or monitor a transaction until it has a known location. This is when the location of the object can be inferred [35].

3) **Fingerprinting.** It is a technology-based on finding the match of fingerprint of a signal such as RSS, which is location dependent. Two stages determine the fingerprinting technology: offline stage and online stage. Offline stage builds a radio map to collect and store radio signatures. This data is then used to make a further comparison and matching. In order to build the radio map, the area of interest is divided into different cells and for a certain period of time, RSS values of radio signals[5] are collected and stored in the radio map [44]. The online stage estimates the object's location by using both observed signal strengths with the previously collected information. Signal strengths are received when the user sends a location query and its corresponding RSS fingerprint. At this point, matched fingerprints and corresponding locations are returned back once the fingerprint database is retrieved. The big challenge comes when the signal strength could be diffracted or scattered while travelling through an indoor area [35].

4) **Vision analysis.** In many applications, multiple cameras are used to cover the maximum area possible and take real-time images. However, it is possible to estimate a location from the images collected from one single point. Tracked targets are then identified from these images and are searched for in the pre-measured database to estimate the position [44].

Indoor positioning algorithms

This section describes the existing algorithms for positioning objects and the robot itself in an indoor environment.

1) **Time of Arrival (TOA).** This algorithm measures the time of flight taken for the signal to reach the receiver from the transmitter. Distance is then directly calculated as signals travel with a known velocity. Several beacons are the receivers and are synchronized with the transmitter following a precise time source. Firstly, the distance between both devices is measured based on the speed and the measured time of the signal. Secondly, the triangulation technique is applied in order to estimate the object's location [35].

2) **Time Difference of Arrival (TDOA).** The difference with TOA algorithm is that TDOA measures the propagation time difference between the transmitter and different receivers, and so

[4] ID systems such as credit card point-of-sale, computer login histories, and hand-line telephone records.
[5] Signals such as Wi-Fi signal strengths transmitted by Aps (multiple Access Points).

measures the distance from the time difference. The time for which the signal is sent is unknown for nodes receivers. This is because they rather use the intersection of many hyperbolic curves produced by each difference measurement of arrival time, which allows specifying the possible locations of the transmitter. A limitation of this algorithm is that only one intersection is the representation of the real location, therefore previous knowledge is needed in order to eliminate position ambiguity [46].

3) **Angle of Arrival (AOA).** AOA works with measurements of two or more angles. It consists of a mobile receiver that compares the signal amplitude sent by two or more beacons that have a known position. These signals are evaluated by multiple antennas present in the mobile receiver and therefore an estimation of the position is possible [46], by triangulating the intersection of the angle line from each signal source.

4) **Received Signal Strength (RSS).** RSS can handle radio signals only [46]. It is based on the attenuation of emitted signal strength from the unknown node to be capable of estimating the distance between this node and the reference one. The propagation model or the fingerprinting algorithm are the ones suitable for RSS localization method. Propagation model algorithm (PMA) establishes the model between RSS and the distance: the closer to the access point (AP), the higher value of the RSS. However, it is complex for indoor areas [35]. This is because furniture or equipment may interfere in the signal propagation, causing diffraction for instance.

5) **Kalman Filter (KF).** The Kalman Filter (KF) algorithm provides estimations of unknown variables given the measurements observed over time. It is recursive and it is divided into two steps. The first one is the prediction step, in which estimations of the current state variables are produced [47]. Further measurements are taken and observed along with their amount of error (including random noise) in order to update the previous estimations. To do so, a weighted average is used, giving a greater value for those estimations with a higher certainty. This algorithm is able to run in real-time by the use of present input measurements and the previously calculated state, with no need for past information.

   Extended Kalman Filter (EKF) algorithm is an extended version of the KF as it uses Linear Algebra to update the robot's state and measurements [47]. It adds linearization to the nonlinear state transition model and measurement model. Moreover, using landmarks allows the robot to update its state accordingly. This is because the location of such landmarks is known, so when the robot calculates a low variance of the state estimate and the measurement estimate, it can quickly locate it with respect to the landmarks.

6) **Monte Carlo Localization (MCL).** This method can be also known as particle filter localization as it uses the particle filter technique. The algorithm works by estimating the position and orientation of the robot as it moves and senses the environment [48]. This means that it generates random particles of where it is going to be next, containing a full description of a possible future state. Particles that are inconsistent with the observed environment are

discarded and new ones are generated close to those that appear consistent. Finally, most of the particles would appear on the robot’s location.

7) **Bayesian inference.** As mentioned in [49], Bayesian inference is a method of statistical inference that uses the Bayes’ theorem to update the probability for a hypothesis as more evidence or information is available. For indoor location purposes, a set of possible locations is established. The likelihood of a set is a given location in an observed RSS.

2.5 Navigation

Navigation is the science or art of guiding a mobile robot in the sense of how to travel through the environment[6]. Navigation is one of the most challenging tasks of a mobile robot. The robot needs to have high precision in positioning to ensure a correct autonomous movement of the robot to a certain point[7]. Traditionally, navigation has been understood by determining where the robot is, path-planning to comprehend how to reach the target, and navigation to execute the corresponding trajectory. However, further research on perception and modelling have taken place and so additional features are now involved in robot navigation: surrounding places elements, and structures determination.

2.5.1 Indoor navigation

Since there can be several applications that can benefit from using indoor location service, it has been a very important research area in recent years. Although GPS solutions for indoor navigation are directly discarded, as the reception of these signals inside almost every building is not reliable, companies are bringing innovative solutions to this problem. As a result of this, the robot’s navigation can be achieved by the use of a coordinate-based system, a behaviour-based system, and a hybrid system. Some of the most popular techniques are described below.

SLAM

Stands for Simultaneous Localization and Mapping. It is a navigation technique in which an autonomous robot operates in an a priori unknown environment, using its sensors to build a map of the environment and at the same time, self-locate. It includes various useful packages:

1) **gMapping.** The gMapping package provides laser-based SLAM as a ROS node called slam_gmapping. 2D occupancy grid maps may be created from the laser and pose data collected by the robot.

2) **Hector.** Hector SLAM uses only laser data collected to create the occupancy grid and it is able to work with laser mounted not planar to ground (as required by gMapping) [50]**.**

3) **Cartographer.** This graph-based system provides real-time SLAM in 2D and 3D across multiple platforms and sensor configurations.

[6] McKerrow, 1991.
[7] Beinhofer et al. 2013.

4) **Frontier.** Frontier exploration involves dividing the whole environment into cells [51]. The cells are used by three different classifiers (free, occupied, and unknown) and they are responsible for the representation of the boundary between the mapped and the unexplored areas.

ROS Navigation Stack

It is based on taking information from sensor streams and outputs velocity commands to send them to a mobile base. The use of the Navigation Stack requires that the robot is running ROS, has a tf[8] transform tree in place, and publish sensor data using the correct ROS Message type [52]. It is also required to be configured according to the shape and dynamics of the robot to ensure a high-performance level

Dijkstra algorithm

This algorithm calculates the shortest path between nodes in a graph and therefore ensures an efficient path planning. There are different existing variants for the Dijkstra algorithm but most of them use fixed nodes as their source for finding the optimal path and eventually construct the shortest path tree [53].

Iterative Closest Point (ICP)

The ICP algorithm is based on finding an optimal mapping by minimizing the distance between two corresponding points by the use of translation and rotation methods [54]. 3D data which describes a specific model shape is given to the coordinate system. This model shape is given in a different geometric shape representation; therefore, the system then uses these translation and rotation estimations to determine the equivalence of both shapes via a mean-square distance metric.

2.6 Computer vision

Computer vision is a subfield of machine learning, which teaches computers how to "see", and using the suitable methods, allows them to obtain, process and analyse any kind of content from digital images.

2.6.1 Facial Recognition

Thanks to computer vision, different useful systems have been developed. One of them is the face recognition system, a technology to identify or verify the identity of a person from an image or a video of his face. There are several methods of implementing facial recognition, but generally, they are based on comparing the indicated facial characteristics from a given image with the faces in a database. Compared to other biometric techniques, facial recognition is not the most reliable one. The illumination, noise and expression are critical factors during the face capture and can make the recognition process fail. Nowadays and using the available community documentation, real-time face recognition can be implemented with not many problems using OpenCV, a development board and a camera module.

[8] ROS package.

2.7 Software

2.7.1 ROS

Robot Operating System (ROS) is a framework and operating system used for robotics software development. It is confirmed by a collection of libraries and tools that simplify the development of complex robot behaviour in the wide variety of robotic platforms. ROS provides a good analysis of how the diverse processes communicate; thanks to its modular design and the available packages developed by other users which can be easily implemented, ROS has become necessary for mobile robots’ projects. The main ROS concepts necessary to get started are the following. For further information about ROS concepts and structure, visit the ROS website [55].

Nodes are the processes. Each one of them has a specific and limited scope (e.g. multiply two numbers) so they can be reused for different applications. Nodes are designed to build simple processes in order to make them easier to debug and they are usually written using C++ or Python.

Messages are the information that nodes use to communicate with each other. They exchange this information using a predefined message structure, a node may not send an arbitrary message if it is not well-defined.

Topics are the place where nodes can publish their messages. A node can publish a message on a topic and other nodes can subscribe to it. There is no limit to the number of subscribers or publishers a topic can have, nor to the number of topics a node can subscribe or publish on. As stated above, only the messages with the specified predefined structure can be distributed on a topic.

Services are an alternative method of communication between nodes. Instead of using the publish-subscribe system of ROS topics, a node can request another node to perform a service. This request-response method must be (like messages) well-defined to ensure satisfactory communication.

In other words: nodes represent the processes; the most important fact is that each node has a specific and limited functionality. In a graphical structure, they are connected using ROS topics. A node can post a message on one or several topics and other nodes can subscribe to these topics or several others. The topic is the channel the nodes use to send messages between each other. These messages must be well-defined and, if the node wants to send a message on a topic, it must be of a pre-defined type. For example, an image and a string cannot be published on the same topic. Services are an alternative communication method between nodes [31].

2.7.2 MATLAB

MATrix LABoratory (MATLAB) is a numerical computing environment with its own high-level object-oriented programming language (M) used for matrix manipulation, data and function plotting, algorithm implementation, user interface creation and communication between programs in different programming languages and different hardware devices. MATLAB is used in diverse areas such as machine learning, signal processing, image processing, healthcare monitoring, robotics, mobile networks, artificial vision, communications, control design and much more [56]. The MATLAB official page provides lots of documentation and the official community forum is a big help to solve any possible doubt.

2.7.3 OpenCV
Open-source computer vision (OpenCV) is an open-source software library used for computer vision and machine learning applications. There is no official IDE to use when working with OpenCV, so the user needs to use C, C++, Python or Java to build an application [57]. Due to the vast community and user base, there are many forums where problems can be discussed.

2.8 Communication
In order to allow communication between the robot and the patient, some kind of interface must be implemented. Different ways of allowing human-robot communication are proposed with the aim of, later on, performing an evaluation and selection of the most suitable one for this project.

2.8.1 Button
The most basic interface is to use a set of switch buttons as inputs to send commands to the robot. As many vacuum cleaner robots, the orders can be sent by pressing the buttons attached to its structure. This is the easiest and maybe cheapest way to implement an effective human-robot communication, but it is not the most user-friendly.

2.8.2 Touchscreen
Using a touchscreen attached to the robot's body as a communication interface can be an option. The implementation of the screen input is not so complex, and neither is the price. Touch-screen control is satisfying regarding human interaction since nowadays touch-controlled devices have increased their popularity with the arrival of smartphones.

2.8.3 Voice
Sending the commands to the robot using voice recognition techniques is the most satisfactory way of implementing a user-friendly communication interface. In order to obtain the best results, a speech-to-text API is needed; Google Speech-to-Text, for example, costs money. This interface is maybe the most complex and also the most time-consuming one, but it is an interesting option to simulate a real-world application where having good quality user experience is an important objective.

2.8.4 Gestures
In order to control the robot, gesture recognition can be implemented. This prevents having to touch both the interface and the robot, and so the patient would be able to send the commands to the robot remotely. A camera or a pair of gesture-based gloves are needed to identify the gestures and allow the patient to control the robot.

2.8.5 Remote control
The mobile robot can be controlled remotely using Bluetooth or Wi-Fi protocols. A remote controller or even an Android device can be used to send the commands to the robot. If the Bluetooth capability is not already integrated on the development board chosen, an additional Bluetooth module needs to be attached to the mobile robot order to exchange information with the emitter; these modules are cheap and their integration is well described on the internet. Using this method as a communication interface results in being friendly to the human, who is familiar with controlling TVs and many daily-life devices remotely.

2.9 Suitable robots

In order to make the selection of the most suitable robot for testing the specified tasks, five different options have been proposed:

2.9.1 AlphaBot2-Pi

This robot kit integrates the Raspberry Pi 3 and it is designed for educational purposes. Features like line tracking, remote control (Bluetooth, infrared, Wi-Fi), video and image processing and monitoring and obstacle avoidance can be easily implemented thanks to the Raspberry Pi versatility and the open-source code demos available on the supplier's website [58]. The modular design along with the simple mounting structure makes possible the attachment of an additional LiDAR sensor for indoor mapping. This small and lightweight mobile robot is the cheapest of all the proposed options (~100€).

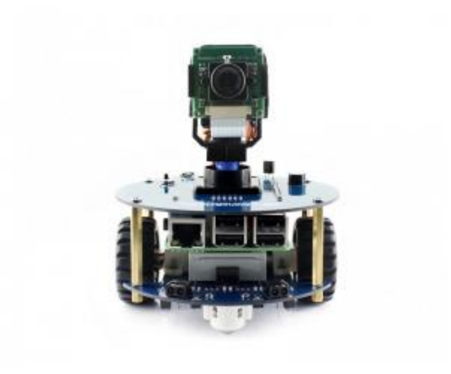


Figure 15. Waveshare AlphaBot2 [59]

2.9.2 SLAM Robot

This four-wheel mobile robot is equipped with Kinect sensors, Lidar and gyro, and can perform tasks like path following, path planning, obstacle avoidance, localization and mapping [60]. It uses Raspberry Pi 3 with ROS, which makes things easier to implement thanks to the abundant information about Raspberry projects. The robot is interfaced with RPLiDAR, which sends the scanned values to MATLAB, for instance, mapping the environment. It lacks a sensor to implement a visioning system, and the documentation available on the internet about this specific robot is null, but the price is low and affordable (~400€).

