Development and Simulation of a Low-Cost Ground-Truth Localization System for Mobile Robots

Gustavo Soares de Moura - 42821

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Industrial Engineering.

Work oriented by:

José Luís Sousa de Magalhães Lima

Paulo José Cerqueira Gomes da Costa

Wagner Endo

This dissertation does not include the criticisms and suggestions made by the jury

Bragança

2020

Development and Simulation of a Low-Cost Ground-Truth Localization System for Mobile Robots

Gustavo Soares de Moura - 42821

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Industrial Engineering.

Work oriented by:

José Luís Sousa de Magalhães Lima

Paulo José Cerqueira Gomes da Costa

Wagner Endo

Bragança

2020

Acknowledgement

To Professor PhD José Lima, supervisor of this thesis, I would like to express my thanks for all his support, guidance and availability.

To Professor PhD Paulo Costa, co-supervisor of this thesis, I would like to thank him for his assistance and availability.

To Professor PhD Wagner Endo for having accepted to be a distance co-supervisor in this Dual Diplomacy program.

To my friend Pedro Borghi, for his friendship and companionship throughout this journey.

To Carolina, I want to express a special thanks for the support and encouragement at all times.

Finally, a very special thanks to my parents who accompanied me at all times, encouraged me and offered all the necessary support. Without them, this work would not be possible.

Abstract

Ground-truth systems are extremely useful in the field of robotics, providing accurate external data that is used as a reference in the validation of embedded localization systems in mobile robots. However, ground-truth systems generally have a high cost. The present work aims at the development of a low-cost ground-truth system capable of determining the location and orientation of mobile agents using a laser rangefinder sensor and a monocular camera. The methodology created is applicable to several types of mobile wheeled robots, provided that it is possible to fix a cylindrical target with colored markers to the robot. The developed system and methodology were implemented and validated through SimTwo, a realistic simulator. The results obtained by the proposed system were compared with the actual data (provided by the simulator), demonstrating that the developed solution meets the proposed objectives. Finally, possible improvements are highlighted to be implemented in future works.

Keywords: Ground-truth, mobile robotics, localization systems, SimTwo.

Resumo

Sistemas ground-truth são extremamente úteis no campo da robótica, fornecendo dados externos precisos que são usados como referência na validação de sistemas de localização embebidos em robôs móveis. No entanto, sistemas ground-truth geralmente apresentam um custo elevado. O presente trabalho visa o desenvolvimento de um sistema ground-truth de baixo custo capaz de determinar a localização e orientação de agentes móveis por meio de um sensor laser rangefinder e uma câmera monocular. A metodologia criada é aplicável a diversos tipos de robôs móveis terrestres, bastando que seja possível fixar ao robô um alvo cilíndrico com marcadores coloridos. O sistema e a metodologia desenvolvida foram implementados e validados através do SimTwo, um simulador realista. Os resultados obtidos pelo sistema proposto foram comparados com os dados reais (fornecidos pelo simulador), demonstrando que a solução desenvolvida atende aos objetivos propostos. Por fim, são ressaltadas possíveis melhorias a serem implementadas em trabalhos futuros.

Palavras-chave: Ground-truth, robótica móvel, sistemas de localização, SimTwo.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Ground-truth systems are extremely useful in the field of robotics, providing accurate external data that is used as a reference for validating localization systems in mobile robots. However, ground-truth systems generally have a high cost, which can be a limitation to many projects. This thesis addresses the development of a low-cost ground-truth system capable of determining the location and orientation of mobile robots using a laser rangefinder sensor and a monocular camera.

The ability to locate itself accurately is essential for mobile robots: any inaccuracies can place objects in the environment, the robot and the lives of others at risk, and can cause financial losses and even irreparable damage. It is through the data collected by the distance sensors that decisions are made about steering, brake and speed control [1]. To validate the correct functioning of the embedded localization system, an external ground-truth system is used as a reference, providing real position and orientation data of the robot.

It is common to use markers in robotics projects that involve cameras. Through image processing, it is possible to identify the markers in order to allow the recognition of objects/robots and obtain important information. For example, a square object to be manipulated by a robot may have different colored markers on each of its faces, so that it is possible to recognize which face is facing up. The system to be developed in the present work will use markers that will be strategically placed on a cylindrical target, in order to

facilitate the estimation of its orientation.

As mentioned in [2], in addition to being used in mobile robotics, marker-based optical movement measurement systems have several other applications, such as in physiotherapeutic treatments aimed at the rehabilitation of patients with mobility loss [3], in the capture of movements for the production of films and games, studies of hydrodynamics in ships, studies of aerodynamics of airplanes in wind tunnels, among other applications.

As stated by [4], the development of projects in the area of robotics is a difficult task, because, in addition to the intellectual effort employed, it is also necessary to use specific hardware, which can cause logistical problems (if the materials in question are not available) and financial (if the material has a high cost). These problems can be enlarged when taking into account that the hardware can be damaged in an eventual accident during development. It is possible to work around these problems through the use of simulators. As long as it captures the essential aspects of reality [5], simulation is an excellent tool, bringing advantages, such as low development cost, access to variables of interest that would be difficult to monitor if working with real hardware [6] and facilitating making changes to the robot's structure without the need for effort in mechanical construction.

The developed system and methodology will be implemented and validated through simulation, with the data obtained by the proposed system being compared to the actual data (provided by the simulator). For that, SimTwo [7] will be used, a realistic simulator developed by Paulo Costa at the Faculty of Engineering of University of Porto.

1.1 Objectives

Initially, this work aimed to develop, in hardware, a laser scanning system capable of generating a two-dimensional point cloud of the objects around it. In addition, it was also intended to develop a software capable of processing and plotting the data obtained. The laser sensor used would be the Broadcom AFBR-S50MV85G [8]. However, due to the scarce documentation related to hardware, several difficulties were encountered in developing a firmware capable of communicating directly with the sensor, since the

manufacturer does not provide a way to access the data without the intermediation of the firmware and software provided by him. Due to the restrictions imposed by the COVID-19 pandemic, it was not possible to continue the hardware testing in the laboratory, so a new theme was proposed, related to the previous one and that could be implemented through simulations. Thus, the present work aims to:

- Develop a low-cost ground-truth system capable of determining the location and orientation of mobile robots;
- The developed solution must be applicable to different types of mobile wheeled robots;
- Implement and validate the system through simulation by comparing the results obtained with real values provided by the simulator.

1.2 Document Structure

This paper is organized in six chapters.

In the first chapter, the thesis topic is presented, introducing the subject and emphasizing the objectives of the work.

In the second chapter, a bibliographic review is conducted on the main localization sensors used in mobile robots, ground-truth systems and simulators for robotics.

In the third chapter, the simulator used is presented. In addition, the creation of the simulation scene is described, including the construction of the ground-truth system.

The fourth chapter describes the developed ground-truth system and the methodology used to determine the location and orientation of the robot;

The fifth chapter shows the results obtained with the developed system. To validate the system, the data obtained are compared with the real data (provided by the simulator);

Finally, the sixth chapter holds conclusions and suggestions for future works.

Chapter 2

Related Work

In this chapter, the different types of sensors used in localization systems are addressed and works related to the development of ground-truth systems are presented. In addition, some commercial systems that serve this purpose are listed. As the work to be developed will be implemented and validated through simulation, simulators focused on robotics are also addressed.

2.1 Localization Sensors

This section presents some of the main localization sensors used in robotics: ultrasonic sensors, radars, laser rangefinders, cameras, GPS and IMUs. Positioning systems often combine different types of sensors to ensure greater accuracy.

2.1.1 Ultrasonic Sensor

Ultrasonic sensors use sound waves between 20 kHz to 40 kHz to measure their distance to objects. Its operating principle is based on time-of-flight (ToF), the time interval between the emission of the sound wave and the reception of its echo. Therefore, the distance is calculated by:

$$d = \frac{c \times ToF}{2} \tag{2.1}$$

Where d is the distance between the sensor and the object, c is the speed of sound in meters per second and ToF is the time-of-flight, in seconds.

Ultrasonic sensors stand out for their low cost and for producing good results with objects of any material, of any color and in adverse weather conditions, such as fog or rain. It presents as a disadvantage the fact it has a blind region (intermediate point between the emitter and the receiver) when an object is positioned at a distance less than the recommended minimum. Figure 2.1 shows an ultrasonic sensor.

Figure 2.1: Ultrassonic sensor HC-SR 04 from Adafruit [9].

2.1.2 RADAR (Radio Detect and Ranging)

Radar systems work with wavelengths on the order of millimeters, being used in a wide variety of civil and military applications, from detecting aerial threats to monitoring the speed of vehicles on the roads.

The principle of operation of the radar is based on the time-of-flight, time interval between the emission of the radio wave and the reception of echo, with the distance being determined through equation 2.1, where the constant c must be the speed of light.

Radars stand out for being robust to adverse weather conditions. As a disadvantage, the fact of having a reduced field of view (FOV) stands out. Figure 2.2 shows a RADAR.

Figure 2.2: QT50R RADAR Series from Banner [10].

2.1.3 LIDAR (Light Detection and Ranging)

A ray of light incident on a surface can be reflected at a single angle (specular reflection) or at several other angles (diffuse reflection). Specular reflection occurs on smooth surfaces, while diffuse reflection occurs on rough surfaces [11]. Almost all objects around us reflect light diffusely on their surfaces, with only a few materials producing specular reflection, such as water, glass, transparent plastics and other specific materials.

LIDAR, also known as laser rangefinder (LRF), uses the phenomenon of diffuse reflection. The sensor emits a laser beam towards an object, which reflects part of that beam in the same direction as the sensor. The reflected beam is detected by photoreceptors and, based on the time-of-flight between beam emission and reception, it is possible to determine the distance between the sensor and the object [12]. The equation 2.1 is valid for this type of sensor, however in this case the constant c must be the speed of light.

To cover a large area, the LRF emits several pulses per second, with a rotating mechanism responsible for sweeping the beams around a desired area, in two or three dimensions [13]. In this way, a dense cloud of points (coordinates) of the environment around the sensor is obtained, enabling the elaboration of maps, which are essential for robot navigation.

Laser rangefinders are widely used for the localization of mobile robots, with the advantages of their high precision, high sampling rate, long detection range, ability to estimate the reflectance of surfaces and, as shown in [14], robustness to external lighting

conditions. However, laser sensors have limitations in identifying objects that are transparent or that produce specular reflections. Several techniques have been proposed to work around this problem, one of which is described by [15]. Figure 2.3 shows a LIDAR.

Figure 2.3: 2D LIDAR LMS500-21000 Lite from Sick [16].

2.1.4 Camera

It is possible to classify cameras in two types, based on the wavelength detected by the device: visible spectrum cameras (VSC) capture wavelengths between 400 nm and 780 nm, just like the human eye, while infrared spectrum cameras (IRSC) capture wavelengths greater than 780 nm, not visible to humans.

VSC cameras divide the visible spectrum into three channels, red (R), green (G) and blue (B). These devices are widely applied in autonomous vehicles due to their advantages: low cost, color detection capacity and high resolution. Among the disadvantages, it is possible to mention the fact that they produce a large volume of data, requiring a processing system at their level. In addition, they are highly affected by changes in lighting and adverse weather conditions, such as rain, snow and fog. For these reasons, VSC cameras are often combined with LRFs or radars to increase the robustness of the system.

IRSC cameras work in this spectrum because there is less light interference, being preferred over VSC in situations where changes in lighting occur (at the exit of a tunnel, for example). In addition, they can be used as thermal cameras, allowing view temperature differences.

In addition to the two types of cameras mentioned, there are also ToF cameras, which use the time-of-flight principle to allow a three-dimensional representation of objects in a scene. Infrared light pulses are emitted through a matrix of LEDs, with the distance of the points being calculated through the phase difference between the modulated signal emitted and the received one.

$$d = \frac{c}{2} \times \frac{\Delta\phi}{2\pi f_{\text{mod}}} \tag{2.2}$$

Where d is the distance between the sensor and the object, c is the speed of light, $\Delta\phi$ is the phase difference between the signal sent and the one received and f_{mod} is the frequency with which the signal is modulated. Figure 2.4 shows a camera.

Figure 2.4: MLX90640 thermal camera from Adafruit. Adapted from [9].

2.1.5 GPS (Global Positioning System)

The GPS system consists of orbiting satellites that emit signals with information about their position and orbital parameters. Through the time-of-flight between the signal sent by the satellite and the received signal, a receiver on Earth is able to infer its own position and speed. Unfortunately, satellite signals can be influenced by reflections, resulting in inaccurate measurements.

To minimize inaccuracies caused by reflections, the DGPS (Differential Global Positioning System) was created, which consists of two GPS receivers, one fixed and one mobile, known as a rover. The base station has its location known and continuously sends signal corrections to the rover, significantly increasing the localization accuracy.

GPS or DGPS based solutions are recommended only for outdoor environments, as indoors the signal is weak or non-existent due to obstruction of the line of sight between the receiver and the satellites [17]. Figure 2.5 shows a GPS.

Figure 2.5: GPS based on MAX-6 module from U-blox [18].

2.1.6 Inertial Positioning and Dead-Reckoning

Dead-Reckoning is the process of calculating the current position of a moving robot based on a predetermined position, using speed estimates. It is possible to estimate the speed of the robot using rotary encoders attached to the wheel, however this technique is prone to errors arising from slips or changes in the direction of the robot, causing cumulative errors in the position calculation. Because of this, dead-reckoning is usually complemented with Inertial Measurement Units (IMUs), such as accelerometers, gyroscopes or magnetometers, however cumulative errors are introduced due to the measurement of first and second order variables [19] and sensors drifts. In order to allow more precise positioning, it is common to make a fusion between the data from the IMUs and a GPS, as shown in [20]. Figure 2.6 shows an IMU.

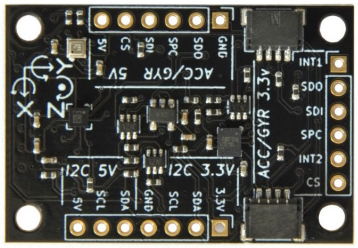


Figure 2.6: BerryIMU: contains gyroscope, accelerometer and magnetometer [21].

2.2 Ground-Truth Systems

Ground-truth systems allow to determine with high precision the position and/or orientation of robots, being frequently used as a reference to analyze the accuracy of sensors, algorithms and localization systems embedded in robots. Ground-truth systems are generally composed of a combination of sensors, with the most common being GPS, cameras and LRFs, allowing measurements without the need of contact.

As stated by [22], systems based on GPS (Global Positioning System) are often used as ground-truth, especially DGPS (Differential Global Positioning System), which have high positioning accuracy. However, indoors, the DGPS accuracy is affected, significantly decreasing or even ceasing completely, if the device is not under the satellite's line of sight. Through literature reviews it is possible to find works with promising results using only cameras and/or LRFs, for this reason this work will focus only on these two sensors.

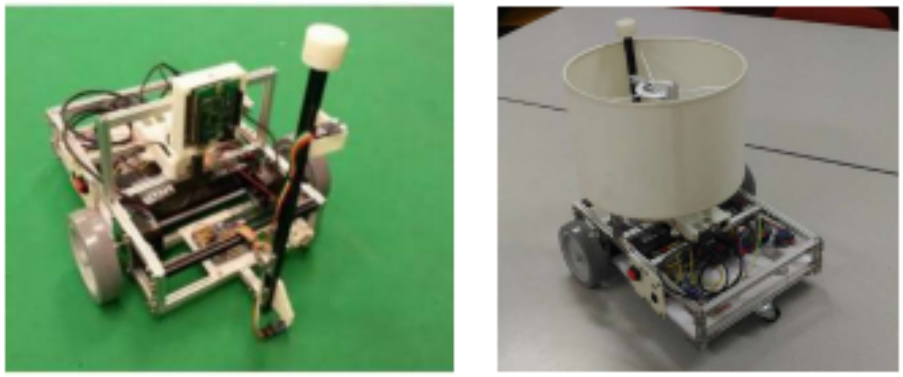
Currently, LRFs and cameras are widely applied in mobile robotics in order to detect obstacles and avoid collisions. They are used in autonomous vehicles [23] and in several applications that require SLAM (simultaneous localization and mapping), involving navigation of robots in unknown environments [12].

In [24] and [25] systems are proposed based on LRF, where the first is embedded in a mobile robot and makes use of reflective panels scattered on the wall of the scene to detect the location/orientation of the robot and the second consists in a system external to the robot and portable, facilitating its installation in different environments.