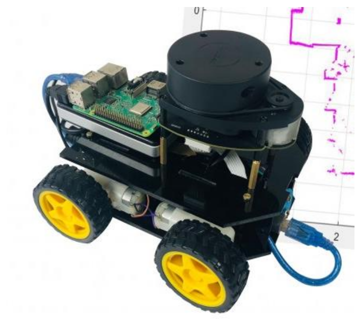


Figure 16. Slam Robot (Pantech Solutions) [60]

2.9.3 Nevon Self Driving Car

The Nevon mobile robot is designed as a prototype for autonomous car driving; the autonomous vehicle is capable of sensing the surroundings thanks to the integrated LiDAR sensor. Obstacle avoidance is already implemented, and the main focus of the company is to offer an automated driven robot where future self-driving cars features can be tested. It uses the ATmega328p Microcontroller, which has to be programmed in C language and lacks documentation from previous mobile robots' projects using it. No camera is included and the backup from the company seems questionable. Nevertheless, the relatively low price (~400€) makes this robot an interesting option.

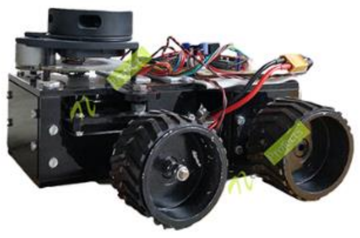


Figure 17. LiDAR-based Self Driving Car (Nevon) [61]

2.9.4 Husarion ROSbot 2.0

This development kit is an open-source autonomous robot, and as the name indicates, it is based on ROS. RGBD camera, IMU sensors and LiDAR sensor are included and the powerful ASUS Tinker board computer is used. The ROSbot is already assembled, using high-quality materials, such as the aluminium chassis and the different components. The company provides the customer with useful tools (Web UI) and a large documentation website about ROS software and the features available on the ROSbot 2.0. The ROSbot 2.0 is quite expensive (~1800€), and even though the quality of this robot is very high in general terms, it includes many unnecessary features that maybe are not going to be used.

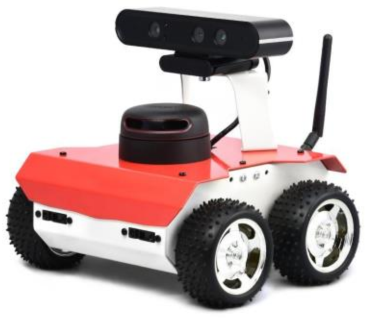


Figure 18. Husarion ROSbot 2.0 [62]

2.9.5 TurtleBot3 Waffle Pi

The TurtleBot is an open-source robot created for education and research purposes. The robot is small and it is possible to be structurally and mechanically customized. LiDAR, encoders on the wheels and camera are the primary needs for mapping and navigating indoors, and this robot has all of them [63]. TurtleBot's biggest strength is the community backup; lots of documentation and projects are available on the Internet, the integration of the sensors included and the installation of ROS software on the versatile onboard Raspberry Pi is well explained on the official page, which allow the customer to quickly start with the development and will ease the implementation of the desired features [63]. Besides, it has a Bluetooth module which allows remote control of the robot. The price of the TurtleBot3 Waffle Pi is moderately low (~1400€), making it a very interesting option.

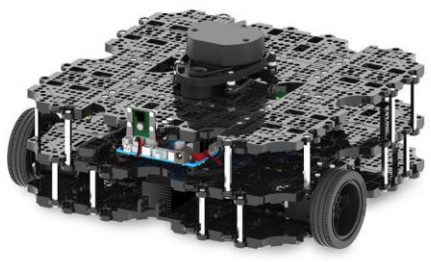


Figure 19. TurtleBot3 Waffle Pi [64]

2.9.6 TurtleBot3 Burger

This mobile robot is the smallest member of the TurtleBot3 family [63] and gathers most of the features existing on the Waffle Pi, costing half of the price (~500€). The documentation and the community backup available when working with the Raspberry Pi 3 is also present in this version of the TurtleBot. However, the structure of the robot is less stable if compared to his "big brother" (Waffle), and it lacks the camera already integrated into the robot frame. The Raspberry camera (~20€) could be attached to the TurtleBot3 Burger without too much trouble, but the balance fact could be problematic.

Figure 20. TurtleBot3 Burger [65]

3. Literature Review

Most of the studied literature related to navigation on mobile robots uses ROS software. According to [66], the main features of ROS are the reusability of the program, the modularization of each program to divide it into smaller parts based on its function, the support of development tools such debugging, 2D and 3D visualization tools, the active community and the construction of an ecosystem in the robotic field. In [29], ROS is used to test LiDAR functionality in order to obtain a realistic map of the environment, knowing the distance to each surrounding object.

In [67], different types of software are analysed and compared to achieve the best simulation environment for an automated vehicle project. Veronika Lappyová [68] uses Gazebo simulator for the implementation of human detection and following algorithm. It uses a LiDAR for obtaining data of the surroundings and a classifier to differentiate between human legs and other objects in an environment with an accuracy of 85%. As seen in [31], this 3D simulator can also be used to import CAD models and perform the specified tasks before testing them in the real world. The TurtleBot3 is used in this project to develop a mobile inspection robot. The author considers it to be the best choice where the ease of development is more important than robustness. Related to the mentioned simulation work, Fiki Jusuf simulates the TurtleBot in [69] using the different ROS included software for auto navigating and mapping around the virtual environment.

Regarding facial recognition applications, the most used software is OpenCV. This open-source C/C++ library is used for many computer visions projects. Plenty of different interesting projects can be developed using this software; from a simple face recognition project [70], where basic features are tested, to an artificially intelligent face detection and investigation system [71], where OpenCV is used to recognize faces of criminals in an image and video stream obtained from a camera in real-time. The researchers insist on the importance of a proper client training in order to get the best results. It is crucial to have a lot of variation of conditions for each person, so the classifier will be able to identify the person in different light scenarios and positions. As stated in [72], it is beneficial to create separate sets of training images for each person (e.g. "Michael_Forward", "Michael_Left") to increase the accuracy. The mentioned projects use the Haar feature-based cascade classifier, where the face detector is trained from a lot of positive and negative images and then used to examine each image to classify it as "Face" or "Not Face".

About the simultaneous localization and mapping (SLAM) problem, a case study on the recursive Bayesian formulation is performed by Bailey & Durrant-Whyte [73]. The main paper focus is on the study of the computational complexity, data association, and environment representation of the method. It is concluded that the SLAM problem presents trouble when working with large-scale mapping, when multiple vehicles are involved, and when working in mixed environments with sensor networks and dynamic landmarks. However, the SLAM method provides a solution for mapping and localizing any autonomous robot, due to the significant development of efficient, consistent, and robust algorithms. Moreover, Pfrunder et al. [74] redesigns a six degrees of freedom LiDAR SLAM algorithm to achieve 3D localization and real-time robot navigation. In this study, high precision SLAM updates are used together with odometric local state robot estimates, as well as a 2D occupancy grid computed from the 3D base map, to achieve a completely autonomous robot that is capable of navigating through a heterogeneous environment. As studied in [75], results obtained of implemented

SLAM algorithms on a radio-controlled car can be observed. The software used is based on ROS, and it also examines Extended Kalman Filter (EKF)-based SLAM, FastSLAM, and GraphSLAM in both theoretical investigations, simulations, and real-world experiments. The thesis proves FastSLAM to be the best SLAM algorithm candidate and that it is successfully implemented through the use of the ROS package gMapping.

In order to achieve robot localization, most of the research papers implement either the fingerprinting or triangulation technique. [76] concludes that although fingerprinting and triangulation techniques are possible approaches to this use case, fingerprinting provides better precision in small buildings. It is also proved that this technique can be performed based on Bluetooth and Wi-Fi if a good filter is used. This filter is needed to remove noise and compensate for possible changes in RSS. Other papers also implement the RSS algorithm. In [77], the trilateration method is implemented and feed with Wi-Fi RSS values. Results show the position accuracy for indoor positioning to be 2-2.5m and concluded that this accuracy can be improved if the number of similar APs are deployed in the system.

The Bluetooth technology is evaluated in many other papers, an example is [78], this paper proposes an algorithm based on separate signal-attenuation models and weighted trilateration using BLE beacons. The experimental results show that the method for obtaining the distance has an accuracy of 2.2m and the method for achieving the positioning has an accuracy of less than 2.4m. Therefore, users' optimal position is precisely determined for different scenarios.

Further research into these two most popular technologies, [79] presents an analysis of indoor positioning systems based on previous surveys. The paper analyses the current status of indoor positioning based on previously published surveys. Although UWB technology is acknowledged to be a successful IPS technology in different surveys, positioning based on Wi-Fi and Bluetooth fingerprinting is found to be the most popular method for indoor purposes.

4. Methodology

This chapter explains the methodology applied to the project. It consists not only of fulfilling the aim of implementing a suitable algorithm for robot indoor positioning and navigation but of a research project. Consequently, knowledge is acquired through the different parts of the methodology used.

Scrum is the research strategy followed. As described in [80], this strategy itself is an iterative and incremental project management methodology or framework for effective team collaboration on complex products. Scrum was initially developed for managing and developing products. Starting in the early 1990s Scrum has been used extensively worldwide to develop software, hardware, embedded software, networks, autonomous vehicles or marketing.

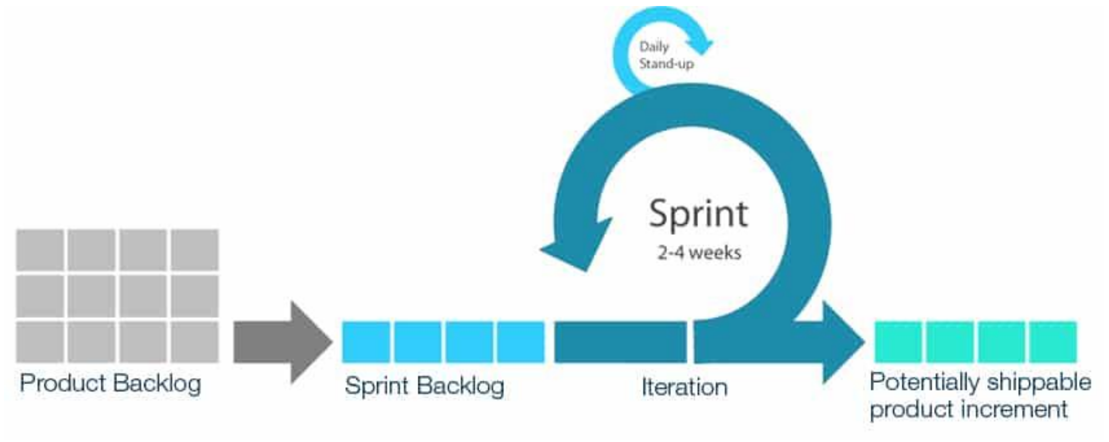


Figure 21. Scrum methodology [81]

Scrum is designed for teams, working as a unit, who divide the work into short actions that should be completed within one month or less. These time-boxed divisions are called Sprints, and they are used to reach specific goals. A new Sprint starts immediately after the previous one, and it is of great importance to always maintain the product in a useable and potentially releasable state once each Sprint is finished. The Scrum methodology diagram (Figure 21) is mainly defined by team roles, events, and artifacts. The artifacts that compose Scrum are the following:

- The **Product Backlog** is an ordered list of everything needed in the product; it comprises all features, functions, fixes, and enhancements that can add value and improve the final result. Requirements never stop changing, so a Product Backlog is a dynamic living artifact.
- The **Sprint Backlog** is the specific list of items taken from the product backlog which are to be completed in a sprint. It is a forecast about the functionalities that will be included on the next delivery and the work needed to accomplish the Sprint goal.
- An **Increment** is the sum of all the Product Backlog items completed during a Sprint, and the value of the increments of all previous Sprints completed so far. The increment must be in useable condition and ready to be released.

The Scrum events that connect the artifacts and create the structure of the methodology are:

- The **Sprint** is the heart of the Scrum methodology. A Sprint is a period of time during which the target objectives are completed and made ready for review. The Sprints of this project are focused on fulfilling specific tasks. Each one is described in their corresponding section below.
- The **Sprint Planning** is an event where the work in the Sprint Backlog is undertaken and the goals to achieve on each Sprint are planned. The planning must be collaboratively done by the entire team, and unforeseen and unexpected events must be considered and so the duration planning should not be too detailed. This event is represented as the Gantt Diagram.
- The **Daily Scrum** is a short daily meeting (15 minutes) where each team member goes through the work done since the last meeting. This optimizes team collaboration and performance and is used to forecast the upcoming Sprint work, analyse the impediments that may be blocking the progress and, if necessary, adapt or replan the rest of the Sprint.
- The **Sprint Review** is a demonstration event for the team to present the work done on the Sprint. The clients give feedback and the work is compared with the expected results to accept or reject it. This can be interpreted as the Final Presentation.
- The **Retrospective** is the final team meeting in the Sprint, it is the opportunity for the team to inspect and determine which processes went well and which did not, and what can be improved in the next Sprint.

5. Development

The development part of this project must be carried out following the Scrum methodology principles. The different objectives targeted at the beginning of the project were identified as separate items, and the development, evaluation, and results part of each one of them was done independently before moving to the next item. There are going to be seven different Sprints in this chapter, some of them were evaluated before release and some others were just addressed to deliver one single item with no need for further testing. Figure 22 shows the eight items that builds up the product backlog. These items are then used in the sprint backlog to define these seven sprints which consists of: robot selection, in order to choose the most suitable robot for the project, both robot assembly and software setup with the aim of verifying the robot functionality, SLAM method to test which method performs the better for our working area, navigation, to test the accuracy of the robot's navigation, face recognition, in order to achieve patient's identification, patient localization, to be able to reach the patient within the given area, and the communication interface, that will allow the patient to communicate with the robot

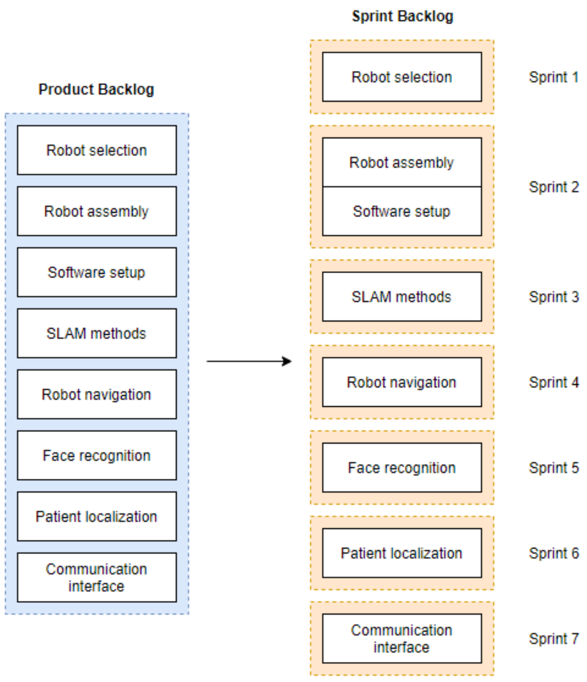


Figure 22 Sprints overview (Scrum)

It is essential to point out that the following subsections below described are written together to help and facilitate the read and understanding of the project structure, but that the development, evaluation and review of each one of them were finished before starting to develop the other tasks. This is, during each iteration (Sprint) the target objectives were completed and made ready for review before starting the next Sprint. Also, an advantage of using Scrum is that if unexpected problems appear and some tasks cannot be successfully finished as planned at the beginning, the product, in this case, the project, would still maintain a potentially releasable state.

5.1 Robot selection

Extensive research was needed to ensure the quality of the selection. Each robot has advantages and disadvantages, and the final choice must be taken only after evaluating which are the strengths that would favour the most this project and would make testing and developing easier to perform.

The goal was to select a mobile robot that included the most useful characteristics for the project. The priorities considered when making the selection were, in the following order:

1) Sensors: the navigation, vision and localization sensors, the level of complexity when working with them and the average quality of the measurements obtained when testing them in a real scenario.
2) Documentation: the backup from the company selling the robot, the information provided by them and the number of theses and investigation projects developed using the specified robot.
3) Implementation: the versatility, power and capacities of the motherboard that mounts the robot and if other extra sensors could be installed on the device. The community support as well as the libraries and packages already implemented for this robot.
4) Construction: the ease of assembly and manuals, the quality of the materials and components and the estimated durability of the robot. The versatility of structure to be modified according to the customer’s needs.
5) Cost: the market price of the robot including shipping costs and the stock. The after-sale service is also considered as a value-adding feature.

In order to make the robot selection, the criteria described above was evaluated. To do so, a weighted-sum matrix was created. As shown in Table 1, a value from 1 (minimum) to 5 (maximum) was used to evaluate the performance of each robot in the different aspects.

Table 1. Weighted-sum matrix for robot selection

	Waveshare AlphaBot2	**SLAM robot**	**LiDAR-based Self Driving Car**	**Husarion ROSbot 2.0**	**TurtleBot3 Waffle Pi**	**TurtleBot3 Burger**
Sensors	3	3	3	5	5	4
Documentation	2	1	1	4	5	5
Implementation	4	4	2	5	5	5
Construction	3	2	2	5	5	5
Cost	5	5	5	1	2	4
Total	**17**	**15**	**13**	**20**	**22**	**23**

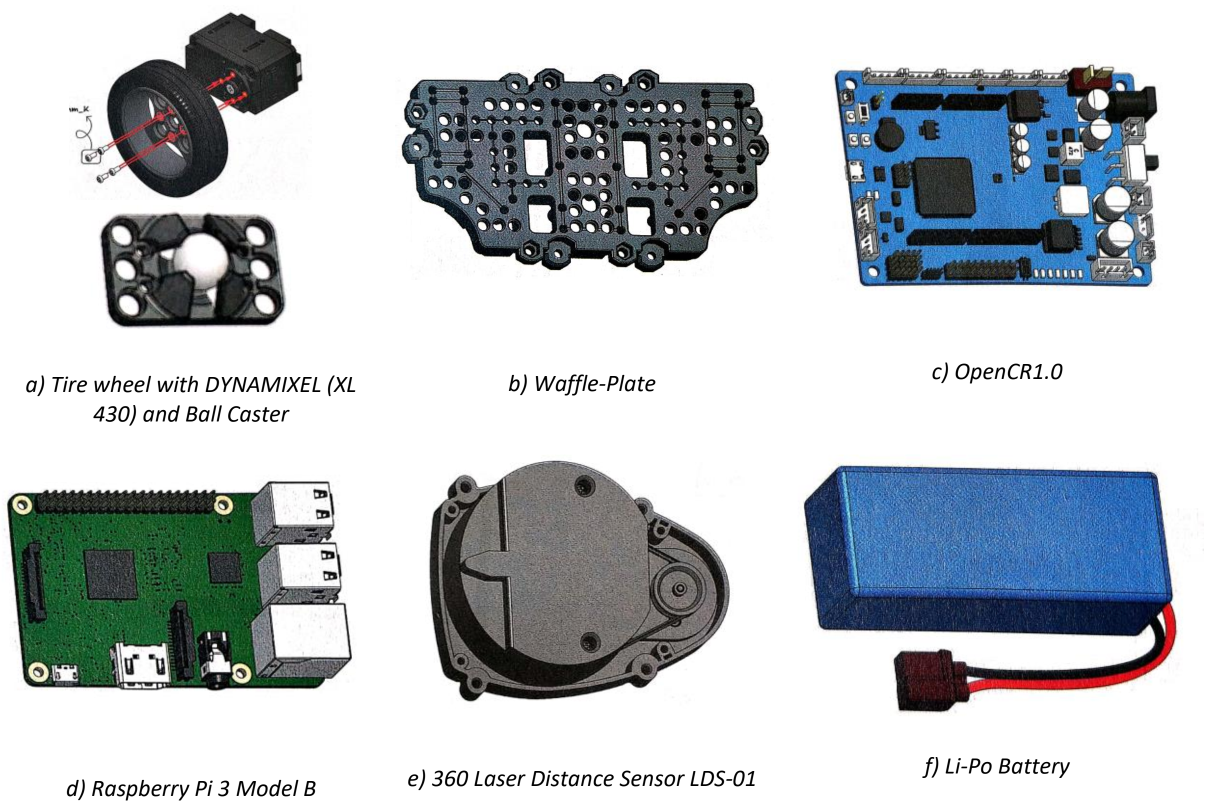
As a result, the robot with the higher total value was the TurtleBot3 Burger. It was the most suitable one to evaluate the implementation of indoor navigation and localization described in the following sections.

5.2 Robot assembly

TurtleBot3 Burger is delevoped by ROBOTIS, which is one of the leading manufacturers of robotic hardware. They specialize in the manufacture of robotic hardware and full platforms for use in all fields of study and industry, as well as educational robotics kits for all ages and skill levels.

The TurtleBot3 Burger kit includes a printed easy-to-follow assembly manual that explains step by step how to build the robot correctly. The e-Manual [82] provides both a PDF to download and a set of videos explaining how to assemble it.

Figure 23 shows the main components of the TurtleBot3 Burger. The encoder is shown in Figure 23a) is of type XL430-W250, leading the robot to have a maximum velocity of 0.22 m/s. The shape of the Waffle plate leads to a highly versatile structure, allowing the user to customize the robot according to the project needs. OpenCR (Open-source Control module for ROS) is developed for ROS embedded systems to provide completely open-source hardware and software. OpenCR provides digital and analogue input/output pins and supports 12V, 5V, 3.3V power outputs for the different sensors and the Raspberry Pi SBC, where the Raspbian Stretch OS is located. The Raspberry Pi Model B communicates with the ROS master (computer) in order to run the required ROS packages remotely. The LiDAR component is positioned at the top of the robot to allow the robot to measure distance with a 360^{o} range. The Li-Po battery has a capacity of 1800 mAh and is capable of providing a 2h 30m operating time to the robot with a charging time of 2h 30m.



a) Tire wheel with DYNAMIXEL (XL 430) and Ball Caster

b) Waffle-Plate

c) OpenCR1.0

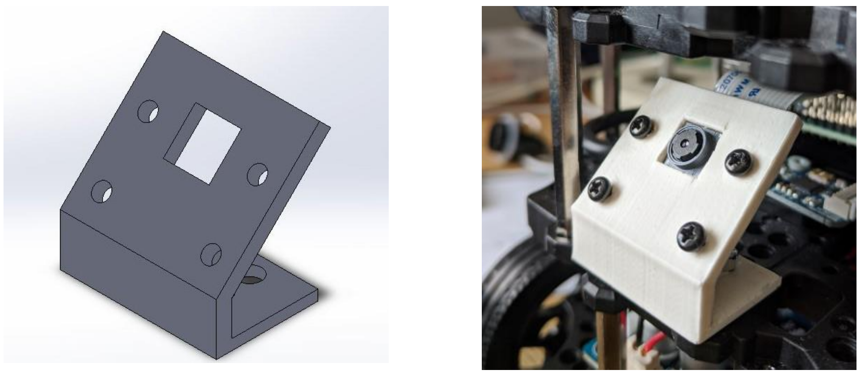
d) Raspberry Pi 3 Model B

e) 360 Laser Distance Sensor LDS-01

f) Li-Po Battery

Figure 23. TurtleBot3 Burger components

The TurtleBot3 is compatible with the Raspberry Pi Camera V2 as the mounted development board is Raspberry Pi 3. The Burger model does not include the camera device as a default sensor, which means that an extra part was needed in order to attach the camera to the robot structure. To make this possible a 3D part (Figure 24 a)) was designed using SolidWorks 3D CAD program so it could be exported in *.stl* format and printed on the university facilities. Figure 24 b) shows the final result with the camera connected to the Raspberry and assembled to the robot's structure using the printed piece.



a) 3D CAD design

b) 3D printed piece on TurtleBot

Figure 24. Raspberry camera mount

After modifying the structure so that the camera mount could be attached without compromising the robot robustness, the dimensions of the robot were 138mm x 178mm x 192mm (Length x Width x Height), weighted 1kg and had a maximum payload of 15kg. The final result is showed in Figure 25.

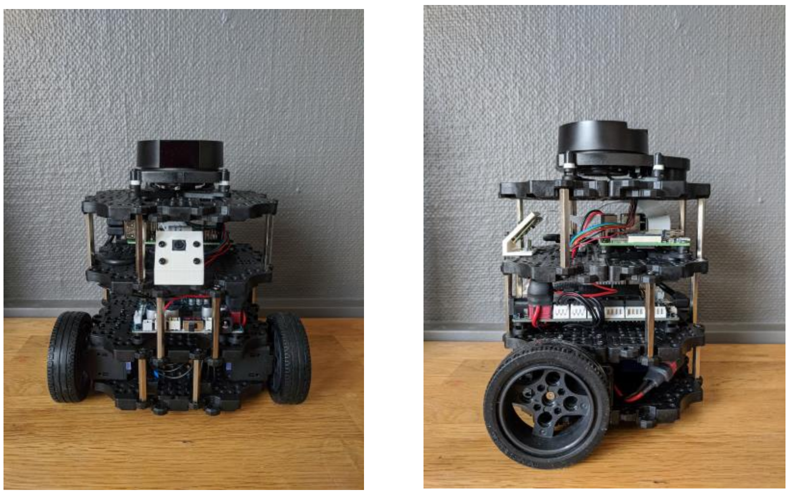


Figure 25. Final robot assembly

5.3 Software setup

TurtleBot3 is also developed by Open Robotics, it is in charge of the software and community activities. Open Robotics works with industry, academia, and government to create and support open software for use in robotics, from research and education to product development. Therefore, it is of great benefit for the project as a lot of documentation is available.

5.3.1 Bring up

Communication between devices was the first thing that was required for the robot to start. They consist of a Remote PC, which is the one used to communicate with the TurtleBot, and an SBC, which is used to communicate with the Remote PC.

The first step was to set up the Remote PC by installing the Ubuntu 16.04 OS and ROS environment. The TurtleBot e-Manual [82] included the respective links for the OS download and the commands that were needed to run in the terminal window, as well as the required packages to be installed. ROS Kinetic Kame is the version that works best for the TurtleBot3, although it also supports Melodic Morenia. Secondly, ROS 1 requires network configuration to communicate between TurtleBot (Slave) and the Remote PC (Master).

Once the communication had been configured, the SBC setup must be done. The Raspbian Stretch was the OS used on this project for the Raspberry Pi 3 Model B (TurtleBot3); it was flashed into a microSD card in order to be inserted in the robot. The dependency ROS and TurtleBot packages were needed to be installed and the network configuration on the TurtleBot was completed. After doing this, the IP address of the Remote PC was assigned as the ROS Master IP and the access to the Raspberry could be easily done from the Master using SSH (Secure Shell) protocol on the terminal window.

Finally, the OpenCR setup. The e-Manual provides two methods for uploading firmware although it is recommended to use shell script. Therefore, the Shell Script method was chosen, and the specified commands were used to connect OpenCR to Remote PC or connect OpenCR to TurtleBot PC.

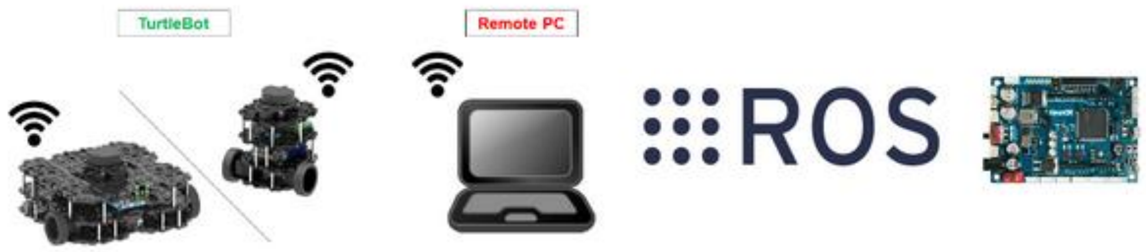
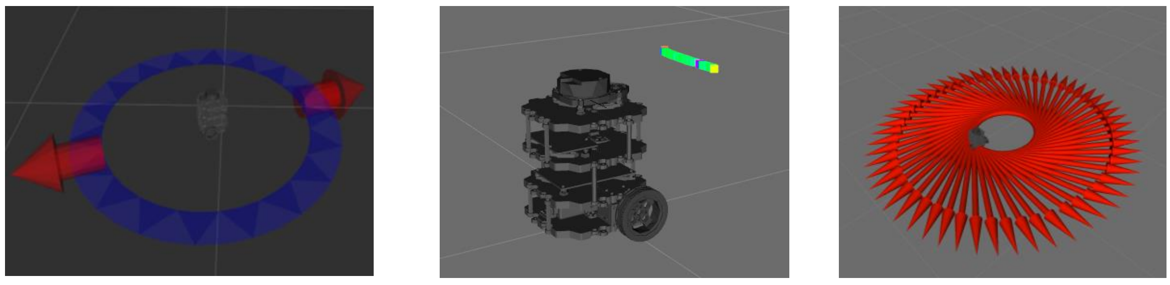


Figure 26. Communication between TurtleBot and Remote PC using ROS

5.3.2 Basic operations

Once the necessary software was set up, different examples were tested to confirm the correct functionality of the robot. The following operations were completed and can be seen in Figure 27.



a) Interactive marker *b) Stop using LDS* *c) Circle pattern*

Figure 27. Basic operations

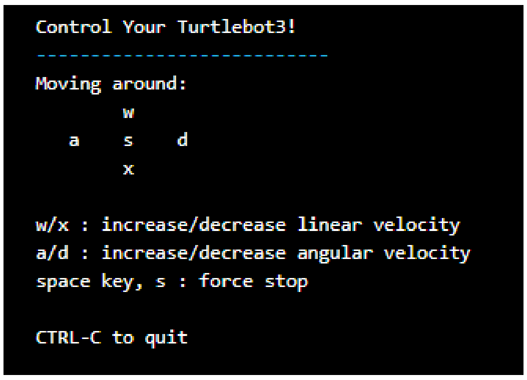
The first instruction was to move the robot by using the interactive marker on the RViz environment. The red arrows allow the user to remotely move the robot linearly side to side, whilst the blue ring allows the user to make the robot rotate. All possible movements were tested with a successful result. The robot smoothly moved without showing any difficulty.