In [22] a low-cost system based on camera and printable markers is presented. In [26] it is presented a low-cost ground-truth system based on Kinect developed for use in the RoboCup, an international football competition aimed at humanoid robots.

In [2] it is proposed a system based on cameras, external to the robot and fixed, to detect the location and orientation of mobile robots with and without markers.

In [27] a system based on an LRF is proposed, that is also external to the robot and fixed, being able to determine the location of a cylindrical target. The approach employed is quite interesting, as it is applicable to different types of robots, as long as it is possible to fix a cylindrical target, as shown in figure 2.7.



(a) Without target. (b) With target.

Figure 2.7: Approach based on the use of a cylindrical target [27].

In view of the promising results achieved in the works mentioned above, the present work will use an LRF and a camera positioned at a fixed point in the scene to determine the position and orientation of a cylindrical target with markers.

2.2.1 Commercial Systems

This subsection aims to present some of the commercial systems used as ground-truth, providing accurate data to be used by localization algorithms. All the information presented was obtained from the manufacturers' websites.

The Arqus A5 from Qualysis [28] shown in figure 2.8 is a motion capture camera

capable of tracking passive and active markers. The camera is suitable for indoor and outdoor environments, presenting a resolution of 2560x1920 pixels at 700 FPS and in indoor environments it is capable of detecting a 16 mm passive marker up to a maximum distance of 26 m.

Figure 2.8: Arqus A5 from Qualysis [28].

The Prime^{x} 41 from OptiTrack [29] shown in figure 2.9 is also a motion capture camera capable of tracking passive and active markers. The camera is suitable for indoor and outdoor environments, presenting a resolution of 2048x2048 pixels at 180 FPS and in indoor environments it is able to detect a 14 mm passive marker up to a maximum distance of 30 m.

Figure 2.9: Prime^{x} 41 from OptiTrack [29].

The LMS101-10000 from Sick [16] shown in figure 2.10 is a two-dimensional LIDAR. It is indicated only for indoor environments, having a maximum range of 20 m, a field of view of 270º with an angular resolution of 0.25º and takes 20 ms to perform a complete scan. It has an accuracy of 30 mm.

Figure 2.10: LMS101-1000 from Sick [16].

The UTM-30LX from Hokuyo [30] shown in figure 2.11 is a two-dimensional LIDAR. It is indicated for indoor and outdoor environments, having a maximum range of 30 m, a field of view of 270º with an angular resolution of 0.25º and takes 25 ms to perform a complete scan. It has an accuracy of 30 mm for distances between 0.1 m and 10 m and 50 mm for distances between 10 m and 30 m.

Figure 2.11: UTM-30LX from Hokuyo [30].

The RT3003 v3 from OXTS [31] shown in figure 2.12 is a localization system composed of IMUs and GPS. It is indicated for indoor and outdoor environments, presenting an accuracy of 1 cm for positioning and 0.150º for orientation and inclination.

Figure 2.12: RT3000 v3 from OXTS [31].

2.3 Robotic Simulators

As said by [4], thanks to the advances in the field of computer science, several tools are being created in terms of developing simulation/visualization tools. Currently, there is a plethora of simulation software capable of assisting in the design of automobiles, civil construction, robotics and several other industries. Simulation softwares represents a simple and economical way to validate complex systems, in order to make possible research projects that would not happen without them, due to limited resources and financial restrictions. In relation to robotics, simulation allows the production and validation of robot software even without access to the robot's hardware, allowing quick reconfigurability and access to many variables of interest, which would be difficult to monitor if working with real hardware [6].

In [4] an analysis is made of which 3D simulators are currently most used by the robotics community. To select which simulators are most used, the number of scientific works found in Google Scholar that mention and/or develop their work using each 3D simulator was used as a criterion. With that, it was possible to conclude that currently the three most popular simulators are: Unity, Gazebo and V-Rep. Analyzing the works, it was found that Unity is widely used in the production of games, while V-Rep and Gazebo are more applied in robotics projects. As this thesis is solely interested in simulators applied to robotics, emphasis will be given only to the last two. In addition, Webots, another widely used simulator, will also be mentioned. The three simulators presented

are available for Windows, Linux and MacOS.

Gazebo is an open source 3D robotics simulator that is completely free [32]. Among its main features are: support for four physics engines (ODE, Bullet, Simbody and DART), simulation of several types of sensors (laser rangefinders, 2D/3D cameras, Kinect style sensors, contact and force sensors, etc.) and possibility of integration with ROS (Robot Operating System), a framework for application development for robots [33]. Thanks to its popularity and the fact that it is an open source project, it has extensive documentation and several libraries of scene elements available. Figure 2.13 shows a project developed based on Gazebo.

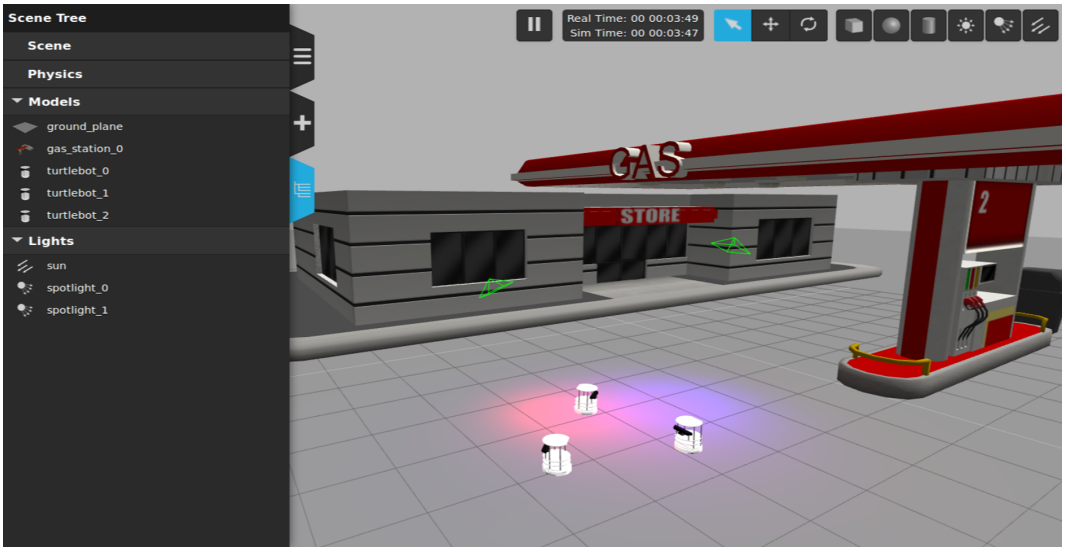


Figure 2.13: Project developed based on Gazebo [32].

V-Rep is a 3D robotics simulator that does not have open source, but presents a free version for non-commercial purposes. Among its main features are: support for four physics engines (ODE, Bullet, Vortex and Newton), simulation of several types of sensors (laser rangefinders, 2D/3D cameras, Kinect style sensors, contact and force sensors, etc.), and the possibility of integration with ROS. As stated in [34], V-Rep was discontinued in November 2019, being replaced by CoppeliaSim, which is fully compatible with V-Rep, but faster and with more features. Figure 2.14 shows a project developed based on V-Rep.

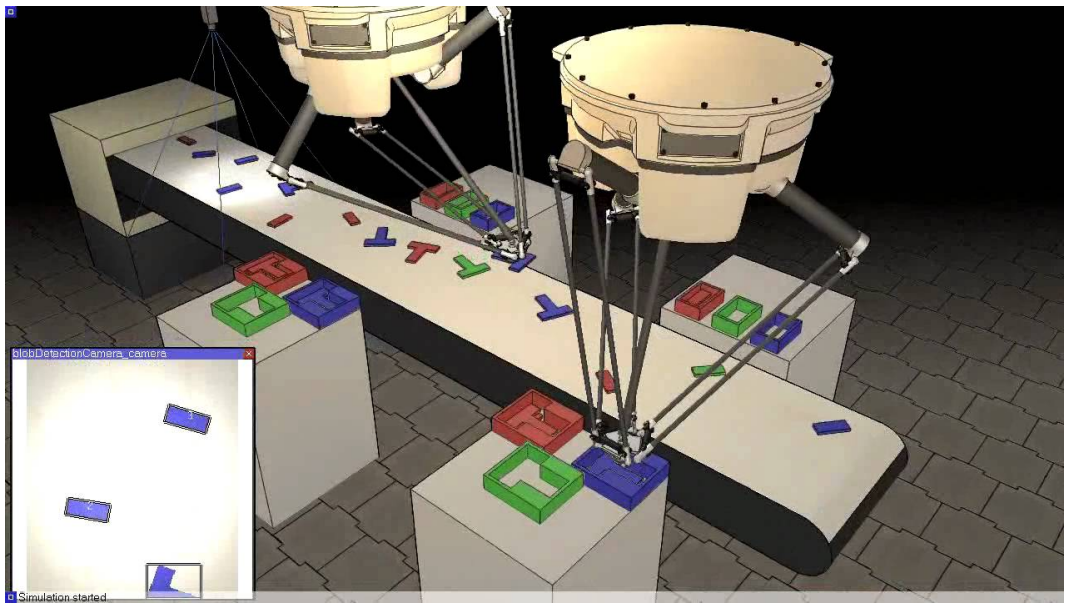


Figure 2.14: Project developed based on the V-Rep [34].

In addition to [4], it is possible to check comparisons between Gazebo and V-Rep in [35] and in [36]. Both simulators have easy integration with ROS and a diversity of physical engines, however all of the works put the V-Rep in prominence due to the fact that it has a more user-friendly and intuitive interface, in addition to offering several useful features for editing scenes.

The third simulator presented is Webots, a free and open source robotics simulator [37]. Among its main features are: support for a single physics engine (ODE), simulation of several types of sensors (distance sensors, cameras, laser rangefinders, radars, etc.) and the possibility of integration with ROS. Figure 2.15 shows a project developed based on Webots.

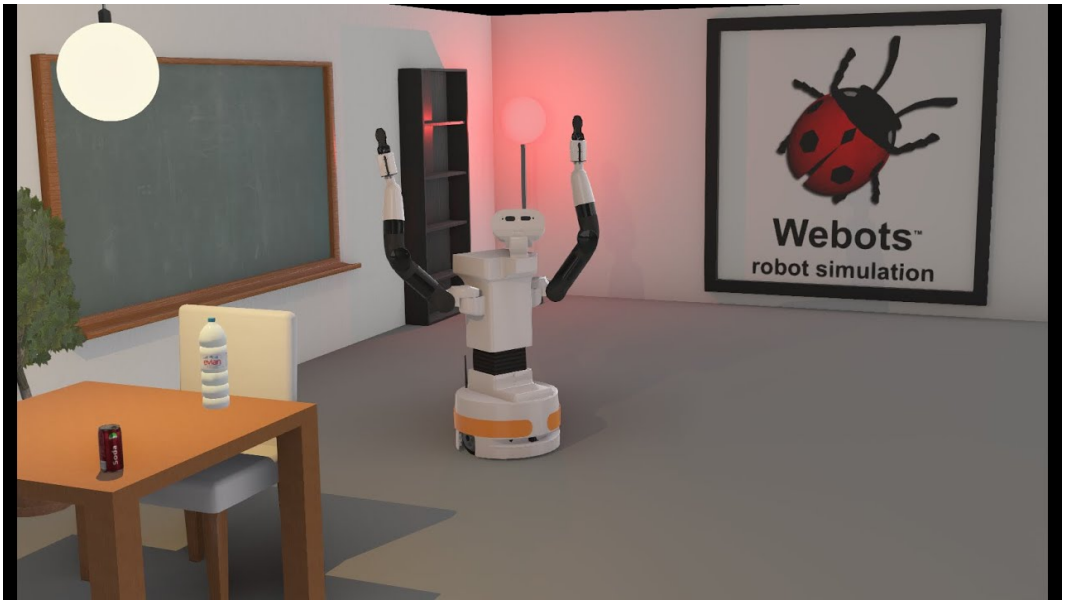


Figure 2.15: Project developed based on Webots [37].

Although the three simulators mentioned are the most popular, the present work will use SimTwo, which will be presented in the next chapter. The choice is due to the involvement of supervisors of this work with the development of the simulator. In addition, it will be an opportunity to carry out tests with the camera, a feature recently added to the simulator. SimTwo has great potential, as it is capable of simulating several applications in robotics, being possible to configure several simulation parameters.

Chapter 3

Simulation Environment

In this chapter the simulator used is presented, explaining its characteristics, the functions of each of the different windows found in the simulator and the elements used to create scenes. Finally, the creation of the simulation scene will be detailed, including the construction of the ground-truth system.

3.1 Introduction to SimTwo

SimTwo is a 3D simulator developed by Paulo Costa, professor at the Faculty of Engineering of the University of Porto and researcher at INESC TEC [38]. According to him, "SimTwo is a realistic simulation system that can support several types of robots. Its main objective is the simulation of mobile robots that may have wheels or legs, although industrial robots, conveyor belts and vehicles that are lighter than air can also be defined. Basically, any type of definable terrestrial robot with rotating joints and/or wheels can be simulated in this software" [39]. SimTwo is free and open source. Currently has only version for Windows.

The software was developed in 2008 and is constantly updated to receive new features [7]. The simulator offers a wide variety of sensors and actuators, as well as the possibility of using remote clients or communicating with any external hardware via the RS-232 serial port. All of these tools allow the development of several scene configurations. Some of

the works developed using SimTwo include humanoid robots [5], omnidirectional robots [6] and [40], air vehicles [41] and games applications [42].

SimTwo is based on several Open Source libraries, such as:

- GLScene: OpenGL-based library that allows the rendering of 3D scenes [43];
- ODE: Physics engine that allows the simulation of rigid body dynamics and collision detection [44];
- Pascal Script: Library composed of a set of units that can be compiled in its executable, eliminating the need to distribute external files [45];
- SynEdit: Library that allows the implementation of the script editor [46];
- OmniXML: Library that supports XML language [47];
- Rx-Lib: Library that allows the construction of user interfaces [48].

3.2 Simulator Structure

This section explains the function of each of the windows found in the simulator.

3.2.1 Scene Visual

Window where it is possible to view the 3D simulation, making it possible for the user to interact with the entities in the scene. If during the simulation any changes are made to the files, the simulation must be rebuilt for the change to take effect.

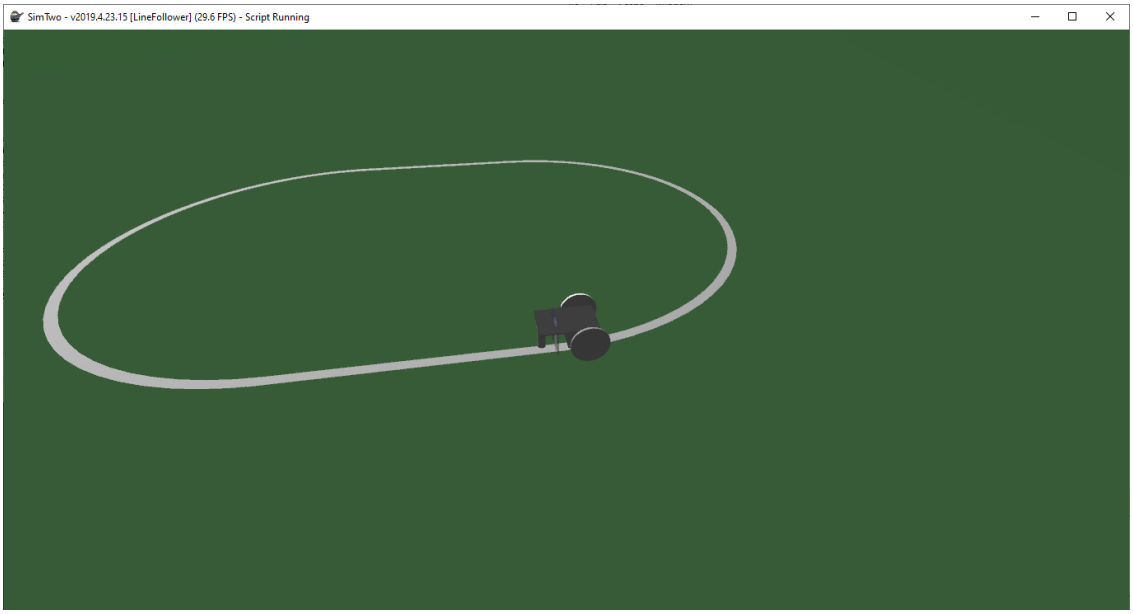


Figure 3.1: Scene Visual.