The second instruction consists of moving the robot and stopping it using Laser Distance Sensor (LDS). Once the LDS package was launched, the robot started moving forward until an object was detected ahead. After testing it in various scenarios, the robot was observed to respond efficiently to different materials and shapes and kept going forward when the object was not present.

The last instruction was to move the robot to custom routes. This consists of choosing a specific path shape (such as square, triangle or circle) that the robot would follow as well as choosing its area and how many times it should repeat the route. This example also showed on the RViz environment red arrows to draw in real-time the robot's direction. The robot responded directly to the instructions given and a smooth path was performed when given the operation for the three different path shapes.

5.3.3 Teleoperation

The next challenge was to make it move with remote control as the TurtleBot3 can be teleoperated with several wireless devices such as PlayStation3 and XBOX 360 controllers or launching the *turtlebot3_teleop* package on the Remote PC. The following instruction appeared on the terminal (Figure 28) window when executing the *teleop* ROS package.



```
Control Your Turtlebot3!
---------------------------
Moving around:
        w
   a    s    d
        x

w/x : increase/decrease linear velocity
a/d : increase/decrease angular velocity
space key, s : force stop

CTRL-C to quit
```

Figure 28. Teleoperation command window

Both teleoperation methods were adopted and completed. The robot was able to smoothly move around the area and respond to the given commands at the moment. However, at this point, the robot was not able to stop if an obstacle was present neither with the Remote PC control nor the Android App.

5.3.4 SLAM Methods

The next step was to be able to draw a map. TurtleBot3 supports gMapping, Cartographer, Hector, and Frontier among various SLAM methods. The Rviz environment is the visualization tool by default once either of the four SLAM files is launched and subsequently tested.

5.3.5 Navigation

The previously saved map was used for the navigation of the robot. The Rviz environment includes a button that allows the user to make an initial pose estimation for the vehicle and a button to establish a navigation goal. The robot, therefore, creates a path to avoid obstacles to its destination based on the map.

5.4 Facial recognition

Face detection and identification in a video stream was implemented using OpenCV, ROS, Python, and deep learning. Facial recognition is conformed of a series of different problems:

- Face detection
- Posing and projecting faces
- Encoding faces
- Compare and find the person from the encoding

In deep learning, a neural network needs to be trained to get a single input image and output a classification for that image. For this project, deep metric learning was used. The main advantage is that, instead of giving a single output, a real-valued feature vector is outputted. The training process consists of loading two images of a known person and a third picture of a different random person, then the algorithm quantifies the faces by constructing a 128-d embedding for each one. After comparing the results, it tweaks the neural network so that it makes sure the images #1 and #2 are similar and #2 and #3 measurements are far away (Figure 29). After repeating this step millions of times for several images of different people, the accuracy of the face recognition using this method can be around 90% [83].

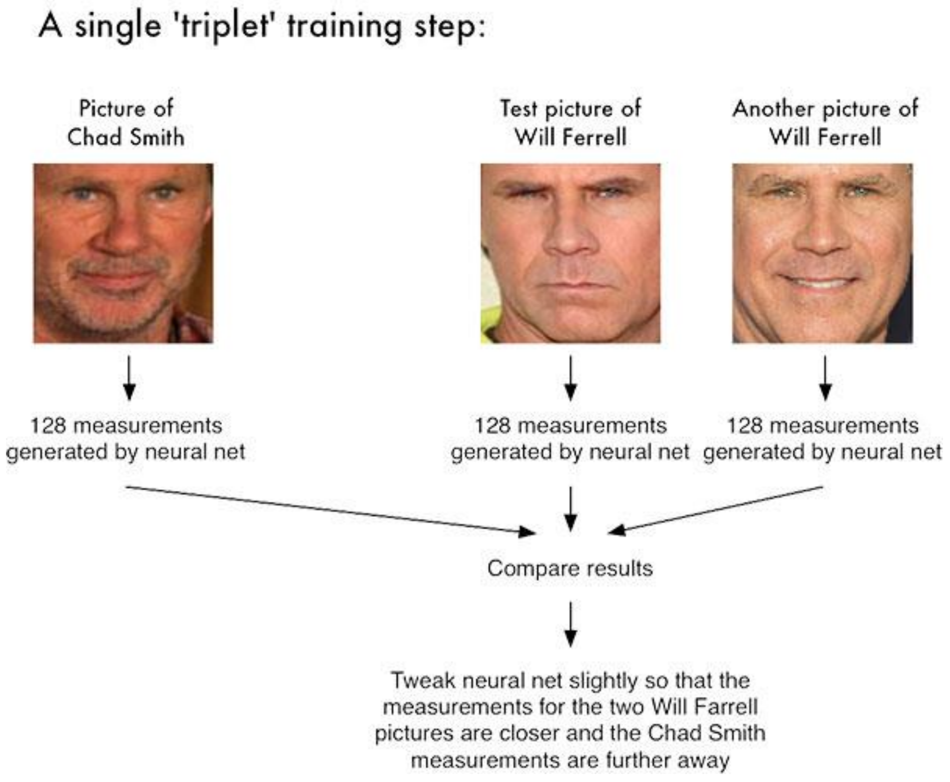

A single 'triplet' training step:
Picture of Chad Smith
Test picture of Will Ferrell
Another picture of Will Ferrell
128 measurements generated by neural net
128 measurements generated by neural net
128 measurements generated by neural net
Compare results
Tweak neural net slightly so that the measurements for the two Will Farrell pictures are closer and the Chad Smith measurements are further away

Figure 29. Triplet training step [83]

On the other hand, high computational power is needed. This part of the project was technically possible to be run into the Raspberry Pi, but after testing the code on the SBC, it was declared as an impossible task due to the lack of computational power and so the ROS master computer was used. This PC (ROS master) encounters the GPU GeForce GTX 1060 from NVIDIA, which is compatible with the CUDA technology. CUDA is a computing platform that reduces enormously the processing times on this type of applications thanks to faster downloads and readbacks to and from the GPU (more information at [84]).

In order to perform the face recognition with Python and OpenCV, *dlib* and *face_recognition* libraries were used. The *dlib* library [85] provides with an implementation of the mentioned “deep metric learning” and the *face_recognition* library makes it easier to work with *dlib* facial recognition functionalities. Before starting to recognize faces, the first thing to do was create the embeddings (quantifications) of each picture in the dataset. It is important to understand that the neural network used had already been trained with a dataset of more than 3 million images to detect faces; the network was not being trained here, it was just used to create the embeddings of the pictures in the project’s dataset which was constructed manually and consisted in a total of 45 images of the project members and the tutor from different perspectives and lighting conditions (Figure 30). A new network could be created and trained from scratch, but it would consume a lot of time and resources without achieving the quality of the pre-trained one; this is the reason why the *dlib* facial recognition network was selected. The functions of the libraries mentioned above were applied to identify the faces in the pictures, encode, and store them with their name on the encodings list that was used to perform the recognition later on [86].

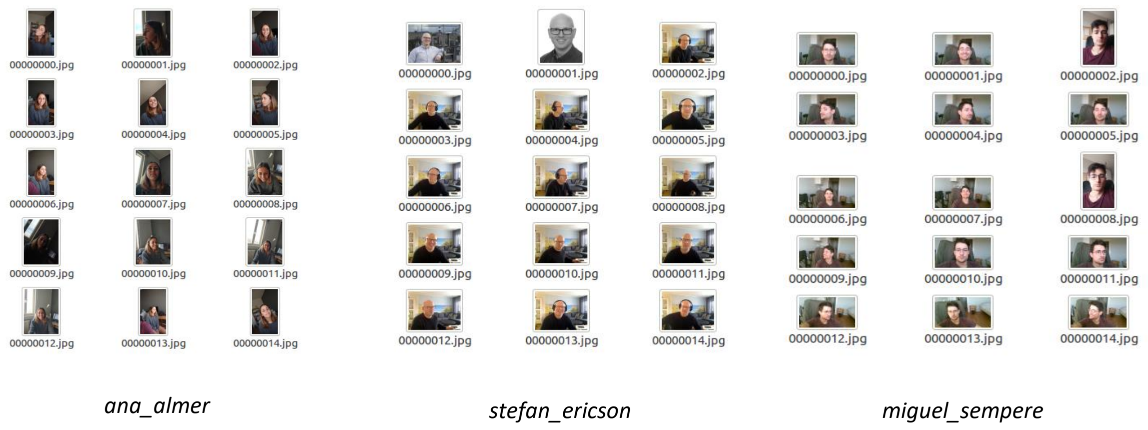


Figure 30. Facial recognition dataset

The next step was to recognize faces in the video stream from the Raspberry Pi camera attached to the TurtleBot3. The key was to use the existing ROS package available that publishes the Raspberry camera video stream into a topic (*/raspicam_node/image*). By creating a node (*/face_recog*) and subscribing to this topic where the image is being published, the face recognition program is able to access the video. Figure 31 shows the node structure.

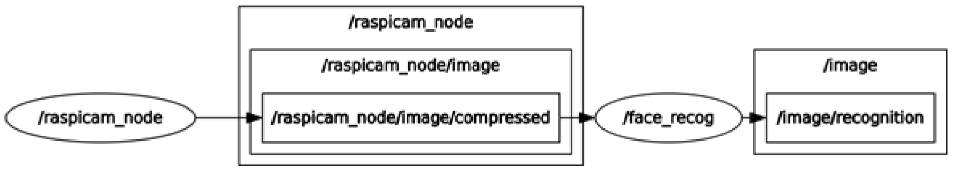


Figure 31. ROS face_recog node structure

After doing this, the image can be processed using OpenCV. Since ROS passes the image with *sensor_msg.msg/CompressedImage* or *sensor_msg.msg/Image* message format, an interface was needed to use the image in OpenCV. CvBridge is a ROS library that allowed this conversion to *cv::Mat* format used in OpenCV [87]. After obtaining the image in *Mat* format, it is pre-processed, the faces are detected and the encodings for each face are calculated. A loop is used to compare the obtained encodings of the particular frame to the ones on the dataset, storing the matching ones. Another loop is then used to draw a box around the faces along with the name of the person above it; if the face of the person is not recognized as one of the dataset, the program labels it as "*Unknown*". The image resulted of this processing is transformed back to *sensor_msg.msg/Image* message format using CvBridge and published on a new topic (*/image/recognition*); this topic is created to be able to access to the face recognition image from other programs.

5.5 Patient localization

Indoor localization can be implemented in many ways. This means that a previous study of the environment used for testing the robot was needed to choose which technology, technique and algorithm were the most suitable.

The area used for testing the robot and its basic operations was the one used for testing indoor localization. It consists of two main corridors; this means that it encompasses little obstacles such as common walls or rooms in between that may affect the accuracy of results. Working with a line of sight was possible and therefore it was suitable for the implementation of the RSS algorithm. Considering this and the fact that it did not require additional hardware and it had a higher versatility than AOA or TDOA, the RSS algorithm was chosen for this project. However, it is complicated for the algorithm to ensure a highly accurate result, but the complex implementation and the higher cost that other possible positioning options require means that the most suitable option was the implementation of RSS.

Based on the literature review previously done, fingerprinting technique is usually proved to be the best option for indoor positioning. On the other hand, triangulation was chosen for this project as the grid created by the robot to navigate could be used to locate the signal transmitters on the map and therefore there was no need to do site survey as in for fingerprinting. Some other facts from this technique to encounter were that it was easier to deploy and that the straight corridors allow line of sight which means that a successful performance may be achieved. Hence, considering that the implementation of fingerprinting is more tedious than triangulation due to the high quantity and complexity of measurements and the amount of technical skill it requires to set up and maintain, the chosen technique for the indoor localization of the patient was triangulation. Triangulation encompasses angulation and lateration. In this project, lateration was used and it can be calculated as follows [77]:

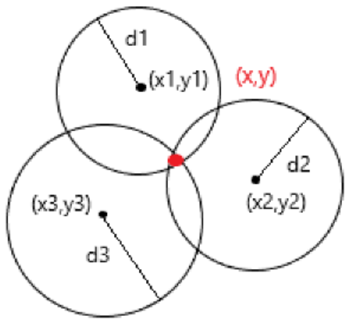


Figure 32. Trilateration algorithm

$$(x_1 - x)^2 + (y_1 - y)^2 = d1^2$$

$$(x_2 - x)^2 + (y_2 - y)^2 = d2^2$$

$$(x_3 - x)^2 + (y_3 - y)^2 = d3^2$$

Where:

x_i = x-coordinate for transmitter i

y_i = y-coordinate for transmitter i

di = distance from transmitter i to patient

These equations reduce to a linear set of simultaneous equations which can be solved using matrices. The system gives a unique solution (x,y) which is the desired position of the patient. The distance between beacons and the patient is measured using attenuation and the path loss model. The path loss model is based on the signal loss between the transmit antenna of an AP and the receiving antenna that the patient carries. Therefore, it relies on the distance loss which is calculated as follows [88]:

$$PL = PL_0 + 10\, n \log\left({}^{d}/_{d_0}\right)$$

Where:

PL_0 = path loss at a distance of d_0 according to the distance loss model

d [m] = distance along the path between the access point and patient

d_0 [m] = reference distance*

n = path loss exponent*

*d_0 was chosen at 1m and a value of 1.8 for the n variable was used.