3.2.2 Scene Editor

In the Scene Editor it is possible to create robots, tracks, walls and any other entities in the scene. It consists of the main file 'scene.xml' and other XML files. Each entity in the scene must have an XML file with its physical description, each of these files being called in the main file.

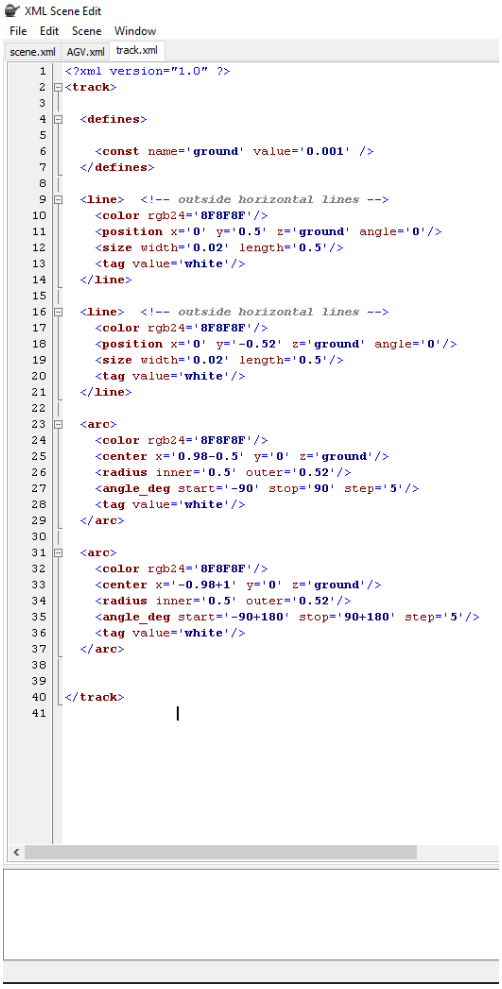


Figure 3.2: Scene Editor.

3.2.3 Control Editor

Control Editor is a Pascal editor through which it is possible to process the signals obtained by the sensors, define the motor controllers and implement the robot's control logic. From this window, the value of any variable can be sent to Sheet (explained below) for user viewing.

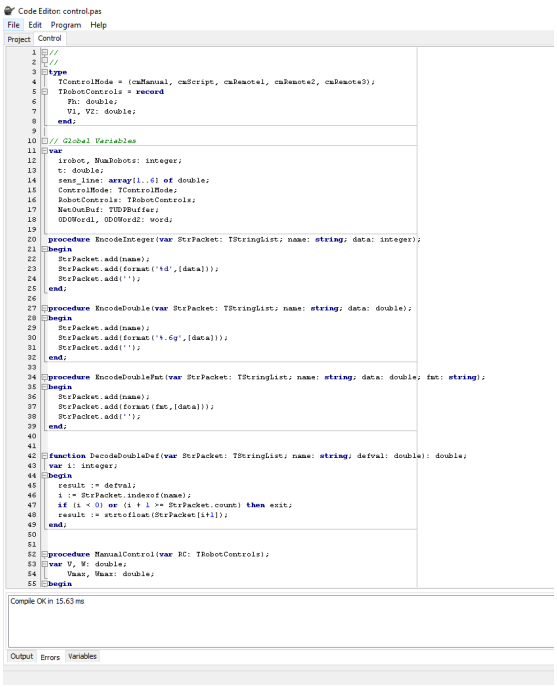


Figure 3.3: Control Editor.

3.2.4 Config

The configuration window allows changing several simulation parameters, such as:

- Position and behavior of the simulation visualization camera;
- Lighting position and attenuation;
- Simulation speed;
- Enable/disable shadows or fog;
- Wind speed (0 m/s by default);

The I/O tab allows configuration of serial, UDP or Modbus connection.

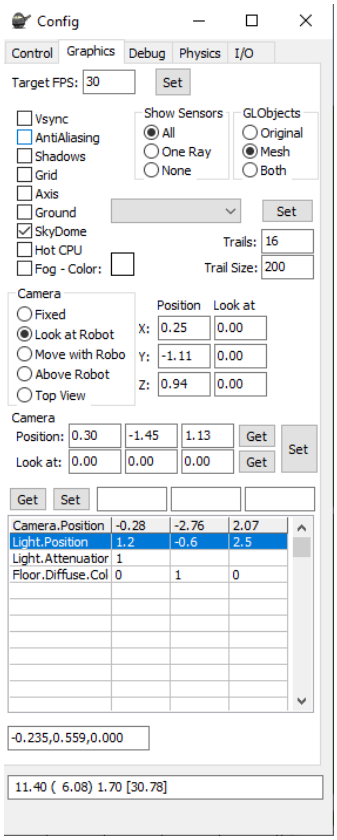


Figure 3.4: Config.

3.2.5 Chart

It allows monitoring the position and speed (in x, y and z) of the robots in relation to time. In addition, it is possible to save simulation logs.

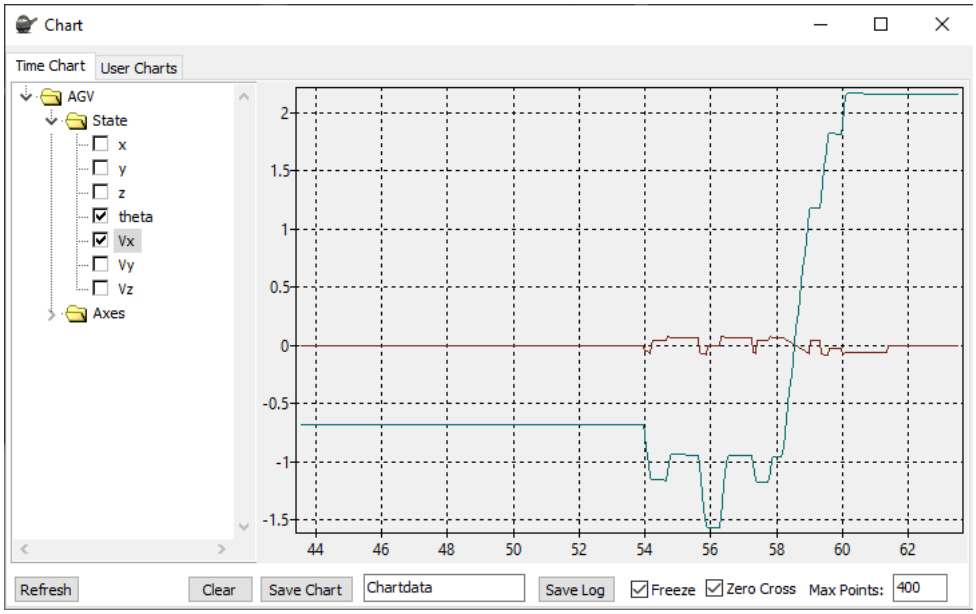


Figure 3.5: Chart.

3.2.6 Sheet

The sheet is a table that allows showing in real time the value of any desired variables. It is also possible to add buttons that trigger routines in the Control Editor.

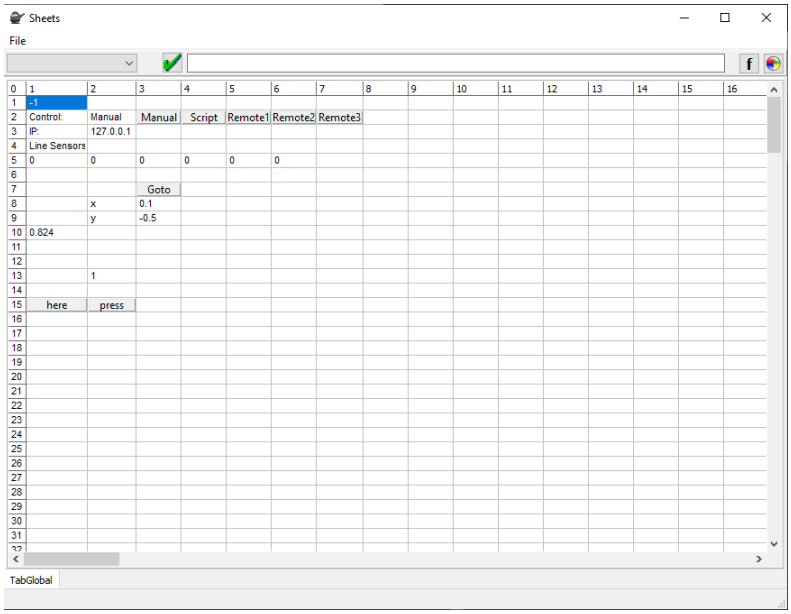


Figure 3.6: Sheet.

3.3 Construction of the Scene

This section presents the scene creation process. The physical description of each entity is made using XML files, with all files present in the GitHub project, whose link is in Appendix A.

Before starting the construction of the scene, it is important to keep in mind which entities will be present in the simulation and how they will interact, so that it is possible to define what the virtual environment will be like. The proposed ground-truth system will consist of a two-dimensional laser scanning subsystem and a camera subsystem, which will estimate in real time the location/orientation of a cylindrical target with colored markers to be placed on a mobile robot. It is intended that the laser scanning subsystem is able to detect the location of the target, sending its coordinates to the camera subsystem so that it tracks it, keeping it in the center of its field of view and estimating its orientation from the markers.

In order for the laser sensor and the camera to have a good coverage of the entire

scene, the layout shown in figure 3.7 was proposed. The LRF must have an intermediate height, higher than camera and lower than target, so that only the target is detected by laser sensor.

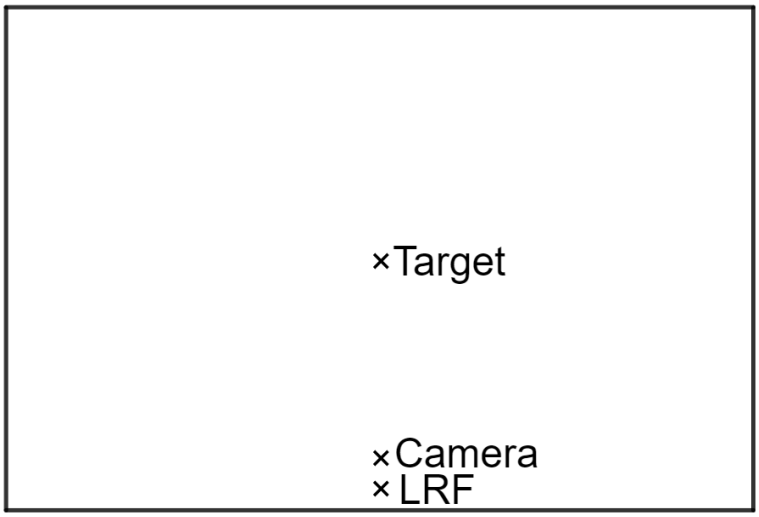


Figure 3.7: Proposed layout for the scene.

To start the construction of the above layout in the simulator, walls were used to create a 4x8m^2 rectangular space where all other entities will be contained.

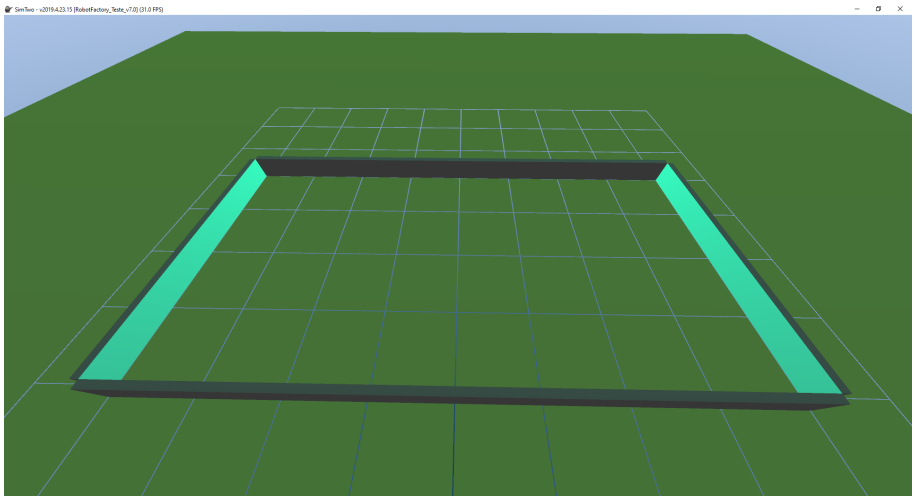


Figure 3.8: Walls.

The laser scanning subsystem consists of a fixed base and the LRF. In SimTwo, the

laser sensor already has the option to rotate around its own axis, making a two-dimensional sweep, simply by setting the following parameters:

- Beam length and thickness;
- Maximum rotation angle around the axis and number of samples to be obtained;
- Period;
- Noise addition.

The laser scanning subsystem will be positioned according to figure 3.9 (where the black rectangle represents the walls and the dashed red line segment represents the laser beam). In order for the beam to reach all points in the scene, its length must be at least $4\sqrt{2}$ m.

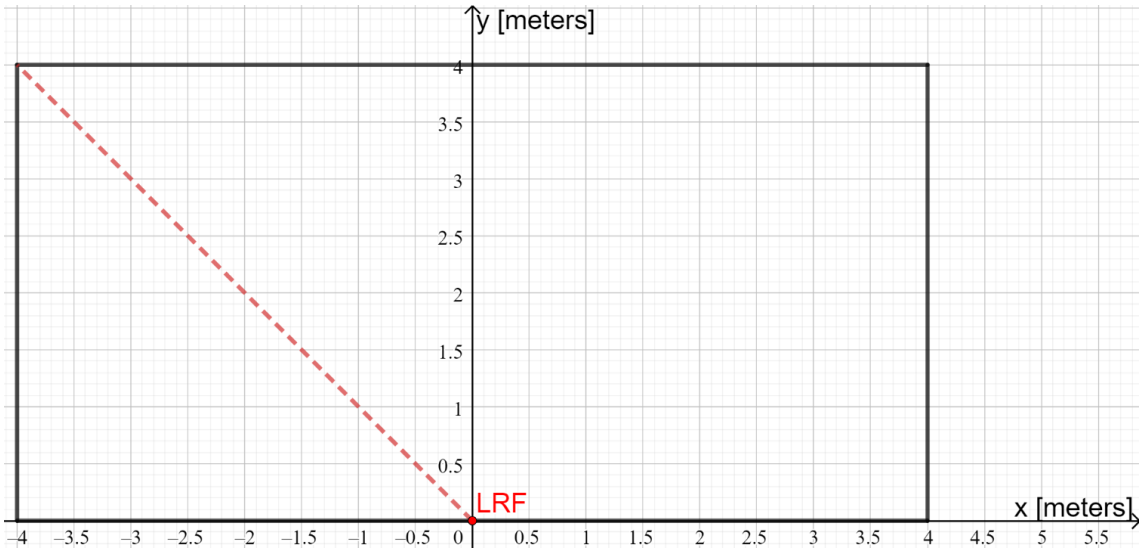


Figure 3.9: Position of the laser scanning subsystem in the scene.

For the sensor to behave closer to reality, its accuracy must be limited due to errors inherent to it. Taking as reference a LRF currently available on the market, the 2D LIDAR LMS101-10000 by Sick (mentioned in section 2.2), there is a systematic error ±30 mm in its measurements [49]. In addition to systematic errors, it is known that real

measures are also affected by random errors. Thus, to make the model's behavior in the SimTwo realistic, noise equivalent to a standard deviation of 30 mm was added to the sensor. The sensor was configured to have a scan time of 100 ms and a field of view of 180º, recording 360 samples during a complete scan. Figure 3.10 shows the laser scanner subsystem.

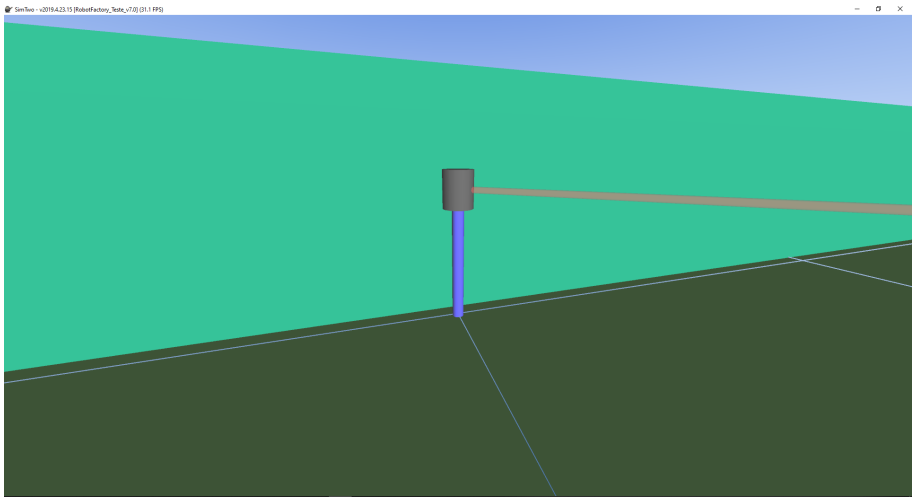


Figure 3.10: Laser scanner subsystem.