The three better radio signals options are UWB, Bluetooth and Wi-Fi. The first one, although it is the most efficient signal, requires expensive equipment that is not affordable for this project. Therefore, an evaluation of the Bluetooth and Wi-Fi signal will be made although the natural signal oscillation that both present requires the final position result to be slightly modified in some cases in order to force the robot to stand inside the working area.

Regarding the Bluetooth technology, Bluetooth 4.0 is a promising option due to the low energy consumption and configuration options compared to the Wi-Fi technology. Existing beacons are revolutionizing the industry as they can sense proximity and position with low cost for indoor spaces. BLE beacons are tiny devices that use BLE signals to transmit messages to mobile devices with a short distance or proximity of the beacon [89]. They have different types of specifications which can be divided into three: iBeacon, Eddystone and AtlBeacon. iBeacon is Apple's implementation of this technology and it is classified as a widely supported and easy to implement beacon. Secondly, Eddystone beacon is a Google protocol classified as a flexible and open format. Regarding AtlBeacon, is an open-source beacon launched by Radius Network which is also flexible and compatible with other operating platforms [90].

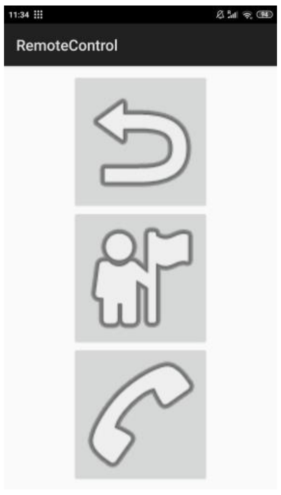
The beacon used for this project is the Radbeacon Dot, developed by Radius Networks. It is described as a fully standalone Bluetooth smart proximity beacon that uses iBeacon, AltBeacon and Eddystone.

This means that high technology support is encountered as well as the capability of simultaneous services across different environments. Due to the simple implementation, the three beacons that are going to be used for indoor localization. Before being able to use them, they have to be previously configured with the use of the RadBeacon App. This app allows the user to choose the name, type, and UUID of the beacon as well as to calibrate it.

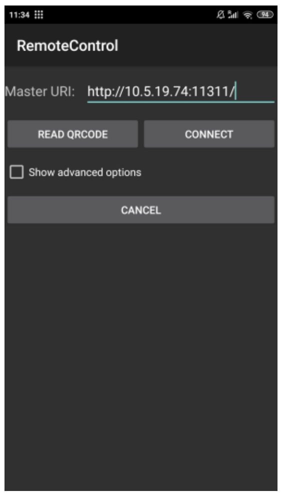
5.6 Communication interface

As this project is working on a mobile robot destined for domestic use, it is of great importance that it is as user-friendly as possible. The objective of this interface is to facilitate as much as possible the task of human-machine communication and so the most logical communication interface to use is the remote control, especially when the robot is designed for people with reduced mobility, who will not be able to make great efforts or reach the robot easily if it is continuously moving around the house. Furthermore, it makes no sense to implement an interface where the robot needs to be manipulated physically because sometimes it will be at the opposite part of the house and the task of coming to the patient's position needs to be commanded from the patient's location. Since Bluetooth protocol may have a limitation of range that can make it possible to achieve a satisfactory interface, an Android application (App) along with ROS is chosen as the most suitable communication interface.

As the core of the project is developed using ROS, it would be interesting and helpful to develop an Android App that communicates with the robot using ROS as well. Android code is based on the *rosjava* [91] client and core libraries and is prepared to work smoothly with Android Studio and the Google supported Android Gradle plug-ins [92]. This means that Android supports ROS and the App must be developed using Java and the Android Studio and SDK tools [93]. The App design needs to be intuitive and will have three simple options (buttons): come here (go to the patient's location), go home (go to the initial position of the robot) and call emergency (directly call 112 in case of an emergency). Figure 33 a) shows the final design with the three different options and Figure 33 c) shows the call function when the emergency option is selected.

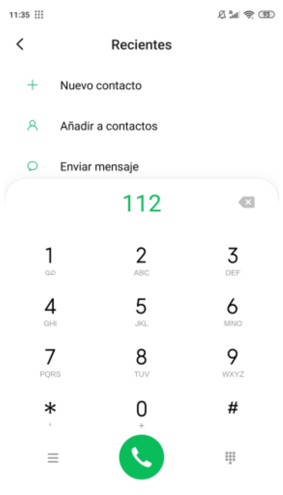


a) Application layout



Master URI: http://10.5.19.74:11311/

b) URI address configuration



c) Call functionality

Figure 33. Android application

The Android App needs to send the patient's choice to the ROS master computer using Wi-Fi, so the master URI must be specified when the App starts up, as seen in Figure 33 b). As shown in Figure 34, the ROS structure of this application is simple. The App creates a topic (*/messager*) and a node (*/control/talker*) subscribed to it, which will publish on the topic a string with the selected option. With another python program, the master computer creates another node (*/listener*), subscribes it to the topic and reads the published string in order to launch one program or another.

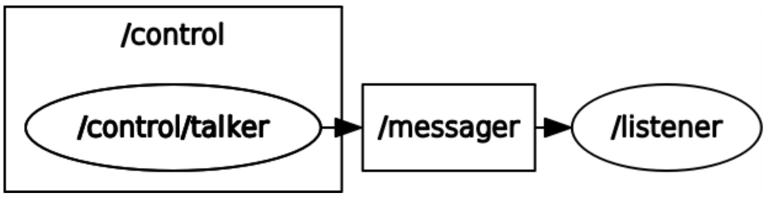


Figure 34. Android App ROS node structure

5.7 Final design

Figure 35 shows the project's structure. The thick arrows connect the main steps of the process, that is, the main flow; thinner arrows are used to connect the secondary tasks coming out of the primary branch and establish the relationship between them. As specified in the diagram legend, hardware blocks are red, ROS packages (scripts and *.launch* files) are blue, and the purple (ROS), yellow (OpenCV), and green (Android Studio) dashed lined areas correspond to the software used to develop each part of the project. The relevant topics and nodes subscribed or created by the different scripts are specified on the arrows connecting the boxes.

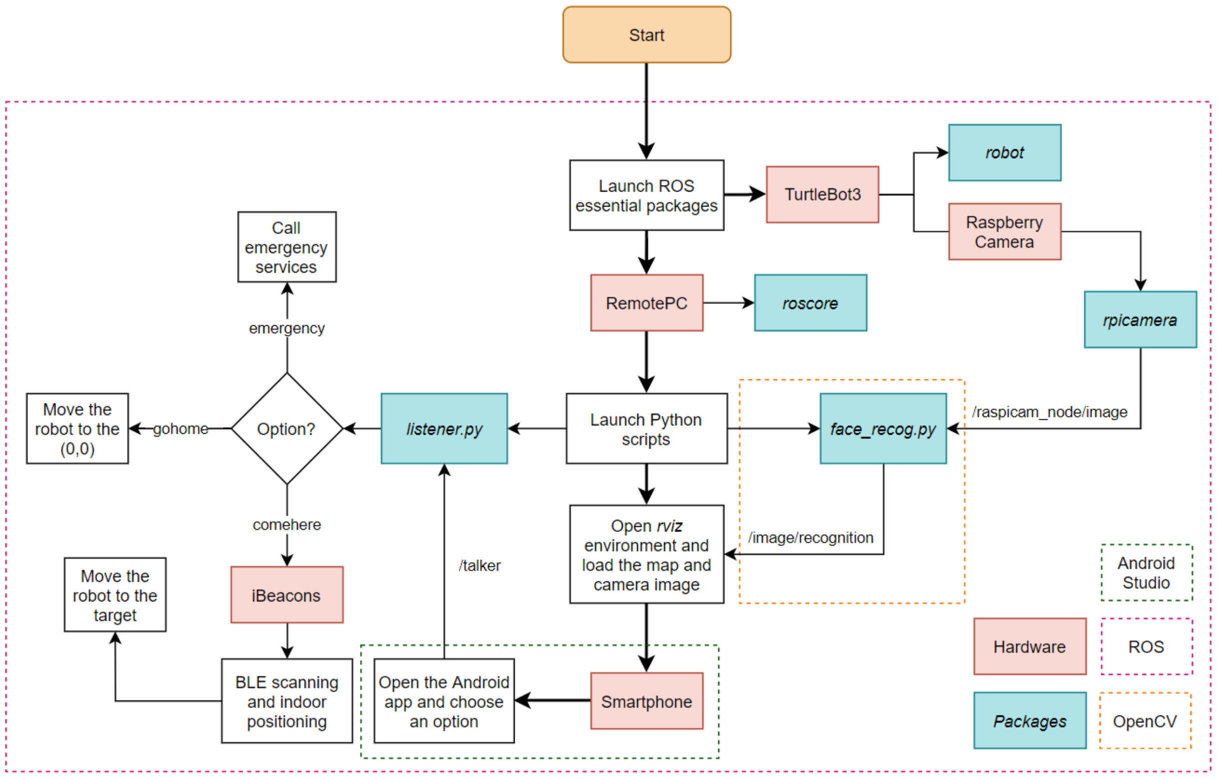


Figure 35. Project structure diagram

6. Evaluation

6.1. SLAM methods

The four different SLAM methods were tested to evaluate which map works best for this project. Before launching the robot, the working area was defined by closing some corridors and adding two static obstacles to test the robot's obstacle avoidance when travelling from one point to another. To test each method, the teleoperation node was launched and driven through the same path until it reached every corner of the area.

6.2. Navigation

The aim was to move the robot from one location to the specified destination inside the working area. To do so, both the map previously created, and the established initial point and orientation are used to test navigation. Ten different points on the map were set as goals to evaluate how the robot responded to it. The ability of the vehicle to reach its destination even with the presence of disturbance was also evaluated.

6.3. Facial recognition

In order to test the accuracy and reliability of the developed face recognition system, a new set of 20 images from five different strangers, the tutor and the members of this project were collected (Figure 36).

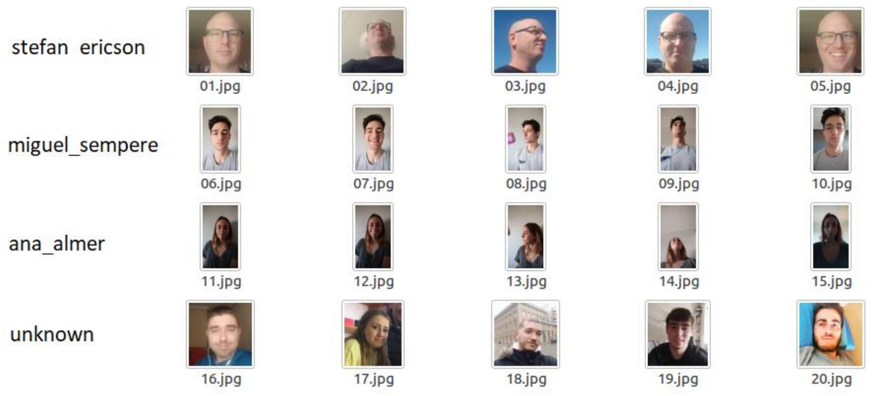


Figure 36. Test image set

The evaluation test consisted in feeding the program with all the images. The code was modified just for this test so that it would detect and identify the faces from image files and not from the Raspberry Pi Camera video stream. The goal was to check how many times the system succeeded in identifying not only the known faces but also the unknown ones. The pictures of the different strangers must be identified as "Unknown", and the ones of the three known people on the dataset used to train the network must be labelled with the correct name; otherwise the system was considered to fail, and the reliability and accuracy decreased.

6.4. Patient localization

In this section, Wi-Fi and Bluetooth technologies were evaluated to choose the one that provided better indoor positioning results and therefore, was the best choice for the final evaluation.

6.4.1 Wi-Fi technology

A great advantage of evaluating the robot in a common working area was that there were Wi-Fi APs already installed. This means that there was no need to change the working area that was previously established for the evaluation of the different SLAM methods, because the Wi-Fi signal coverage was captured from every no.

Firstly, Python code was generated for the signal scan to verify that at least three APs were reachable from the laptop. To do so, various scans were made at different points on the map shown in Figure 37, which was the map created when testing the Hector SLAM method.

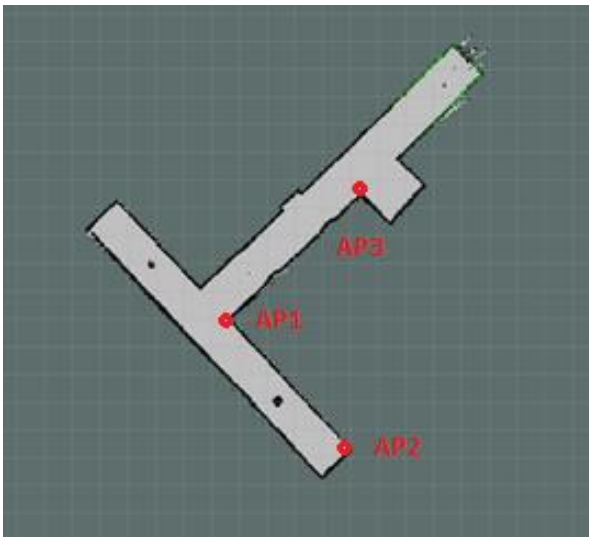


Figure 37. The map used for Wi-Fi signal scan

The scanning code saves the RSS value from each router in a .csv file to feed the path loss model and consequently obtain each distance [m] required for the trilateration algorithm. Coordinates of the three Wi-Fi routers are also required for the algorithm and therefore established using the default robot navigation grid generated in the Rviz environment. The position estimation is then evaluated at different points of the map by comparing the actual point of the laptop with the given one.

6.4.2 Bluetooth technology

As with the Wi-Fi technology, three devices were used. The RadBeacons were configured and positioned randomly across the map; the names established were Beacon1, Beacon2, and Beacon3. Afterwards, a Python code was created to scan the BLE signal and check the signal reachability across the whole working environment. The code was tested by scanning from the map points used for the Wi-Fi technology evaluation. However, due to the shorter signal range of the iBeacon used, the laptop only detected one beacon when standing at the endpoint of each corridor. This is why the working environment was changed (see Figure 38) by shortening the corridors, but the two static obstacles and the use of the Hector SLAM method were maintained. B1, B2, and B3 stand for Beacon1, Beacon2, and Beacon3, respectively.