The camera subsystem (figure 3.11) consists of a fixed base, motor and an RGB camera. The motor allows the camera to rotate 180° around its own axis, being able to track the target detected through the laser sensor, regardless of its location on the scene. The camera subsystem has a lower height than the LRF, so that it cannot be detected by it. In addition, the subsystem was created so that the camera is at half the height of the robot. The SimTwo camera has a resolution of 320x240 pixels and has been configured for a frame rate of 30 FPS.

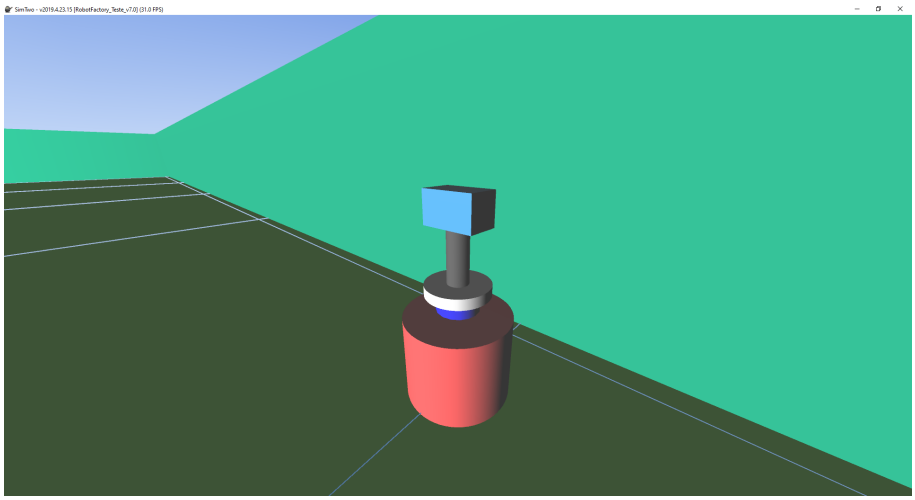


Figure 3.11: Camera subsystem.

Finally, a robot with differential traction is created, with a white cylindrical target of 0.20 m in diameter on it, which will have its position/orientation estimated through the ground-truth system. Measurements related to orientation will be made by means of colored passive markers present on the target and spaced 60° apart. The set made up of the robot and target has a height slightly higher than the LRF, so that it can be detected. Figure 3.12 shows the robot with the cylindrical target on it.

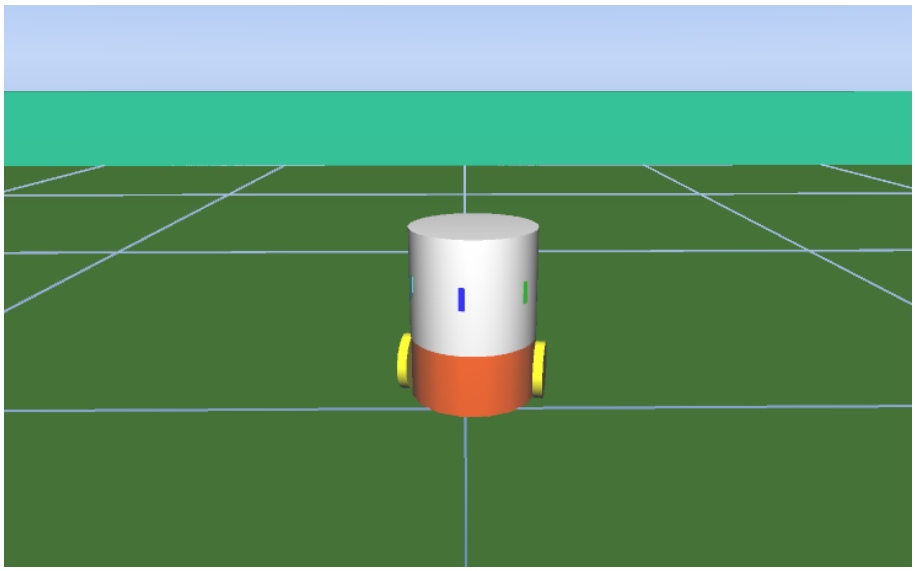


Figure 3.12: Robot (orange) with cylindrical target (white with colored markers).

Figure 3.13 shows all the entities created. Figure 3.14 shows an overview of the scene with all entities in their initial positions.

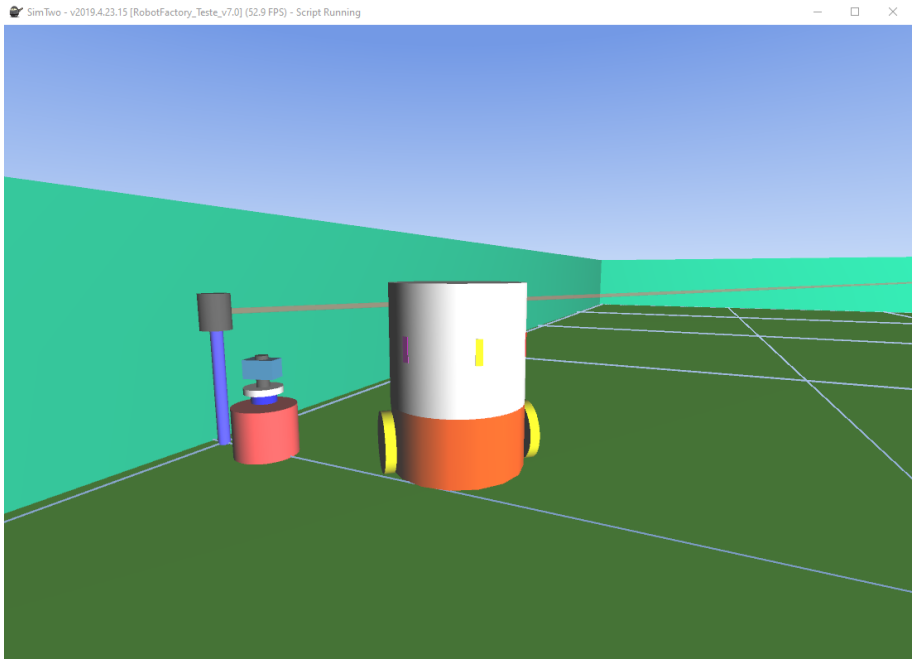


Figure 3.13: Ground-truth system and robot with target.

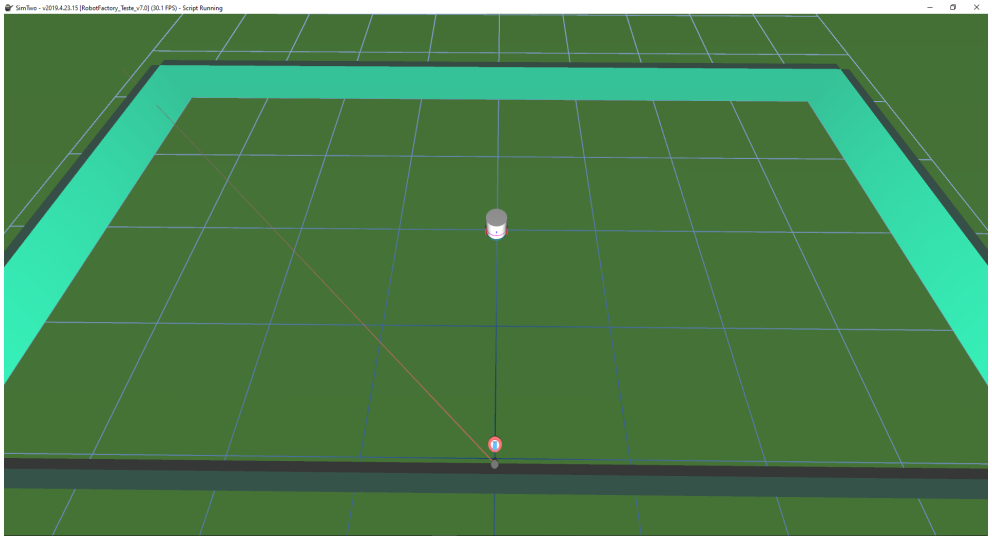


Figure 3.14: Top view of the scene.

The table below contains the coordinates for the initial positions of each entity in the

scene (except the walls). As in the ground-truth system all distances will be measured starting from the laser scanning subsystem, it is considered the origin of the coordinate system.

Table 3.1: Initial positions defined for each entity. All values are in meters.

Entity	**X**	**Y**
LRF	0.00	0.00
Camera	0.00	0.10
Target	0.00	2.00

Chapter 4

Ground-Truth System

This chapter explains the methodology developed so that the ground-truth system is able to estimate the location/orientation of the cylindrical target.

4.1 Localization

This section explains how the cylindrical target location is determined using the proposed ground-truth system. All the mentioned methodology is implemented in the file "control.pas", available in the project present in Appendix A.

The proposed ground-truth system consists of two subsystems: the laser scanning subsystem and the camera subsystem. The target location is estimated using the laser sensor. Positioned according to figure 3.14, the laser sensor has the ability to cover the entire scene in a two-dimensional form, rotating 180° around its own axis. The sensor was configured to obtain 360 samples in this interval, with the data being stored in arrays of 360 positions, where each position represents the polar coordinate of an obstacle found. To determine the coordinates for each position of the array, one must keep in mind that:

- Since 360 samples are obtained over a 180° interval, each index of the collected data array represents an increase of 0.5° in comparison to the previous index;
- Each array element represents the distance between the LRF and the obstacle

encountered, given the corresponding angle.

As explained in the previous section, the "Sheet" window has a table through which one can view variable values and interact with buttons (with functions previously defined in "Control" window). Three buttons were created on the Sheet, one with the function of starting to store the data obtained by the LRF in a text file, a second one with the function of interrupting data storage and a third with the function of saving that file. In this way, it is possible to create a record with all the samples obtained by the sensor and use an external software for data visualization.

Using the MATLAB software, the scene walls and the points obtained by the LRF in a complete scan were plotted. The result is shown in figure 4.1.

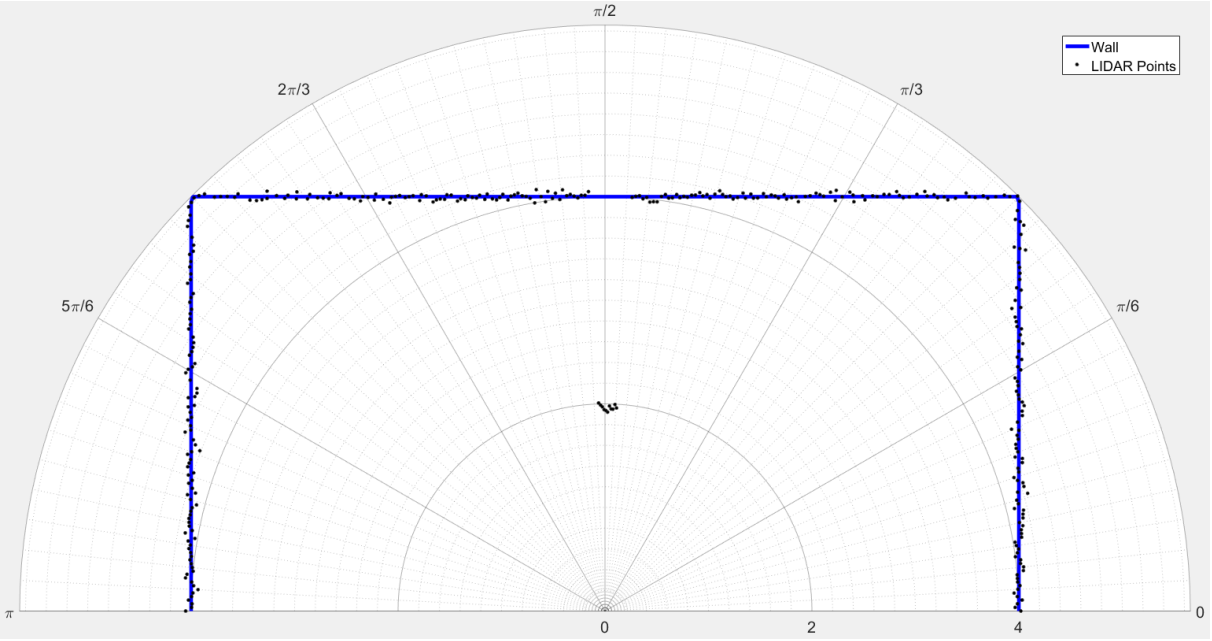


Figure 4.1: Points obtained.

In the figure above it is possible to distinguish the points referring to the target (in the center of the map) from the points referring to the walls (at the borders). However, only data related to the target are of interest, and the data relating to the wall should be discarded. Thus, it is necessary to process the data obtained by the sensor in order to

remove undesired points.

4.1.1 Creation of the Acceptance Zone

In order to remove the points related to the wall, an acceptance zone was created: points located within that zone will be maintained, but all points external to it will be discarded. The acceptance zone must present the same geometry as the scene walls, however, it must have smaller dimensions, since the samples obtained by the LRF are contaminated with noise, so that points referring to the wall may present larger or smaller distances than the real ones, as seen in figure 4.1.

Determining the dimensions of the acceptance zone is a compromise solution: on one hand, the zone must be small enough so that it does not include any point related to the wall that may have a shorter distance than the real one, on other hand, the acceptance zone should be as large as possible, so that no point relative to the target is discarded when it is close to the wall. Once the noise characteristics are known (as explained in the previous section) and the noise is equivalent to a standard deviation of 30 mm in the data, it can be interpreted that each reading of the LRF is contained within a Gaussian probability distribution curve.

Curves with Gaussian distribution have a property known as "empirical rule" [50], which states that in a normal distribution, approximately:

- 68,2% of data are contained within the interval between $\pm 1\sigma$;
- 95,4% of data are contained within the interval between $\pm 2\sigma$;
- 99,7% of data are contained within the interval between $\pm 3\sigma$;

Where σ is the standard deviation. This property is represented in the figure below.

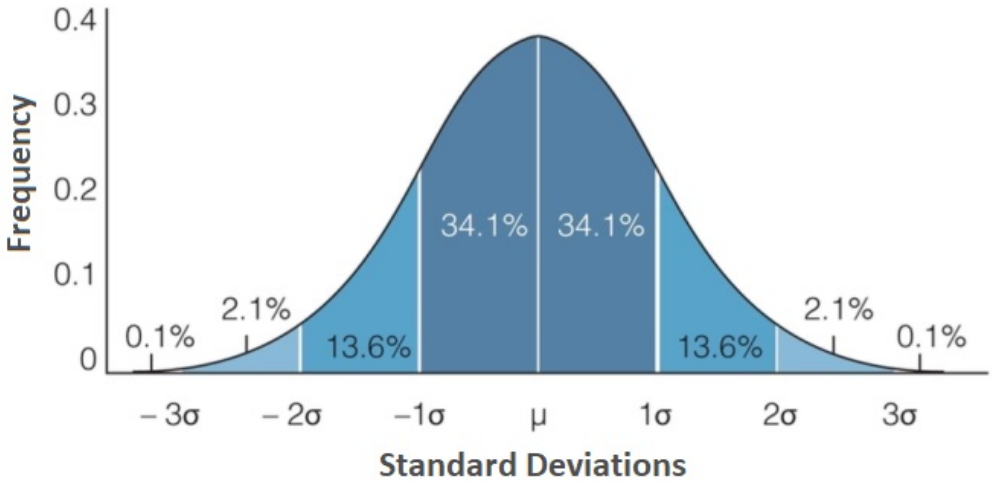


Figure 4.2: Empirical Rule. Adapted from [50].

In order for the acceptance area to be as large as possible and at the same time be able to eliminate undesired points, the mentioned property can be used. Considering that the acceptance zone is internal to the wall, distant from it by 90 mm (three standard deviations below the real value), it is known that approximately 99.8% of the data related to the wall will be discarded, that is, only 0.2% of undesirable data will be located within the acceptance zone. In addition, even if the target is close to the wall, it will be possible to detect it, as its diameter is 0.20 m, thus its points will be located internally to the acceptance zone. Since the noise characteristics are known and have a behavior similar to a Gaussian distribution, this methodology can be applied to remove undesired points relating to any objects present in the scene.