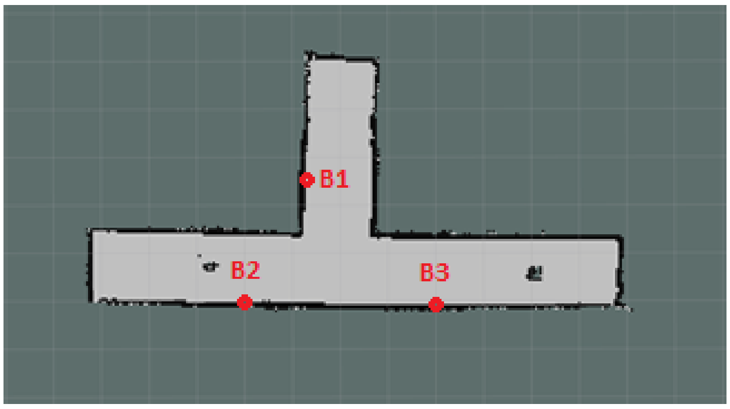


Figure 38. The map used for BLE signal scan

The coordinates of the three beacons were established by using the robot navigation grid generated in the Rviz environment. To test the data gathered from the beacons, the scan code was executed from different positions inside the valid area. When good results were obtained, these values were used in the trilateration algorithm. The final position provided by the algorithm was evaluated at different points of the working environment.

The conditions that force the robot to stay inside the defined area were included in the code of the technology which provides better results for indoor positioning.

6.5. Final evaluation

After evaluating the results obtained from the different aspects of the project, they were compiled to evaluate the final robot performance. The SLAM method and indoor positioning technology that performed better on the performed test were used for this final evaluation. Firstly, a new map was created and saved to avoid any possible position-related issue that might be present from the previous test. The home position was then established and indicated with a painted cross-mark on the floor in order to refer to both the ideal fixed station where the pills are stored and refilled by the nurse and the robot’s charging point. Three more points on the map are established in order to locate the signal devices of the technology used for indoor localization as shown in Figure 39.

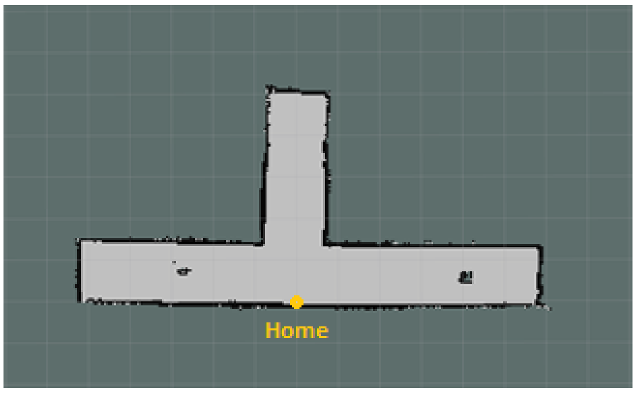


Figure 39. Area used for the robot final evaluation

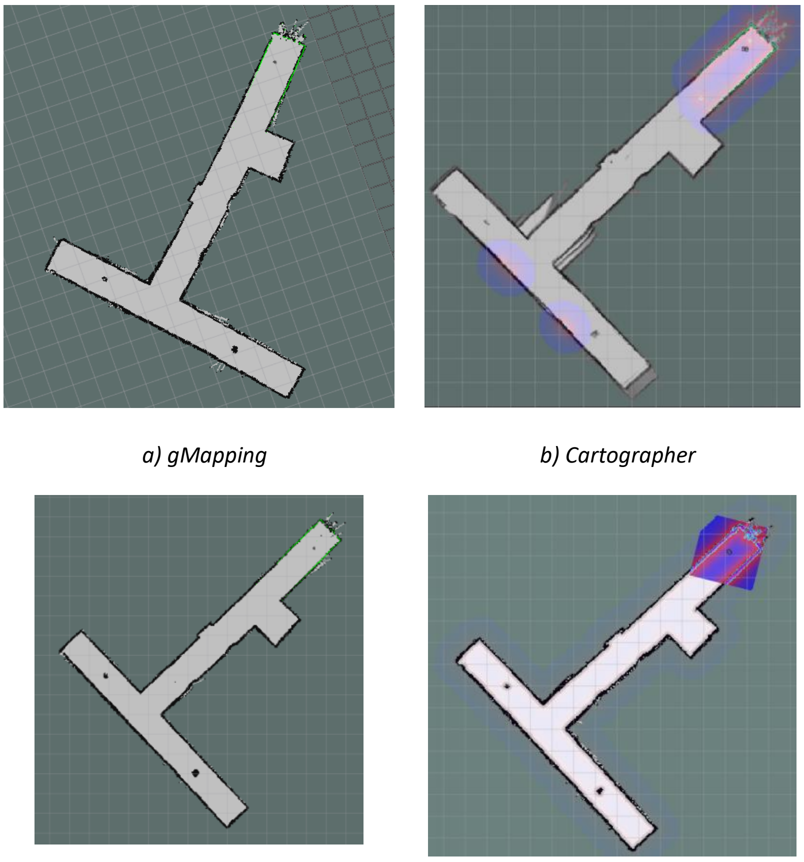
For this evaluation, the patient must stand within the defined working area with a laptop and a smartphone. These devices were incorporated to satisfy the ideal bracelet that the patient would need to wear in a real-life situation. The computer is the ROS master and was used for launching the main package that communicates with the Android application and sends the orders to the robot according to the selected task; if the "*comehere*" button is pressed, the computer locates itself on the environment using Bluetooth and moves the robot to that position; if the "*gohome*" command is received, the robot moves to the (0,0) coordinate, which is the home position. Meanwhile, the laptop shows the camera image along with the facial recognition service, and the map with the robot's position and route.

The robot's performance was tested under different conditions to ensure a solid project conclusion. A total of 10 tests were taken for which the first 5 included dynamic obstacles that appeared and disappeared across the robot's path, and the 5 remaining tests included static obstacles over the defined working area. These conditions tested if the robot would be able to recalculate the path as it travels towards the patient or as it returns to its home position. If the vehicle successfully reached the navigation goal, the patient needed to come closer to the robot's camera in order to evaluate facial recognition.

7. Results

7.1. SLAM method

Figure 40 shows the different results obtained. Based on the visual comparison, it was determined that the Hector and Frontier methods produced a clearly defined result, whereas gMapping and Cartographer showed errors in the output. When Cartographer was launched, some erros appeared when mapping the environment as shown in Figure 40b). Furthermore, gMapping showed some errors as the method updates the representation of the map based on the robot’s position. Drift error in position increases as the robot travels causing the curvature observed in a). This issue is solved with the Hector method as it does not depend on the odometry data, although the map has to be created using a much lower vehicle speed when turning into a new corridor. The frontier method encompasses the same issue when changing the robot’s direction, but it also updates the map when it recognizes that the robot has revisited the same place. This causes the map to shifts slightly from the real environment due to drifts in localization along the whole path.



a) gMapping *b) Cartographer*

c) Hector *d) Frontier*

Figure 40. Slam methods

7.2. Navigation

The robot succesfully reached the ten different established targets despite the obstacles presented along the vehicle's path. An average drift error of 0.30 metres was measured; Figure 41 shows the position deviation present in the robot's navigation.

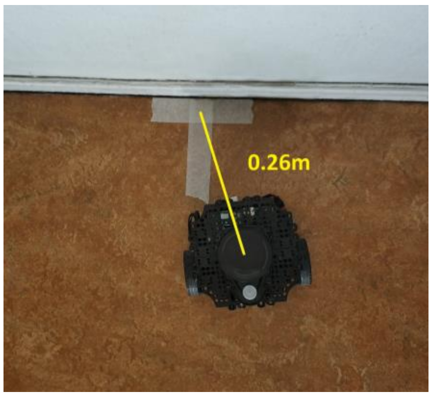


Figure 41. Drift error for robot's position

7.3. Facial recognition

The results obtained after performing the test with the set of 20 images are presented in Table 2.

Table 2. Facial recognition results

Label	Recognised images
miguel_sempere	5/5
ana_almer	5/5
stefan_ericson	5/5
unknown	3/5

The face recognition task succeeded 18 times out of 20 images. This means that, based on the table, the success rate or accuracy of the system is of 90%. The images of the known subjects were identified successfully, even though some of them had poor lighting conditions, strange poses, and different expressions in order to trick the program. Nevertheless, 2 images of the 5 different strangers were labelled as one of the known ones.

It is necessary to point out that the results obtained from this test are not enough to determine the exact accuracy of the face recognition system due to the small sample used, but facial recognition was not the main focus of this project and so the goal of this evaluation was to get an approximation of the system success rate.

7.4. Patient localization

In this section results of Wi-Fi and Bluetooth technology used for indoor localization is presented and evaluated.

7.4.1 Wi-Fi technology

The Wi-Fi scan was successful for each Wi-Fi AP. Table 3 shows the values for the measured distance after using the path loss model. The table includes three different measures obtained from standing 1, 2, 3, 4, 5, and 10 metres away from the Wi-Fi AP, the average value in metres underneath, and the encountered average error from each distance.

Table 3. Measured distance to three Wi-Fi APs

Actual distance (m)	Measured distance to AP1 (m)			Measured distance to AP2 (m)			Measured distance to AP3 (m)			Drift error (m)
1	1.29	1.29	1.29	0.79	0.79	0.79	5.01	5.01	2.51	1.23
		1.29			0.79			4.18		
2	1.81	2.15	2.15	1.00	0.73	1.00	2.51	3.98	5.01	1.00
		2.04			0.93			3.90		
3	3.03	3.03	3.03	1.99	1.99	2.51	2.51	2.51	2.51	0.45
		3.03			2.16			2.51		
4	3.03	3.03	2.56	2.51	2.51	2.51	10.00	7.94	10.00	2.64
		2.87			2.51			9.31		
5	3.59	3.59	2.15	3.98	2.51	5.01	10.00	6.31	6.31	1.87
		3.11			3.83			7.54		
10	5.05	3.59	5.05	15.85	19.95	19.95	6.31	6.31	10.00	5.49
		4.56			18.58			7.54		

The results obtained showed the variance of the Wi-Fi signal. More accurate distances were obtained when standing closer to the AP than standing at 10m. These values were going to be used in the trilateration algorithm and the error was expected to increase. However, an evaluation of the results obtained from the trilateration algorithm with the Wi-Fi signal scan was made; the values shown in Table 4 and it is calculated that the average drift error gathered for the given positions was of 9.88m.

Table 4. Indoor positioning results with Wi-Fi

Actual position	Given position	Drift error (m)
(0, 0)	(4.57, 1.82)	4.92
(3, 3)	(-1.43, -1.92)	6.62
(1, 2)	(10.69, 9.47)	12.24
(-1.5, -1)	(-1.96, -2.35)	1.43
(-2.5, -1.5)	(-2.41, -2.88)	1.38
(-3.5, -3)	(-2.68, -3.04)	0.82
(1.5, -0.5)	(15.37, 13.74)	19.88
(2, -2)	(9.00, 10.20)	14.07
(3, -3)	(12.41, 6.89)	13.65
(7, -7)	(20.14, 12.78)	23.75

7.4.2 Bluetooth technology

The results for the BLE signal scan test when using the modified working environment proved to be successful. Table 5 contains the results for the calculated distances from the laptop standing 1, 2, 3, 4, 5, and 10 metres away from each iBeacon. Like with the Wi-Fi technology, three different measurements were gathered, and an average value was obtained from them. The final column shows the average drift error encountered for each distance.

Table 5. Measured distance to three iBeacons

Actual distance (m)	Measured distance to iBeacon 1 (m)			Measured distance to iBeacon 2 (m)			Measured distance to iBeacon 3 (m)			Drift error (m)
1	1.70	1.62	1.54	1.40	1.24	1.40	1.02	0.98	1.30	0.36
	1.62			1.35			1.10			
2	2.08	3.11	2.72	3.26	2.83	2.64	2.40	2.40	2.26	0.63
	2.64			2.91			2.35			
3	4.92	3.63	4.11	4.08	3.57	4.60	3.90	4.52	4.08	1.16
	4.22			4.08			4.17			

4	6.50	5.44	5.30	5.51	5.02	5.51	6.08	5.79	5.01	1.58
	5.75			5.35			5.63			
5	7.06	5.63	6.08	6.70	6.70	6.01	5.80	6.53	6.07	1.29
	6.26			6.47			6.13			
10	13.80	11.42	11.97	12.03	14.20	9.80	11.87	10.08	11.87	1.89
	12.40			12.01			11.27			

Results from the iBeacon scanning showed much less variation and a smaller average distance error than the measured values from the Wi-Fi APs. However, BLE signals also presented a higher error when standing further from the beacon. Table 6 shows the results obtained for indoor positioning using the trilateration algorithm and the iBeacon scan.

Table 6. Indoor positioning results with iBeacons

Actual position	Given position	Drift Error (m)
(0, 0)	(0.05, -0.37)	0.37
(1, 4)	(2.06, 6.88)	3.07
(0, 3)	(-1.36, 4.47)	2.00
(1, 2)	(-1.29, -0.04)	1.98
(3, 0.5)	(5.34, 1.76)	2.66
(4, 0)	(6.96, 0.00)	2.96
(2, 0)	(3.39, 0.04)	1.39
(1, -1)	(1.89, -1.12)	0.90
(1, -3)	(1.91, -4.23)	1.53
(0, -5)	(0.33, -7.95)	2.97

The resultant coordinates proved to be more accurate than the given coordinates after using the Wi-Fi scan results as the encountered average drift error was of 1.98m. A set of conditions were established in the Bluetooth code to ensure the robot target coordinate to stand inside the working area and the results in Table 7 were obtained. Values showed that the average drift error decreased from 1.98m to 0.88m after including the conditions that modified the resultant position.

Table 7. Final indoor position goal

Actual position	Modified position	Drift error (m)
(0, 0)	(0.20, 0.00)	0.20
(1, 4)	(0.50, 4.38)	0.63
(0.2, 3)	(0.50, 4.75)	1.78
(1, 2)	(0.50, 3.05)	1.16
(3, 0.5)	(2.43, 0.25)	0.62
(4, 0)	(4.47, 0.25)	0.53
(2, 0)	(1.11, -0.54)	1.04
(1, -1)	(0.50, -0.91)	0.51
(1, -3)	(0.50, -3.22)	0.55
(0.2, -5)	(0.5, -3.25)	1.78

7.5. Final result

The Hector SLAM method was selected for mapping and navigation, and for indoor positioning, the Bluetooth technology and the trilateration algorithm were chosen. The ten different tests were completed and presented in Table 8. The first column represents the point on the map on which the patient was standing while the second column represents the resultant coordinate of the trilateration algorithm. The value for the measured drift error is included in the third column, and the fourth and fifth one consists of a survey to know if the robot has reached the patient and if it had recognized it or not.

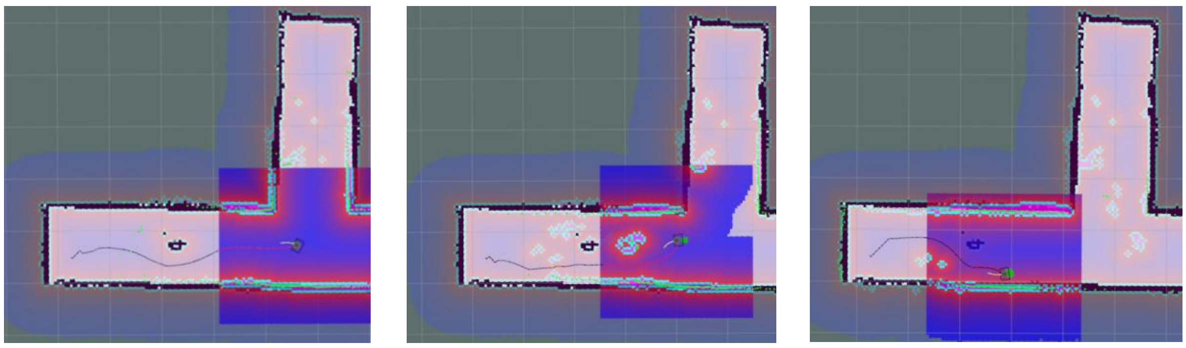
Table 8. Results for final robot evaluation

Actual position	Navigation goal	Drift error (m)	Obstacle avoidance	Reached destination?	Recognized the patient?
(0.5, 4)	(0.50, 4.75)	0.75	Dynamic	Yes	Yes
(1, 3)	(0.50, 4.24)	1.34	Dynamic	No	-
(0.5, 1.5)	(0.50, 2.52)	1.02	Dynamic	Yes	Yes
(1, 0.5)	(0.50, -0.01)	0.70	Dynamic	No	-
(2.5, 0)	(2.20, -0.25)	0.39	Dynamic	Yes	Yes

(3.5, 0.5)	(3.36, 0.25)	0.29	Static	Yes	Yes
(4, 0)	(4.50, 0.25)	0.56	Static	Yes	Yes
(1, -2)	(0.50, -2.81)	0.95	Static	Yes	Yes
(0.5, -3)	(0.50, -2.45)	0.55	Static	Yes	Yes
(1, -4.5)	(0.50, -2.80)	1.77	Static	Yes	Yes

The results obtained presented an accurate average drift error of 0.83 metres. An average accuracy of 80% was obtained for the navigation of the robot to the target, and facial recognition presented an accuracy rate of 100%. Recognizing the patient was not possible for cases in which the robot could not reach the desired position, but successful from different perspective and distances for the cases in which the robot was nearby.

Figure 42 is an example of the robot's path created with the presence of dynamic obstacles. In this case, the robot was going from the home to the patient's position. The path created to return back to the home position is shown in Figure 43.



a) Path created

b) Path update when obstacle detected

c) Path update when obstacle disappeared

Figure 42. TurtleBot3 Burger reaching the patient

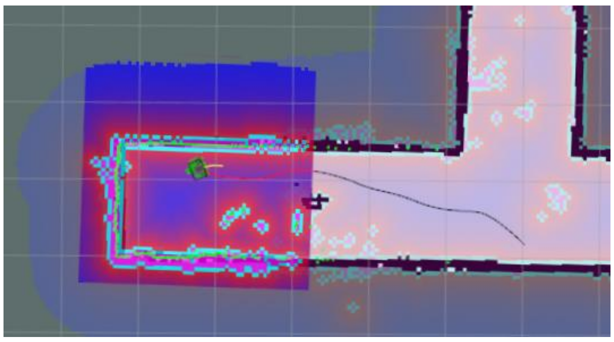


Figure 43. TurtleBot3 Burger returning to home position

The overall view of the computer screen is shown below in Figure 44. The left part of the screen shows the camera image along with facial recognition, while the right part shows the map of the environment allowing the patient to know in real-time the position of the robot and present obstacles across the area.

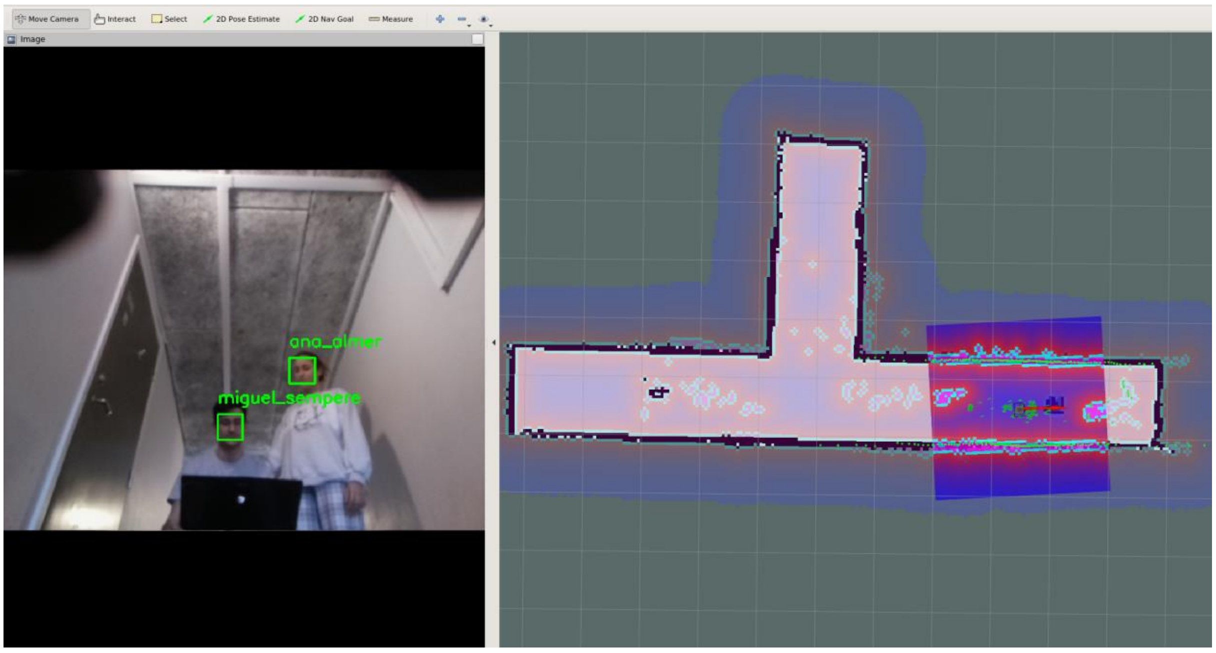


Figure 44. Computer screen layout during the execution

8. Discussion

Robot navigation was evaluated and concluded to be good enough for this project. For navigation to be possible, a map had to be created in order to work upon it. This means that an evaluation of the SLAM methods supported by the TurtleBot3 Burger was completed to ensure that the map used for further tests was as accurate as possible. First trials were performed with the gMapping method in a considerably large area but indicated errors in the output. This meant that the working area had to be reduced to keep testing the different SLAM methods. Finally, the Hector method proved to be the best one, and therefore it was chosen to be used for the following tests. The created map needed to be saved in order to test the TurtleBot3 Burger's navigation accuracy. To do so, a cross mark was painted on the floor to establish the (0, 0) coordinate of the environment with the aim of instructing the robot back to the point and measure the position error, which resulted to be about 0.30 metres for each time tested. This value was considered accurate enough to conclude that the robot navigation operated correctly, and it was, therefore, suitable for this project

The implementation of facial recognition using the camera attached to the robot was successful. Three different samples of photos were used to evaluate the algorithm. After adding a data set of 45 images from the target users to the neural network, 20 new photos were used to test the accuracy of the system. Tests were performed from different perspective points, illumination, poses, and expressions; results showed facial recognition to have an accuracy of 90%. In order to achieve better results, a bigger dataset sample could be added to the pre-trained neural network used.

Results regarding the indoor localization tests using three Wi-Fi APs were not accurate enough, an average drift error of around 10 metres was measured, which means that the Wi-Fi technology was not suitable to determine the position of the patient successfully as the length of the corridors were about 10 metres. Therefore, a new Sprint was settled, but three iBeacons were used instead in order to obtain the distance to the scanning device. The results showed that Bluetooth technology performed better than Wi-Fi for indoor positioning, and so the patient could be located in an environment with an error of 2 metres approximately; this technology was concluded to be the best one for this project. A set of conditions was then established to ensure that the obtained target position stands within the working area; consequently, the average drift error decreased to a value of 0.88 metres.

The implementation of the Android App was satisfactory and works as expected, allowing the patient to choose between the displayed options and send the choice to the ROS master publishing it into a topic. The App allows the patient to control the robot movement or get assistance from the emergency services just by pressing a button.

A final evaluation was designed in order to test the mentioned features all together; these tests consisted of different scenarios that had both dynamic and static obstacles, and results showed the robot was able to recognize the patient and reach the target position most of the times. In other words, the robot was able to locate the patient indoors and reach his position avoiding the obstacles in real-time and recalculating the path if necessary.

9. Conclusions

The main contribution of this thesis is the development of an autonomous medical robot able to reach the patient in an indoor environment and to recognize it. This project addresses the problem of indoor navigation with the use of the LiDAR sensor on the TurtleBot3 Burger and indoor positioning using Bluetooth technology. ROS was the primary software for the development of the whole project, and OpenCV was used for facial recognition working with a neural network.

This thesis undertakes indoor navigation using SLAM. Indoor positioning can be used on a map of the environment obtained using SLAM methods where the robot will travel and self-locate with a position error of 0.3 metres. Results prove that the Bluetooth technology (iBeacon), as well as the trilateration algorithm, are suitable choices for detecting a human in an indoor environment as a successful average drift error of 0.83 metres was obtained for indoor localization. For eight out of the ten tests taken for the final evaluation, the robot was able to reach the target without difficulties by performing obstacle avoidance and was also capable of identifying the patient once it was near the user. Results show the facial recognition system achieved an accuracy of around 90% using a training dataset of 45 images. A communication interface was implemented by using an Android App that allows the patient to control the robot's movements or get assistance from the emergency services. Once the target position has been reached, the patient is able to send the robot back to the home position or to a new position inside the working area.

It is concluded that this project can be considered as the first step in the implementation of a totally autonomous medical robot ready for domestic use. The proposed idea is destined for pill delivery to disabled patients who prefer to stay in their own home rather than choosing to live in the special houses provided by the country. This will eventually reduce the demand for healthcare personnel that is currently the cause of problems such as high waiting time for assistance, which especially affects the elderly.

10. Future Work

This project has a great potential for growth. Although all the objectives have been successfully completed, some future improvements can be considered in order to improve this robot as a real autonomous medical robot for drug delivery to people with reduced mobility.

- Implement the signal scanning on the smartphone device in order to transfer this data to the ROS master remotely so that the triangulation algorithm can be completed. There would be no need for the patient to have the computer with him.
- Provide the robot with the capability of locating two or more patients living in the same house simultaneously. The use of facial recognition would be very useful as the robot would be able to disregard the patient if it is not the correct target.
- Improve the navigation and indoor localization so that the robot would be able to operate in houses of two or more floors; it is a necessary characteristic for people who live in big houses with several floors. The positioning could be performed by using the *z* coordinate when calculating the triangulation values.
- Implement a component to allow the robot to deliver the drugs to the patient (robotic arm, crane, deposit…).
- Improve the appearance of the robot in order to make it more user-friendly and socially acceptable.

References

[1] Z. Tomlinson, "15 Medical Robots That Are Changing the World," 11 October 2018. [Online]. Available: https://interestingengineering.com/15-medical-robots-that-are-changing-the-world. [Accessed March 2020].

[2] Sweden.se, "Healthcare in Sweden," 2019. [Online]. Available: https://sweden.se/society/health-care-in-sweden/. [Accessed February 2020].

[3] Sweden.se, "Health care in the home," 2019. [Online]. Available: https://imagebank.sweden.se/health+care+in+the+home/2599. [Accessed February 2020].

[4] Estocolmo (AFP), "En Suecia, uno de los países con mejor sistema de salud, faltan enfermeras," 2018. [Online]. Available: https://www.france24.com/es/20180904-en-suecia-uno-de-los-paises-con-mejor-sistema-de-. [Accessed February 2020].

[5] I. Borowy, Defining Sustainable Development for Our Common Future, 2014.

[6] B. M. M. &. O. G. Hopwood, "Sustainable development: mapping different approaches," 2005.

[7] Robot Platform, "Wheeled Robots," [Online]. Available: http://www.robotplatform.com/knowledge/Classification_of_Robots/Types_of_robot_wheels.html. [Accessed February 2020].

[8] J. M. R. H. F. F. L. K. A. R. a. R. A. T. Y. Silas F. R. Alves, "Conceptual Bases of Robot Navigation Modeling, Control and Applications, Advances in Robot Navigation," 5 July 2011. [Online]. Available: https://www.intechopen.com/books/advances-in-robot-navigation/conceptual-bases-of-robot-navigation-modeling-control-and-applications. [Accessed February 2020].

[9] K. Goris, "Autonomous Mobile Robot Mechanical Design," 2005. [Online]. Available: http://mech.vub.ac.be/multibody/final_works/ThesisKristofGoris.pdf. [Accessed February 2020].

[10] K. S. M. A.A. A. Razak A.H.Abdullah, "Mobile robot structure design, modeling and simulation for confined space application," September 2016. [Online]. Available: https://www.researchgate.net/publication/313539743_Mobile_robot_structure_design_modeling_and_simulation_for_confined_space_application. [Accessed February 2020].

[11] Vex Robotics, "Traction Wheels & Tread," [Online]. Available: https://www.vexrobotics.com/traction-wheels.html. [Accessed February 2020].

[12] Takigen, "Door Wheel," [Online]. Available: https://www.takigen.com/products/list/16070. [Accessed February 2020].