The implementation of the acceptance zone with the aforementioned characteristics will be executed through the creation of a array, defined as:

$$allowed\,[\theta] = \begin{cases} \frac{4-0.09}{cos(\theta)}, & \text{if } 0 \leq \theta < \pi/4 \\ \frac{4-0.09}{sin(\theta)}, & \text{if } \pi/4 \leq \theta < 3\pi/4 \\ -\frac{4-0.09}{cos(\theta)}, & \text{if } 3\pi/4 \leq \theta < \pi \end{cases} \tag{4.1}$$

Figure 4.3 shows a map of the scene with the walls, the acceptance zone and the points obtained by the LRF in a complete scan. It is noted that almost all points referring to the wall are outside the acceptance zone.

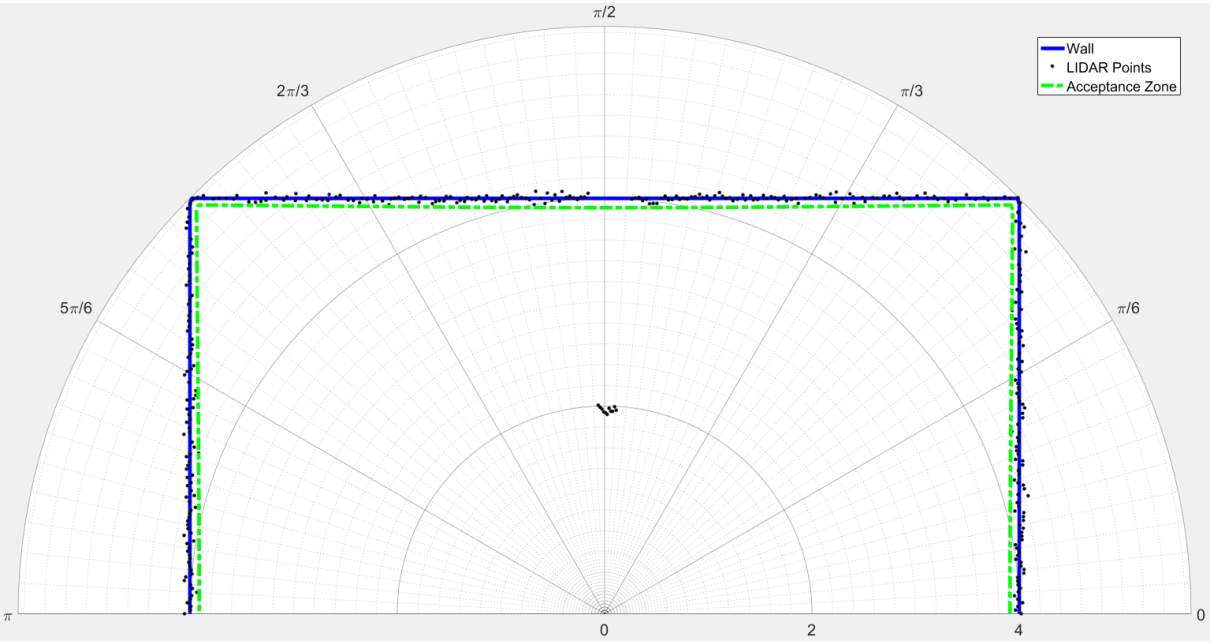


Figure 4.3: Acceptance zone and points obtained.

Figure 4.4 shows the same elements of the previous image, but with data referring to a new scan of the LRF made with the target placed as close as possible to the wall. As it's possible to observe, the points belonging to the target are easily distinguishable from the remaining undesired points.

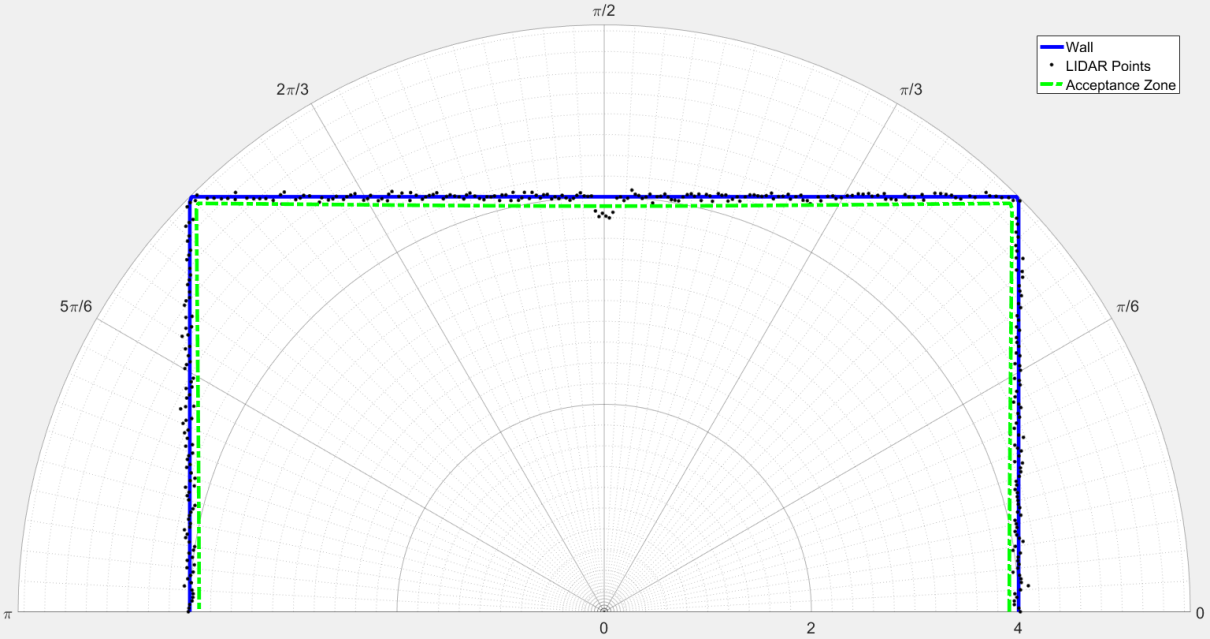


Figure 4.4: Acceptance zone and points obtained with the target next to the wall.

4.1.2 Determining the Location of the Target

Once the acceptance zone is created, it is necessary to use it in order to eliminate the points related to the wall. Therefore, a comparison between the data obtained by the LRF and the acceptance zone must be performed. For each angle, it is verified if the distance obtained by the LRF is equal to or greater than the values given by equation 4.1: if true, its value must be equaled to zero; if the distance is shorter, the point is within the acceptance zone and must be maintained. Thus, when analyzing the resulting array, all points with a distance different than zero are potential candidates to be related to the target, but as previously stated, 0.2% of the points referring to the wall may occasionally be within the acceptance zone. Figure 4.5 shows the data of the resulting array after processing.

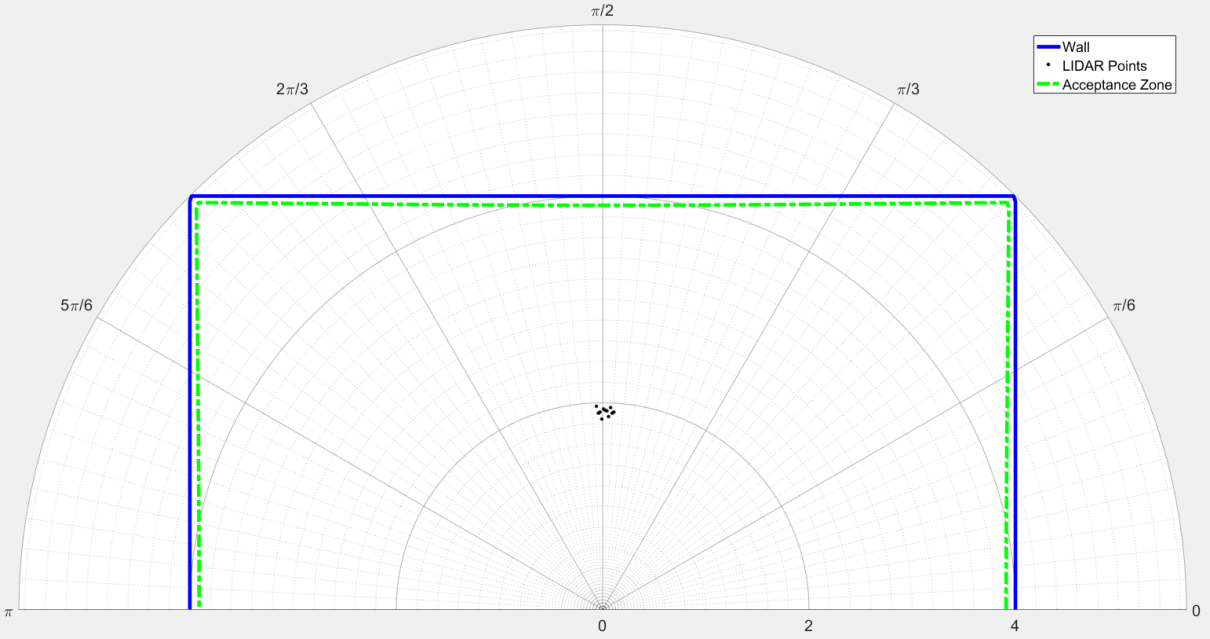


Figure 4.5: Resulting points after processing.

To determine the points relative to the target, all positions of the resulting array are traversed and the following method is applied:

- The first transition between zero and any value other than zero is sought. This transition can represent the target's edge;
- If immediately after the first transition another transition to zero occurs, it means that the previous point was an outlier, meaning any remaining undesired point. However, if after the first transition other element different than zero are found, it is safe to assert that the previous transition actually represented the target's border, so the index of the array where this transition occurred is saved as the initial border ("start");
- If the starting position has already been defined and another transition to zero is found, it is safe to assert that that transition represents the other edge of the target, so the index of the array where this transition occurred is saved as the final edge ("end").

To facilitate the explanation of the problem, figure 4.6 was created, which represents the LRF and the target. Once the array indexes corresponding to the initial and final border of the target have been found, it can be asserted that the central index between the edges represents point B, expressed through the distance AB and angle α. Therefore, considering a cartesian plane where the LRF is at the origin, the distance between it and the center of the target is given by the distance AC, which can be expressed through the same angle α and the distance AB plus the radius of the target. Therefore, the coordinates of the target will be given by:

$$\text{Target}[x, y] = [(AB + BC) \times \cos(\alpha), (AB + BC) \times \sin(\alpha)] \tag{4.2}$$

Similarly, on SimTwo:

$$\text{T}[x, y] = \left[(V(k) + r) \times \cos\left(\frac{k}{2}\right), (V(k) + r) \times \sin\left(\frac{k}{2}\right)\right] \tag{4.3}$$

Where V is the LRF data array, k is the central index and r is the radius of the target. It is important to note that, as mentioned earlier, the array has 360 positions that represent a range from 0° to 180°, so k is divided by 2.

Finally, a five-point moving average filter was implemented to smooth the results, in order to reduce the effects of noise.

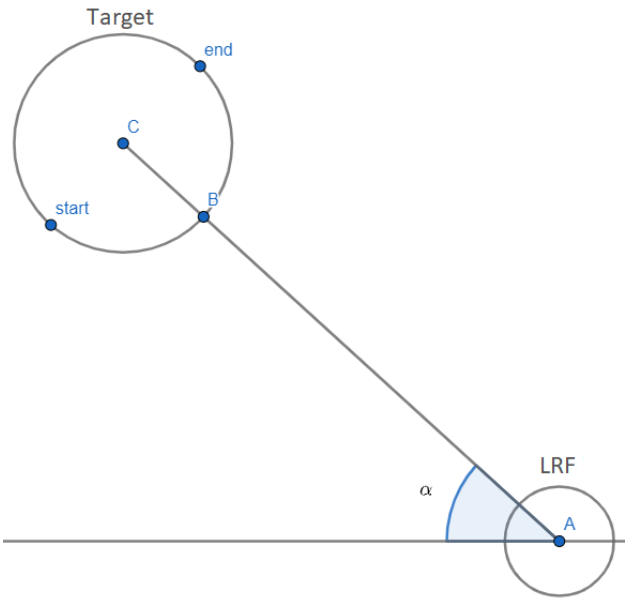


Figure 4.6: LRF and the target.

4.2 Orientation

In this section, it is explained how the robot's orientation is determined using the proposed ground-truth system. All the mentioned methodology is implemented in the file "control.pas", present in the project available in Appendix A.

The proposed ground-truth system consists of two subsystems: the laser scanning subsystem and the camera subsystem, with the target orientation being estimated by the latter. Positioned according to figure 3.14, the subsystem has an motor that allows the camera to rotate around its own axis, being able to track any object in the scene.

4.2.1 Target Tracking

As the location of the target is estimated in real time by the laser scanning sensor, the coordinates obtained are used to determine the angle of the motor to which the camera is able to center the target in its field of view. Figure 4.7 exemplifies the situation: it shows the LRF, the camera and the target. For the camera to track the target, the motor must

reach a β angle, given by:

$$\beta = \arctan\left(\frac{\Delta y - d}{\Delta x}\right) \tag{4.4}$$

Where Δx and Δy are the coordinates x and y of the target estimated by LRF, respectively, and d is the distance between LRF and camera (0.10 m).

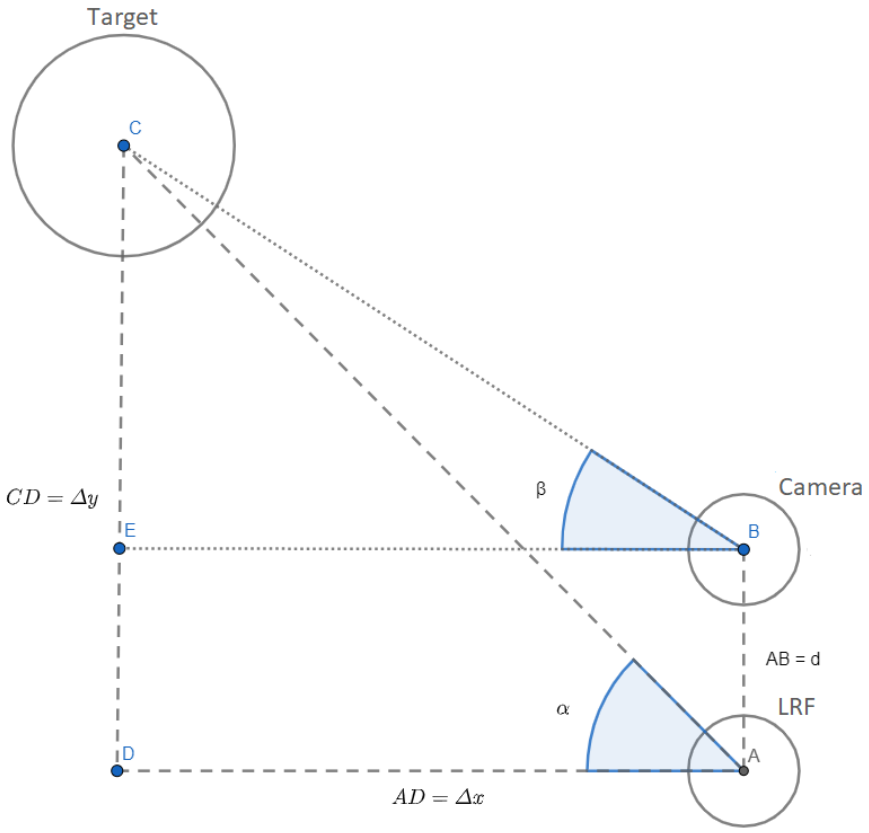


Figure 4.7: LRF, camera and target.

The angle β changes as the target moves, however, in a situation where the step of the motor that controls the camera is greater than a small variation of the angle β, undesired oscillations in the camera may occur, which can harm image processing. To avoid undesired oscillations, a dead zone has been implemented, so it is tolerable that there is an absolute difference of one degree between the current angle of motor and angle β. In this way, the dead zone is able to prevent oscillations without compromising the

tracking of the target, which means that, even if the camera does not move, the target remains in its field of view.

4.2.2 Determination of Target Orientation

Once the target is located in the camera's field of view, it is necessary to process the image to identify the target and determine its orientation. The determination is made through the colored markers placed on the target spaced 60° apart. The elaborated methodology can be applied to determine the location/orientation of robots with different geometries, as long as its possible to attach the cylindrical target with the colored markers to them.

The camera available in the simulator uses the RGB color space and has a resolution of 240x320 pixels. As explained in the previous section, the camera was positioned to be half the height of the target. To ensure that the images have good lighting, the scene's light source was placed close to the camera, in order to minimize undesired shadows that may interfere negatively in the image processing. Figure 4.8 shows a record of the target (and consequently, the robot) made by the camera.

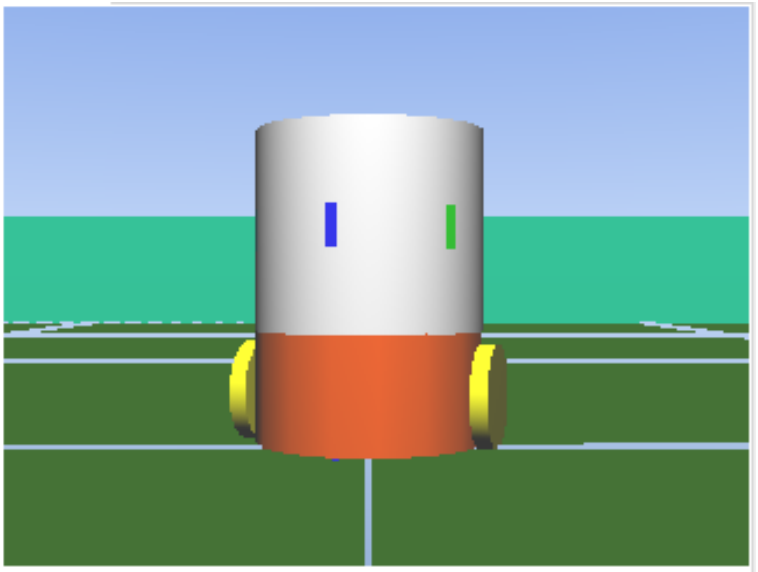


Figure 4.8: Record of the camera containing the target and the robot.

When creating the simulation, the white color was used exclusively on the target, with the intent that no other element of the scene would have this color, then, during image processing it would be possible to identify the target looking for any color transitions to white colors. However, as the current version of SimTwo is limited to having only one point of light in the scene, the edges of the target are not fully illuminated, in order to present shadows and appear to have shades of gray, which is noticeable in the figure above.

To identify the pixel relative to the upper edge of the target, it is necessary to analyse from top to bottom the central column of the image matrix, until the first pixel that simultaneously satisfies the following conditions can be found:

- All color components (R, G and B) must have the same intensity;
- The components must have an intensity greater than or equal to a reference constant.

The reference constant is related to the gray levels to be tolerated as part of the target. In this way, an attempt is made to minimize the effect of shadows detecting their edges. The constant must be larger than zero (otherwise, any pixels would be allowed) and less than 255 (in this case, only fully white pixels would be tolerable). The constant must be determined empirically, testing different values until it is realized that all the edges of the target are being detected (which can be assessed using the ratio between the diameter and the height of the target).

After finding the first pixel relating to the upper end of the target, the same method must be applied to the lower edge, but reversing the direction in which the column is traversed. To determine the horizontal edges, the line corresponding to a quarter of the distance between the vertical ends must be determined and the same method should be used, just changing the direction in which the line is followed. Knowing the pixels referring to the vertical and horizontal edges, it is possible to determine the radius of the target in the image and also its height.

Figure 4.9 exemplifies the data found so far. In it, it is possible to observe the pixels related to the edges (highlighted in black) and two imaginary lines. The pixels referring to the horizontal borders will always be located on the pink line (which represents a quarter of the distance between the vertical ends). The yellow line represents half the height of the target. The target was built in such a way that the colored markers are centered at half its height, so they will always be on the imaginary yellow line.

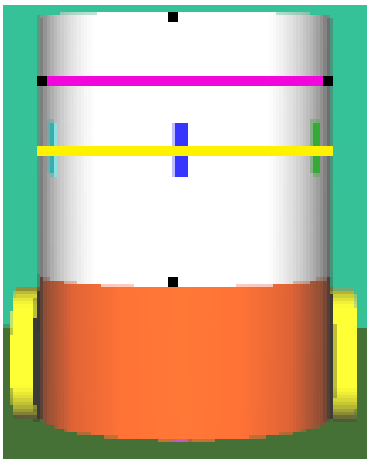


Figure 4.9: Imaginary lines and pixels relative to the target's edges.

Once the target radius has been determined, it is possible to estimate its orientation by analyzing only the pixels referring to the yellow imaginary line. For this, it is necessary to take a pixel as a fixed point and measure the distance between it and the center of the next colored marker (for this reason six colored markers were placed around its body, being spaced 60° from each other, so that regardless of the target's orientation, there is always a colored marker visible to the camera). Figure 4.10 illustrates the situation:

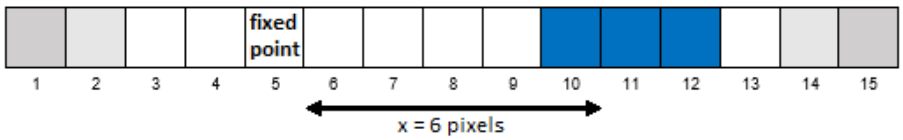


Figure 4.10: Pixels referring to the yellow imaginary line.

Figure 4.11 shows a representation of the top view of the target. All points shown are

coplanar, being that:

- The AB segment represents the distance between the target center and the chosen fixed point;
- The AC segment represents the distance between the center of the target and the center of the colored marker;
- The BC segment represents the distance between the chosen fixed point and the center of the colored marker;
- The angle θ represents the angle of the target in relation to the chosen fixed point.

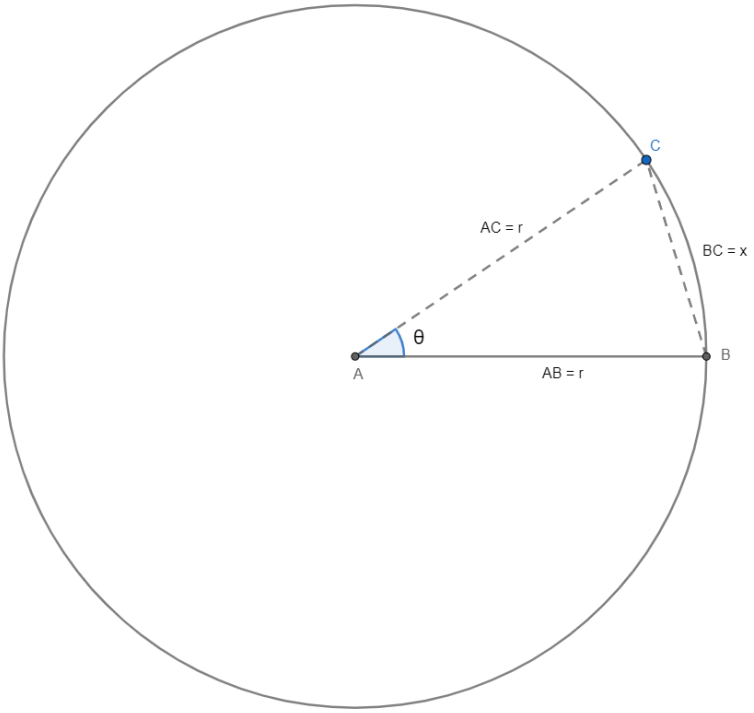


Figure 4.11: Top view representation of the target.

The points A, B and C form an isosceles triangle, with the length of the segments and the angle θ being related to each other through the cosine law:

$$BC^2 = AB^2 + AC^2 - 2 \times AB \times AC \times \cos(\theta) \tag{4.5}$$

However, as $AB = AC = r$ (radius of the target in the image) and $BC = x$ (distance between the chosen fixed point and the center of the colored marker), equation 4.5 can be rewritten as:

$$x^2 = 2r^2 - 2r^2 \cos(\theta) \tag{4.6}$$

Therefore:

$$\theta = \arccos\left(\frac{2r^2 - x^2}{2r^2}\right) \tag{4.7}$$

Using the Desmos [51] graphing calculator, figure 4.12 was created, which illustrates the graph of equation 4.7 (in blue) for a random radius value ($r = 100$ pixels, in this case), represented by the green circumference.

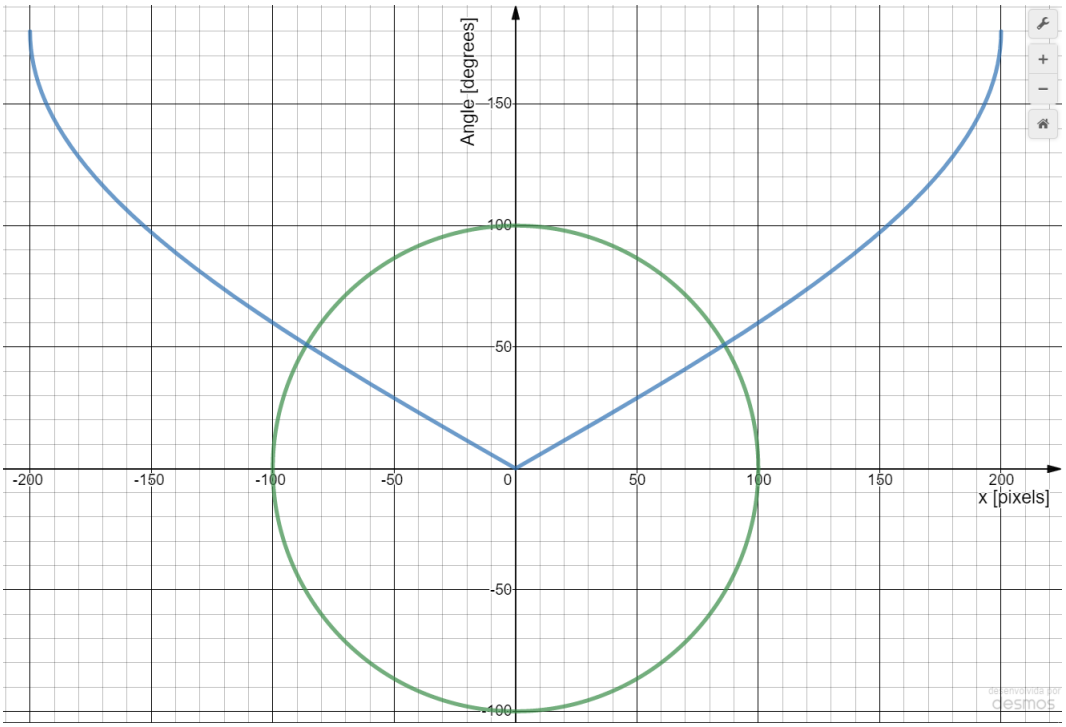


Figure 4.12: Graph of equation 4.7 and the target radius.

As previously mentioned, the analysis of the target's orientation is made considering only the pixels referring to the yellow imaginary line. In addition, there must always be a pixel to be considered as a fixed point and a colored marker visible to the camera. As the colored markers are spaced 60° apart, the pixel located at a distance x_p to the left of the target center was chosen as a fixed point, so that the distance x_p corresponds to an angle of 30° between the fixed point and the center of the target. By manipulating equation 4.7 algebraically, it is possible to calculate the value of x_p:

$$x_{\mathrm{p}} = r\sqrt{2\left(1 - \cos(\theta)\right)} \tag{4.8}$$

Since $\cos(30) = \sqrt{3}/2$,

$$x_{\mathrm{p}} = r\sqrt{2\left(1 - \frac{\sqrt{3}}{2}\right)} \approx 0.518r \tag{4.9}$$

Therefore, the pixel that is $0.518r$ pixels to the left of the target's center is considered the fixed point and the distance x to the center of the next colored marker located to its right is calculated from it. Since the markers are spaced 60° apart, it is expected that the center of the next colored marker will be located less than r pixels away (x_r) from the fixed point, because according to equation 4.7 when $x = r$, $\theta = 60^{\circ}$. Once the x distance has been determined, equation 4.7 is used to calculate the corresponding angle. The fixed point was chosen as mentioned above so that the calculation of the distance x is made with the pixels referring to the central region of the target, avoiding pixels close to the edges, which are strongly influenced by the shadows and could introduce errors in the measurements. Figure 4.13 shows the graph of equation 4.7 (in blue), the target circumference (in green), the points x_p and x_r (in black) with the interval between them (in orange).

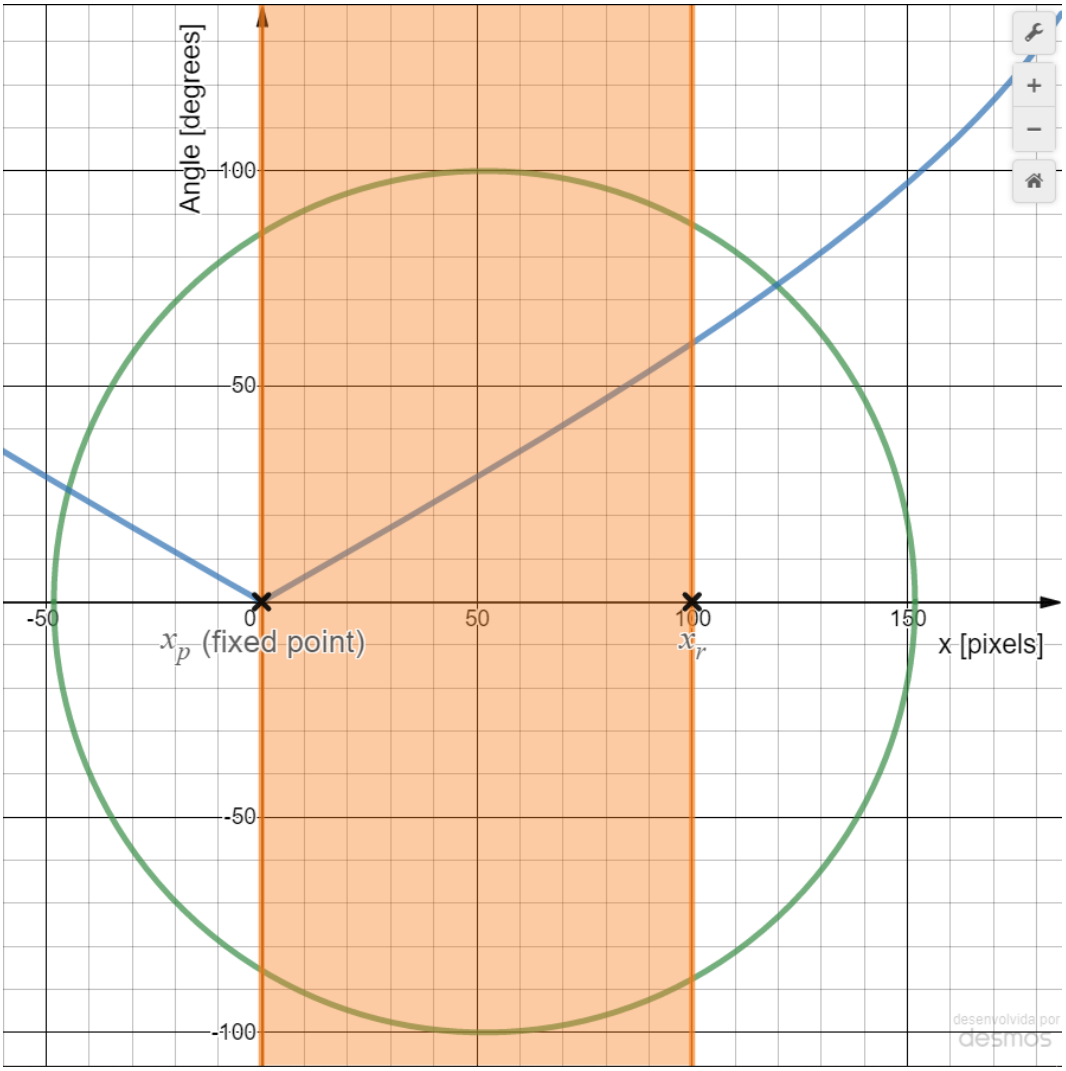


Figure 4.13: Graph of equation 4.7, the target radius and the points x_p and x_r with the interval between them.

A constant must be added to the result obtained, depending on the color of the colored markers found:

- Blue: -30°;
- Green: -90°;
- Pink: -150°;
- Yellow: -210°;

- Red: -270°;
- Light blue: -330°.

Finally, to determine the orientation of the target (and consequently, the robot), the angle β (mentioned in the previous subsection) is added to the obtained angle.