[13] Wikipedia, "Omni wheel," [Online]. Available: https://en.wikipedia.org/wiki/Omni_wheel. [Accessed February 2020].

[14] Vex Robotics, "Mecanum Wheels," [Online]. Available: https://www.vexrobotics.com/mecanum-wheels.html. [Accessed February 2020].

[15] Á. G. G. R. a. A. G. Rodríguez, in *Advanced Mechanics in Robotic Systems, Mobile Robots*, 2011, p. 43.

[16] CFP Robotic Group, "Ballbot Robot: A Single-Wheeled Balancing Robot," [Online]. Available: https://robot.cfp.co.ir/en/newsdetail/378. [Accessed February 2020].

[17] G. C. a. W. Chung, "Robot Structures," [Online]. Available: http://home.deib.polimi.it/gini/robot/docs/wheeled.pdf. [Accessed February 2020].

[18] Wikipedia, "Microcontrolador," [Online]. Available: https://es.wikipedia.org/wiki/Microcontrolador#:~:text=Un%20microcontrolador%20(abreviado%20%CE%BCC%2C%20UC,que%20cumplen%20una%20tarea%20espec%C3%ADfica.. [Accessed March 2020].

[19] opensource.com, "What is a Raspberry Pi?," [Online]. Available: https://opensource.com/resources/raspberry-pi. [Accessed March 2020].

[20] Arduino, "What is Arduino?," [Online]. Available: http://arduino.cc/en/guide/introduction. [Accessed March 2020].

[21] Sparkfun, "What is an Arduino?," [Online]. Available: https://learn.sparkfun.com/tutorials/what-is-an-arduino/all. [Accessed March 2020].

[22] L. Ruggeri, "Asus Tinker Board Vs Raspberry Pi," 2017. [Online]. Available: https://www.open-electronics.org/asus-tinker-board-vs-raspberry-pi/. [Accessed March 2020].

[23] Wikipedia, "Asus Tinker Board," [Online]. Available: https://en.wikipedia.org/wiki/Asus_Tinker_Board. [Accessed March 2020].

[24] R. S. a. I. Nourbakhsh, Autonomous Mobile Robots, vol. Chapter 3, EPFL, CMU.

[25] "E6A2-CW5C 100P/R 2M 128501 OMRON Incremental," [Online]. Available: https://www.electricautomationnetwork.com/en/omron/rotary-encoder-omron-e6a2-cw5c-100-128501. [Accessed March 2020].

[26] DigiKey, "PING (Parallax Inc.)," [Online]. Available: https://www.digikey.se/product-detail/en/parallax-inc/28015/28015-ND/1774419. [Accessed March 2020].

[27] A. S. G. V. P. B. P. M. Torres Medina, Robots y Sistemas Sensoriales, 2005.

[28] Robu, "Sharp Distance Measuring Sensor," [Online]. Available: https://robu.in/product/sharp-distance-measuring-sensor-unit-10-80-cm-gp2y0a21yk0f/. [Accessed March 2020].

[29] I. Dobriakov, "Design and Development of a ROS Based LiDAR Platform to Scan Environment for a Mobile Robot's Autonomous Motion," 2019. [Online]. Available: https://www.theseus.fi/bitstream/handle/10024/221992/Dobriakov_Ilia.pdf?sequence=2&isAllowed=. [Accessed February 2020].

[30] "RPLIDAR A3M1 Laser Range Scanner (360°)," [Online]. Available: https://www.generationrobots.com/en/402914-rplidar-a3m1-laser-range-scanner-360%C2%B0.html. [Accessed March 2020].

[31] K. Nesland, "Mobile Robot for Inspection on an Unmanned Offshore Production Platform," June 2019.

[32] E. Optics, "Understanding Camera Sensors for Machine Vision Applications," [Online]. Available: https://www.edmundoptics.com/knowledge-center/application-notes/imaging/understanding-camera-sensors-for-machine-vision-applications/. [Accessed March 2020].

[33] G. G. F. A. Fraile Mora, Instrumentación Aplicada a la Ingeniería, 2012.

[34] "e-CAM40_CUMI4682_MOD - 4 MP OV4682 RGB IR Camera Module," [Online]. Available: https://www.e-consystems.com/OV4682-RGB-IR-MIPI-CAMERA-Module.asp. [Accessed March 2020].

[35] S. A. A. A.-S. A. A. H. S. A.-K. A. A. a. M. A. Mai A. Al-Ammar, "Comparative Survey of Indoor Positioning Technologies, Techniques, and Algorithms," International Conference on Cyberworlds, 2014.

[36] J. P. G.-V. C. E. G.-T. D. M.-R. C. V.-R. a. J. F. Ramon F. Brena, "Evolution of Indoor Positioning Technologies: A Survey," 29 March 2017. [Online]. Available: https://www.hindawi.com/journals/js/2017/2630413/. [Accessed April 2020].

[37] S. B. a. H. Haas, "Pedestrian dead reckoning: A basis for personal positioning. In Proceedings of the 3rd Workshop on Positioning, Navigation and Communication," 2006.

[38] M. S. Svalastog, "Indoor positioning-technologies, services and architectures," 2007.

[39] Wikipedia, "Ultrawideband," 2020. [Online]. Available: https://es.wikipedia.org/wiki/Ultrawideband. [Accessed March 2020].

[40] M. Rouse, "IR wireless (infrared wireless)," September 2005. [Online]. Available: https://searchmobilecomputing.techtarget.com/definition/IR-wireless. [Accessed March 2020].

[41] Digi International Inc, "Zigbee Wireless Mesh Networking," [Online]. Available: https://www.digi.com/resources/standards-and-technologies/zigbee-wireless-mesh-networking. [Accessed March 2020].

[42] Technopedia, "Wireless Local Area Network (WLAN)," 2018. [Online]. Available: https://www.techopedia.com/definition/5107/wireless-local-area-network-wlan. [Accessed March 2020].

[43] RightsLink, "How does Bluetooth work?," Scientific America, a Division of Springer Nature America, Inc., 5 Nov 2007. [Online]. Available: https://www.scientificamerican.com/article/experts-how-does-bluetooth-work/. [Accessed March 2020].

[44] A. L. a. I. N. Y. Gu, "A survey of indoor positioning systems for wireless personal networks.," Communications Surveys & Tutorials, IEEE, 2009.

[45] IoT For All, Leverege, "Trilateration vs. Triangulation for Indoor Positioning Systems," 2019.

[46] R. N. a. M. I. Z. Farid, "Recent advances in wireless indoor localization techniques and system," Journal of Computer Networks and Communications, 2013.

[47] J. Zürn, "Robot localization with Kalman-Filters and landmarks," *Medium,* 2018.

[48] Wikipedia, "Monte Carlo localization," 4 July 2019. [Online]. Available: https://en.wikipedia.org/wiki/Monte_Carlo_localization. [Accessed March 2020].

[49] Wikipedia, "Bayesian inference," 15 March 2020. [Online]. Available: https://en.wikipedia.org/wiki/Bayesian_inference#Computer_applications. [Accessed March 2020].

[50] M. F. Rome, "Navigation stack test: GMapping vs Hector Slam," 1 April 2015. [Online]. Available: http://www.geduino.org/site/archives/36. [Accessed March 2020].

[51] R. B. V. M. Samaahita S Belavadi, "Frontier Exploration Technique for 3D Autonomous SLAM Using K-means Based Divisive Clustering," Bengaluru, India, 2017.

[52] E. Marder-Eppstein, "Navigation," ROS.org, [Online]. Available: http://wiki.ros.org/navigation. [Accessed March 2020].

[53] F. H. W. L. Lyle Parungao, "Dijkstra algorithm based intelligent path planning with topological map and wireless communication," Philippines, 2018.

[54] J. Procházková and D. Martišek, "Notes on Iterative Closest Point Algorithm," 13 04 2013. [Online]. Available:

https://www.researchgate.net/publication/324500004_Notes_on_Iterative_Closest_Point_Algorithm. [Accessed April 2020].

[55] ROS.org, "About ROS," [Online]. Available: https://www.ros.org/. [Accessed March 2020].

[56] MathWorks, "MATLAB Product Description," [Online]. Available: https://es.mathworks.com/help/matlab/learn_matlab/product-description.html?lang=en. [Accessed March 2020].

[57] OpenCV, "About," [Online]. Available: https://opencv.org/about/. [Accessed March 2020].

[58] Waveshare, "AlphaBot2 robot building kit for Raspberry Pi 3 Model B," [Online]. Available: https://www.waveshare.com/alphabot2-pi.htm. [Accessed March 2020].

[59] Waveshare Wiki, "AlphaBot2-Pi," [Online]. Available: https://www.waveshare.com/wiki/AlphaBot2-Pi. [Accessed March 2020].

[60] Pantech Solutions, "SLAM Robot," [Online]. Available: https://www.pantechsolutions.net/slam-robot. [Accessed March 2020].

[61] Nevon Projects, "LiDAR based Self Driving Car," [Online]. Available: https://nevonprojects.com/lidar-based-self-driving-car/. [Accessed March 2020].

[62] Husarion, "Husarion ROSbot 2.0," [Online]. Available: https://store.husarion.com/products/rosbot. [Accessed March 2020].

[63] Robotis, "TurtleBot3," [Online]. Available: https://www.turtlebot.com/. [Accessed March 2020].

[64] RoboSavvy, "TurtleBot3 Waffle Pi," [Online]. Available: https://robosavvy.com/store/turtlebot3-waffle-pi.html. [Accessed March 2020].

[65] RosComponents, "TurtleBot3 Burger," [Online]. Available: https://www.roscomponents.com/es/robots-moviles/214-turtlebot-3.html. [Accessed March 2020].

[66] L. Galtarossa, "Obstacle Avoidance Algorithms for Autonomous Navigation system in Unstructured Indoor areas," Torino, 2018.

[67] A. Hussein, "Control and Communication Systems for Automated Vehicles Cooperation and Coordination," Leganés, Madrid, 2018.

[68] V. Lappyová, "Development and implementation of human detection and following method on a mobile robot," Oslo, 2019.

[69] F. Jusuf, "Auto-Navigation For Robots. Implementation of ROS," Helsinki, 2016.

[70] B. P. K. L. S. N. P. V. P. Mrs. Madhuram.M, "Face Detection and Recognition Using OpenCV," Ramapuram, Chennai, India, 2018.

[71] B. Kiyimba, "Artificially Intelligent Forensic Face Detection and Identification System Using Open Computer Vision and Deep Learning (AIFFD)," Kampala, 2019.

[72] V. P. Shervin EMAMI, "Facial Recognition using OpenCV," Romania.

[73] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): part II," Semptember 2006. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1678144/authors#authors. [Accessed April 2020].

[74] P. V. K. B. A. R. R. G. C. A. E. Andreas Pfrunder, "Real-time autonomous ground vehicle navigation in heterogeneous environments using a 3D LiDAR," September 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8206083/authors#authors. [Accessed April 2020].

[75] O. N. Johan Alexandersson, "Implementation of SLAM algorithms in a small-scale vehicle using model-based development," Linköping, Sweden, 2017.

[76] F. H. Hampus Engström, "Evaluation and testing of techniques for indoor positioning," Lund, Sweden, 2015.

[77] P. P. R. P. M. T. OnkarPathak, "Wi-Fi Indoor Positioning System Based on RSSI Measurements from Wi-Fi Access Points - A Tri-lateration Approach," April 2014. [Online]. Available: https://www.ijser.org/researchpaper/Wi-Fi-Indoor-Positioning-System-Based-on-RSSI-Measurements.pdf. [Accessed April 2020].

[78] J. L. W. S. a. F. Y. Baichuan Huang, "A Robust Indoor Positioning Method based on Bluetooth Low Energy with Separate Channel Information," 22 June 2019. [Online]. Available: https://www.mdpi.com/journal/sensors. [Accessed May 2020].

[79] a. J. T.-S. a. J. H. Germán Martín Mendoza-Silva, "A Meta-Review of Indoor Positioning Systems," Castellón, 2019.

[80] J. S. Ken Schwaber, "The Scrum Guide," November 2017. [Online]. Available: https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100. [Accessed February 2020].

[81] nutcache, "What is Scrum? Methodology and Project Management," [Online]. Available: https://www.nutcache.com/blog/what-is-scrum-methodology-and-project-management/. [Accessed February 2020].

[82] Robotis, "e-Manual TurtleBot3," [Online]. Available: http://emanual.robotis.com/docs/en/platform/turtlebot3/overview/#turtlebot. [Accessed May 2020].

[83] A. Geitgey, "Modern Face Recognition with Deep Learning," 2016. [Online]. Available: https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78. [Accessed May 2020].

[84] NVIDIA, "CUDA Zone," [Online]. Available: https://developer.nvidia.com/cuda-zone. [Accessed May 2020].

[85] D. King, "High Quality Face Recognition with Deep Metric Learning," 2017. [Online]. Available: http://blog.dlib.net/2017/02/high-quality-face-recognition-with-deep.html. [Accessed May 2020].

[86] A. Rosebrock, "Face recognition with OpenCV, Python, and deep learning," 2018. [Online]. Available: https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/. [Accessed May 2020].

[87] ROS.org, "Converting between ROS images and OpenCV images (Python)," [Online]. Available: http://opensource-robotics.tokyo.jp/ros.org/wiki.ros.org/cv_bridge(2f)Tutorials(2f)ConvertingBetweenROSImagesAndOpenCVImagesPython.html. [Accessed May 2020].

[88] W. J. K. V. E. T. L. M. David Plets, "Simple Indoor Path Loss Prediction Algorithm and Validation in Living Lab Setting," Ghent, Belgium.

[89] K. Solutions, "The Beacon War – Know what is a iBeacon, a Eddystone and a AltBeacon," 2 January 2017. [Online]. Available: https://www.k2bindia.com/ibeacon-vs-eddystone-vs-altbeacon/. [Accessed May 2020].

[90] J. Makadia, "What are the key differences between iBeacon, Eddystone & AltBeacon?," 9 Jan 2018. [Online]. Available: https://www.quora.com/What-are-the-key-differences-between-iBeacon-Eddystone-AltBeacon. [Accessed April 2020].

[91] ROS, "rosjava - ROS," [Online]. Available: http://wiki.ros.org/rosjava. [Accessed May 2020].

[92] ROS.org, "android - ROS," [Online]. Available: http://wiki.ros.org/android. [Accessed May 2020].

[93] Android, [Online]. Available: https://developer.android.com/. [Accessed May 2020].