Chapter 5

Results

This chapter presents the results obtained in this thesis. To carry out the validation of the proposed ground-truth system, SimTwo was configured in order to record in a text file the estimated and real values related to the location and orientation of the target (and consequently, the robot) during three simulations. Using the MATLAB software it was possible to analyze the behavior of the data obtained.

5.1 Static Target

This section contains the results obtained in the tests carried out with the static target. For this, seven tests were performed in different positions (A, B, C, D, E, F, G), as shown in figure 5.1. Table 5.1 shows the coordinates and the orientation with which the target was positioned at each point.

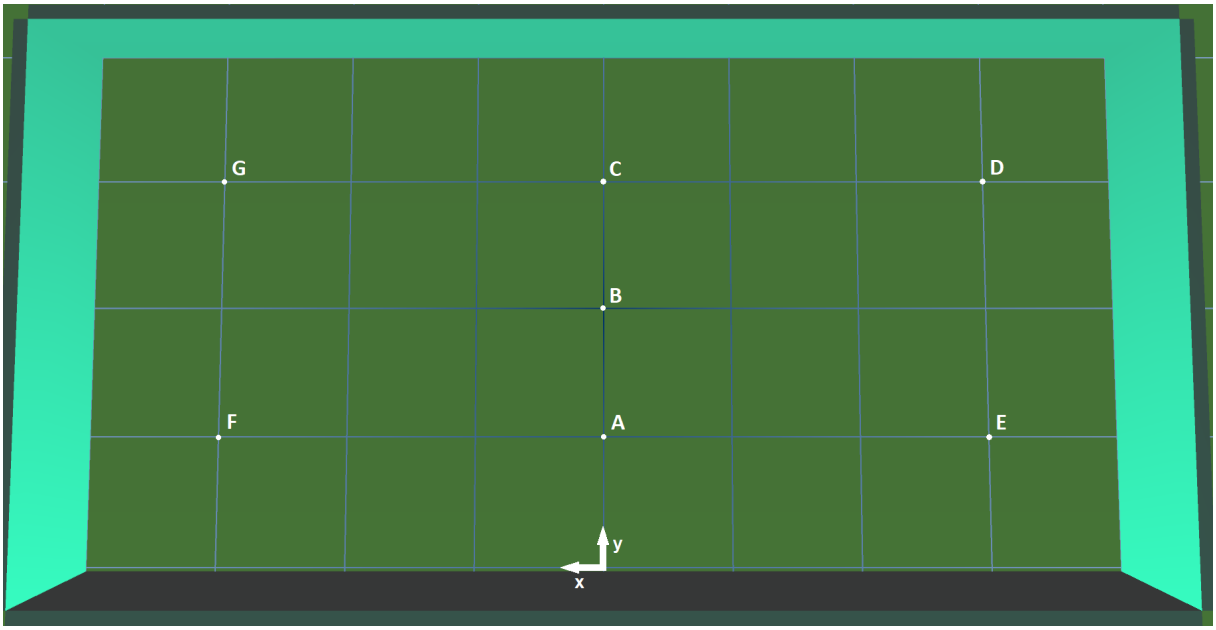


Figure 5.1: Test points.

Table 5.1: Target position and orientation (x and y coordinates in meters and angle in degrees).

Position	**X**	**Y**	**Angle**
A	0.00	1.00	0.00
B	0.00	2.00	0.00
C	0.00	3.00	0.00
D	-3.00	3.00	0.00
E	-3.00	1.00	0.00
F	3.00	1.00	0.00
G	3.00	3.00	0.00

For each position, 1000 samples of each data estimated by the ground-truth system were collected. Figures 5.2 to 5.8 contain the frequency distributions (histograms) of the data.

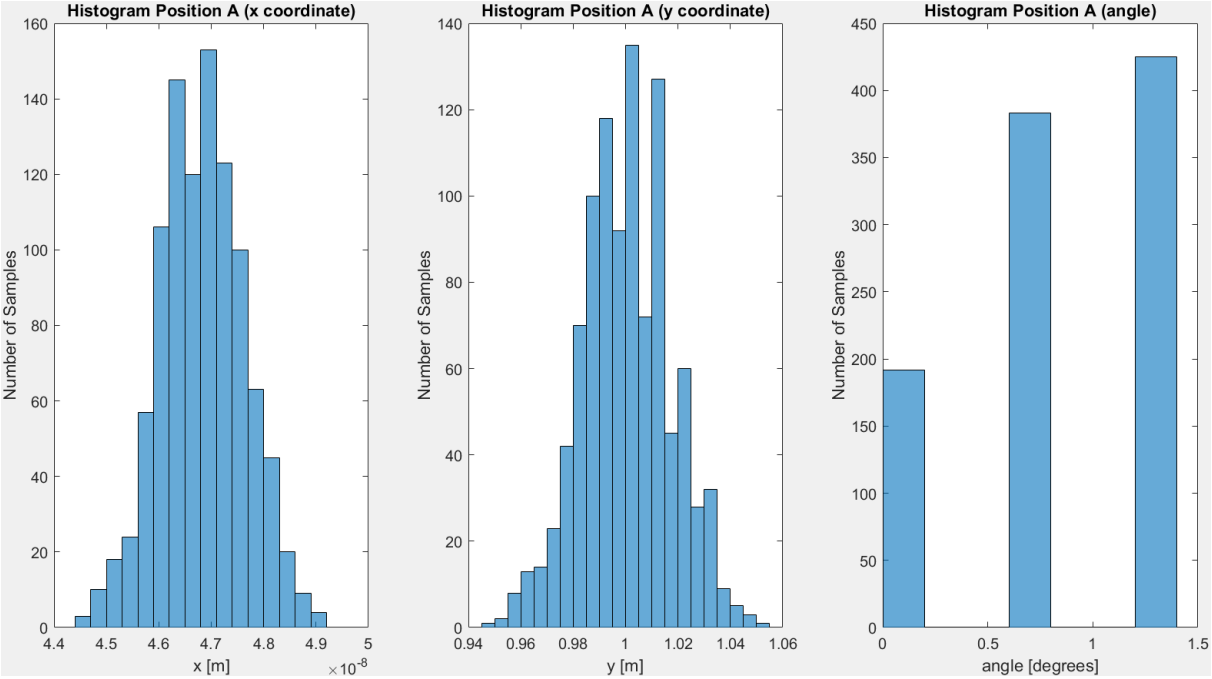


Figure 5.2: Histogram - Position A (0,1).

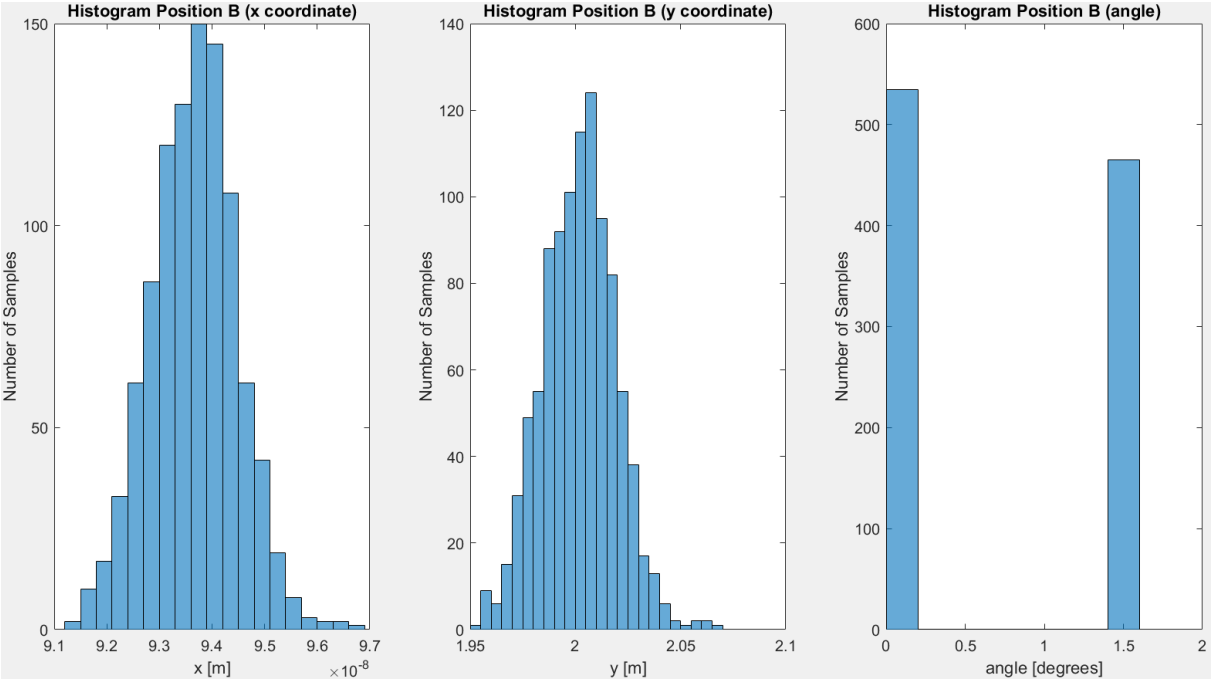


Figure 5.3: Histogram - Position B (0,2).

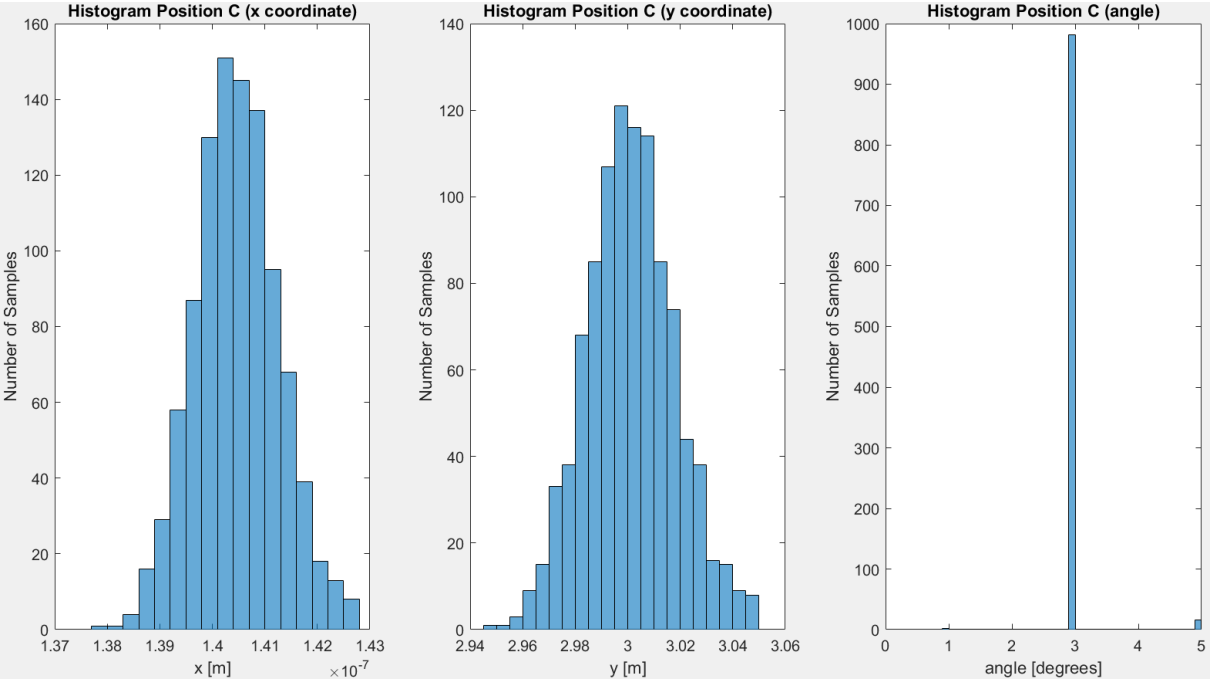


Figure 5.4: Histogram - Position C (0,3).

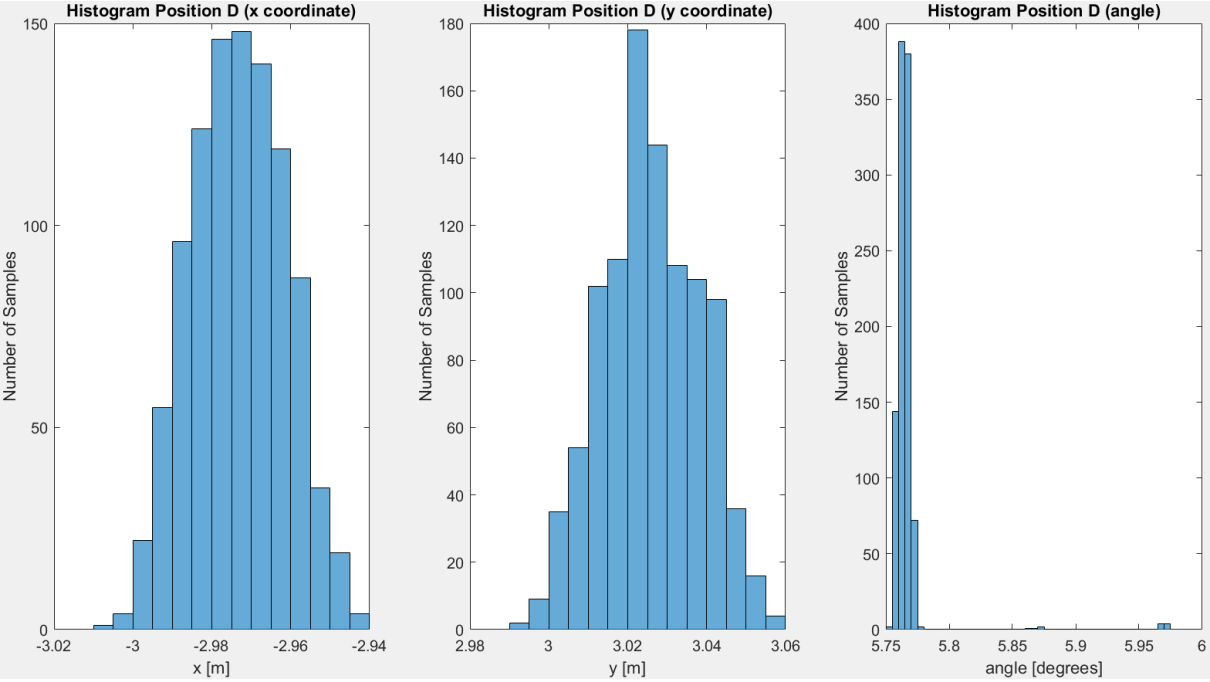


Figure 5.5: Histogram - Position D (-3,3).

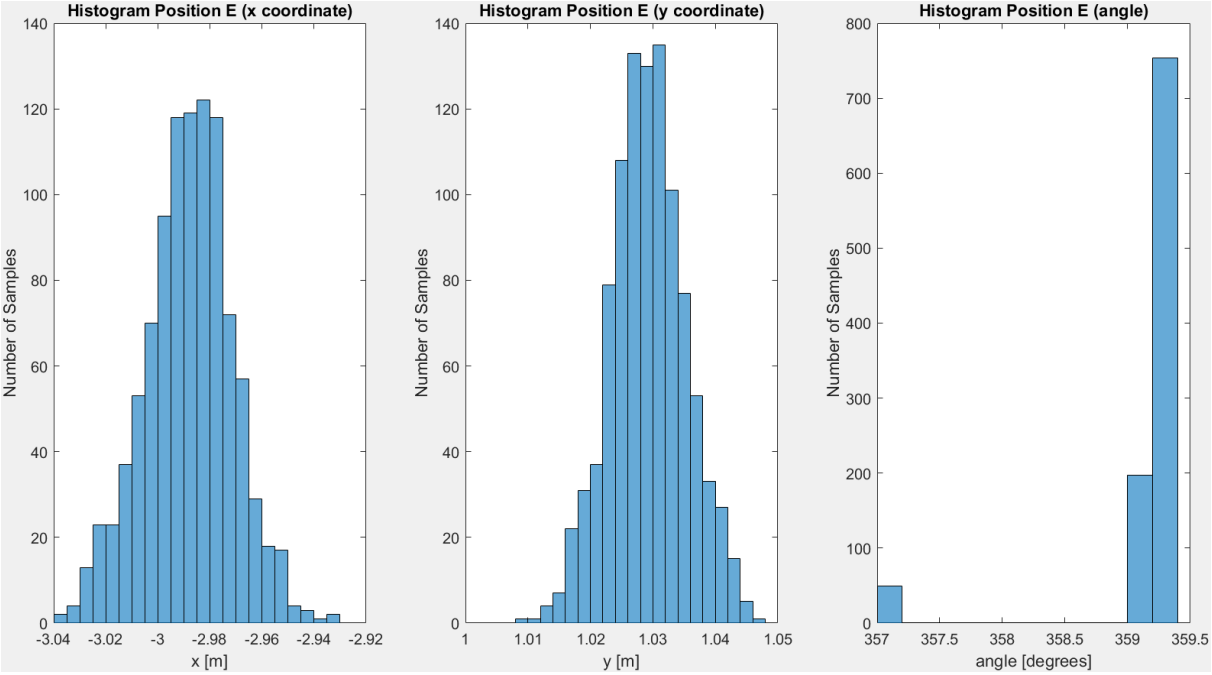


Figure 5.6: Histogram - Position E (-3,1).

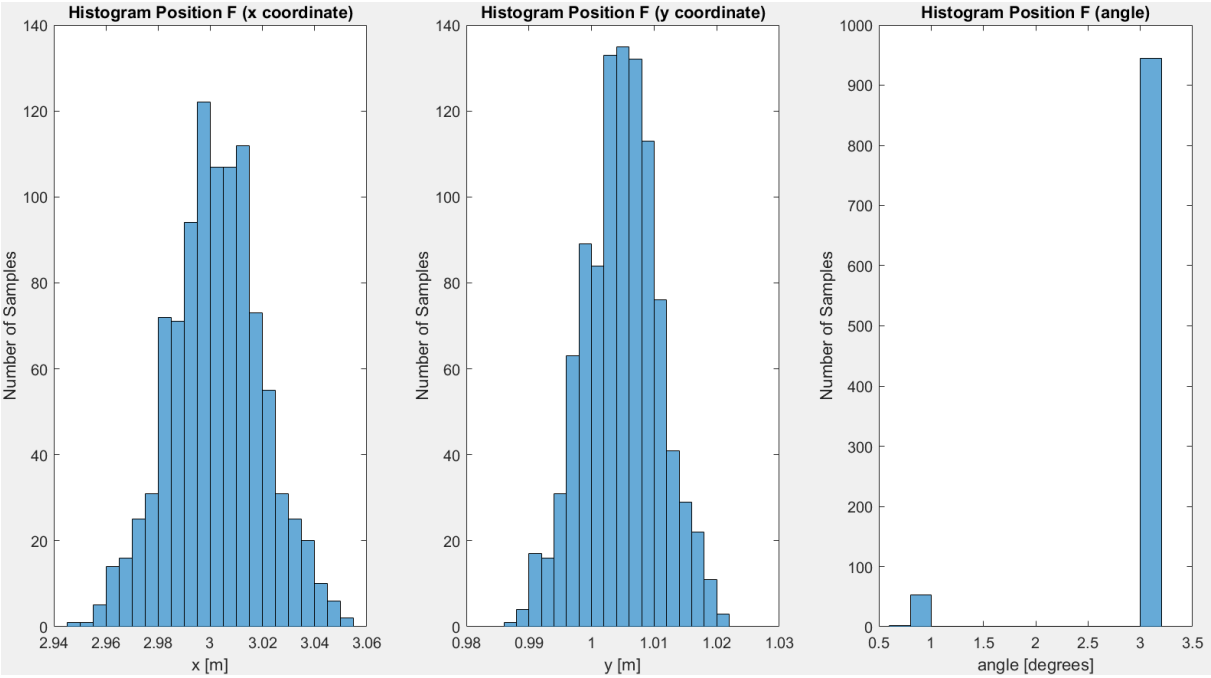


Figure 5.7: Histogram - Position F (3,1).

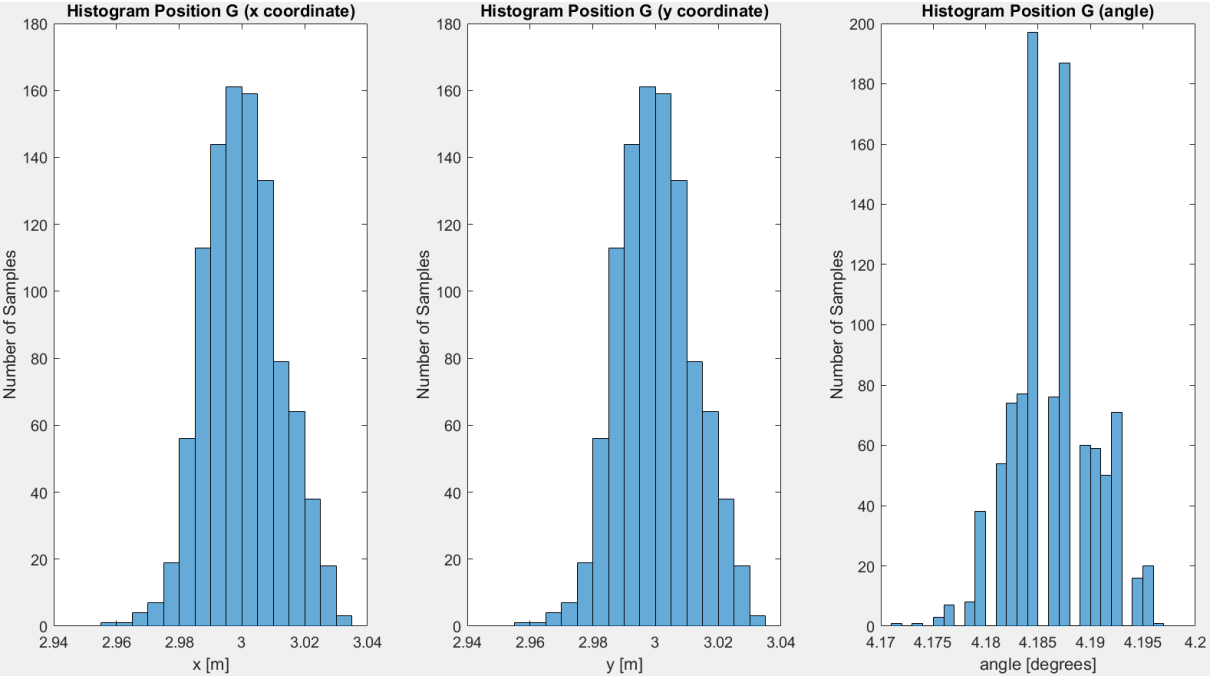


Figure 5.8: Histogram - Position G (3,3).

It is possible to notice that most of the data referring to the x and y coordinates have Gaussian distribution. The gaps observed in the angle data are explained by the low resolution of the SimTwo camera. A low resolution camera produces an image with inaccurate representation of the objects in the scene, which can result in inconsistencies in the measurement of the target radius and also in the distance between the fixed point and the center of the colored marker, introducing errors in the angle estimate.

Table 5.2 contains the mean, standard deviation and mode of the data obtained for each position.

Table 5.2: Mean, standard deviation and mode of measured data. All values are in meters for x and y coordinates and in degrees for the angle.

Pos	**Mean**			**Standard Deviation**			**Mode**		
	X	**Y**	**Angle**	**X**	**Y**	**Angle**	**X**	**Y**	**Angle**
A	0.0000	1.0006	0.8728	0.0000	0.0171	0.4462	0.0000	1.0010	1.3290
B	0.0000	2.0013	0.8157	0.0000	0.0173	0.6719	0.0000	2.0090	0.1896
C	0.0000	3.0006	2.9460	0.0000	0.0169	0.2664	0.0000	2.9950	2.9180
D	-2.9738	3.0259	5.7660	0.0120	0.0123	0.0198	-2.9760	3.0240	5.7650
E	-2.9886	1.0288	359.0814	0.0168	0.0060	0.4568	-2.9840	1.0270	359.2000
F	3.0018	1.0044	2.9095	0.0175	0.0058	0.5051	3.0020	1.0040	3.0280
G	2.9998	2.9998	4.1862	0.0121	0.0121	0.0040	3.0020	3.0020	4.1850

Table 5.3 contains the average absolute error of the data obtained.

Table 5.3: Mean absolute error of measured data. Values in meters for x and y coordinates and in degrees for the angle.

Pos	**Mean absolute error**		
	X	**Y**	**Angle**
A	0.0000	0.0137	0.8720
B	0.0000	0.0138	0.8125
C	0.0000	0.0134	2.9458
D	0.0262	0.0260	5.7660
E	0.0167	0.0289	0.9186
F	0.0139	0.0060	2.9000
G	0.0098	0.0098	4.1862

It is possible to notice that the developed system presents mean absolute error of 9 mm on the X axis, 16 mm on the Y axis and 2.63 degrees in the angle estimate for a static target.

5.2 Moving Target

This section contains the results obtained in the tests carried out with the moving target. For this purpose, three simulations were carried out, the first aiming to analyze the location estimate and the others aiming to analyze the target orientation estimate by the proposed ground-truth system.

5.2.1 Localization

To assess the performance of the location estimate, in the first simulation the robot was manually controlled in order to perform the path shown in figure 5.9, where the blue line represents the real coordinates obtained during the route, while the red line represents the estimated coordinates. This color pattern will be used for all other figures.

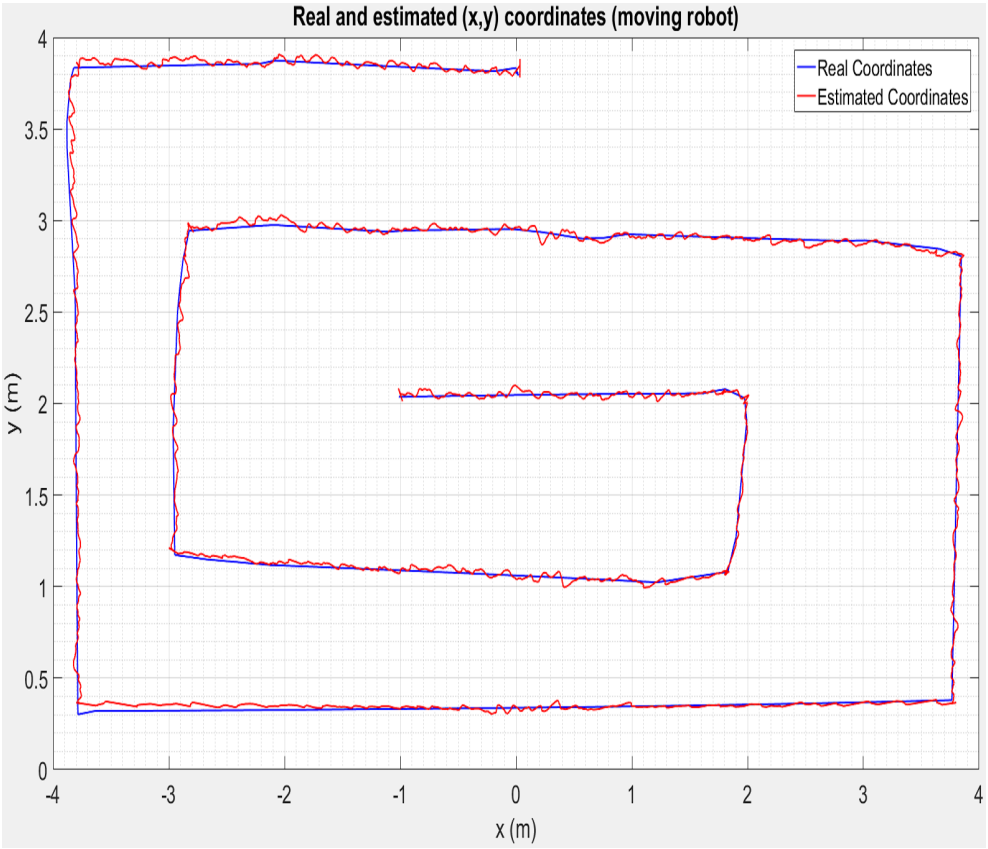


Figure 5.9: Real and estimated (x,y) coordinates data (moving robot).

Figure 5.10 shows the data referring only to the x coordinate, considering the same path shown above. In addition, the absolute error between actual and estimated values is shown.

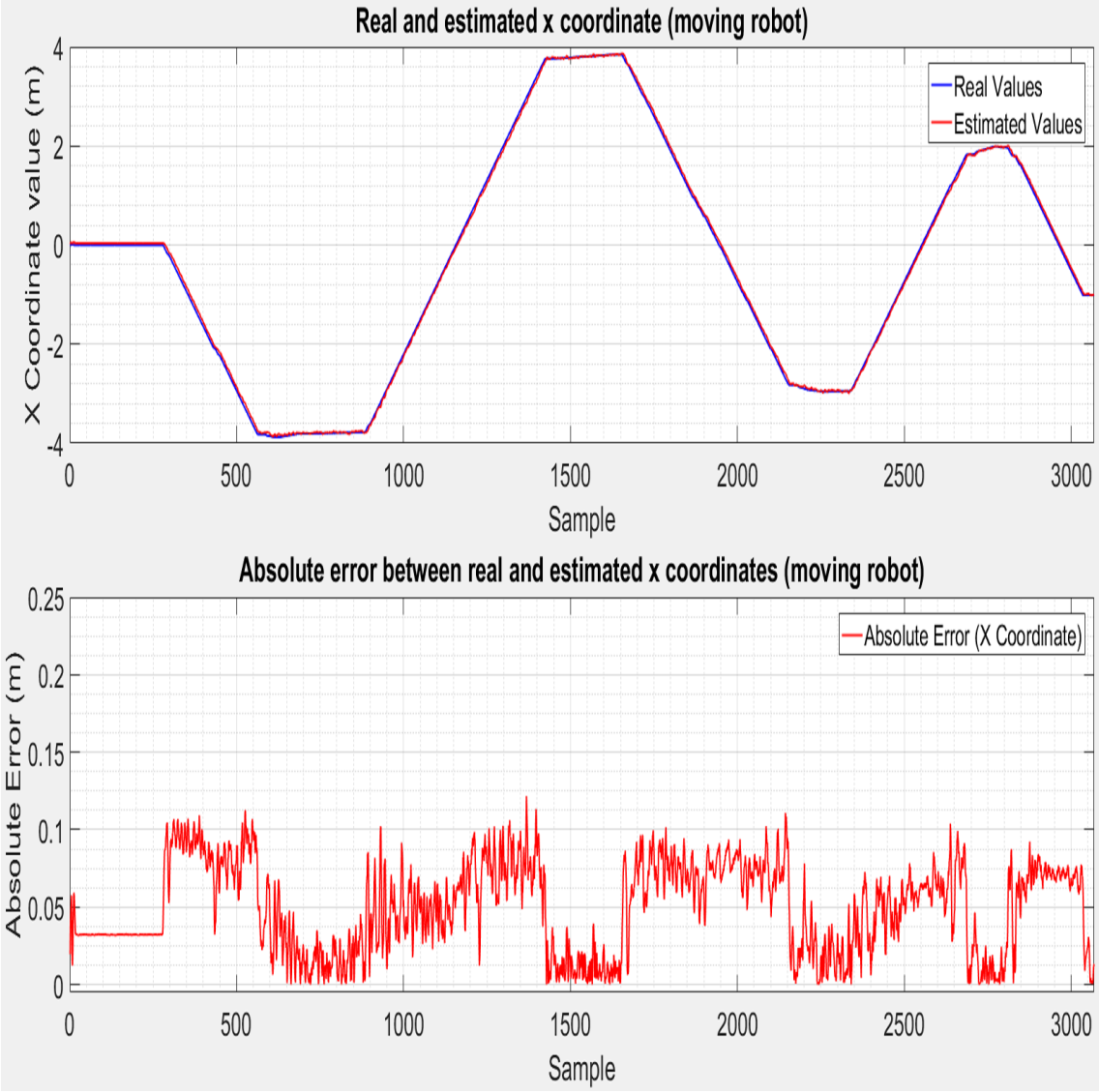


Figure 5.10: Real and estimated x coordinate data (moving robot).

As it is possible to notice, the absolute error tends to be smaller when the target is at rest and when it is moving, the error increases, with the maximum recorded error being approximately 0.12 cm. This behavior can be explained due to the smoothing introduced in the data by the moving average present in the localization algorithm.

Figure 5.11 shows the data referring only to the y coordinate, considering the same path shown previously. In addition, the absolute error between actual and estimated values is shown. As in the previous figure, it is possible to observe again that the absolute error tends to be smaller when the robot is at rest, increasing while it moves, with the maximum recorded error being approximately 0.11 cm.

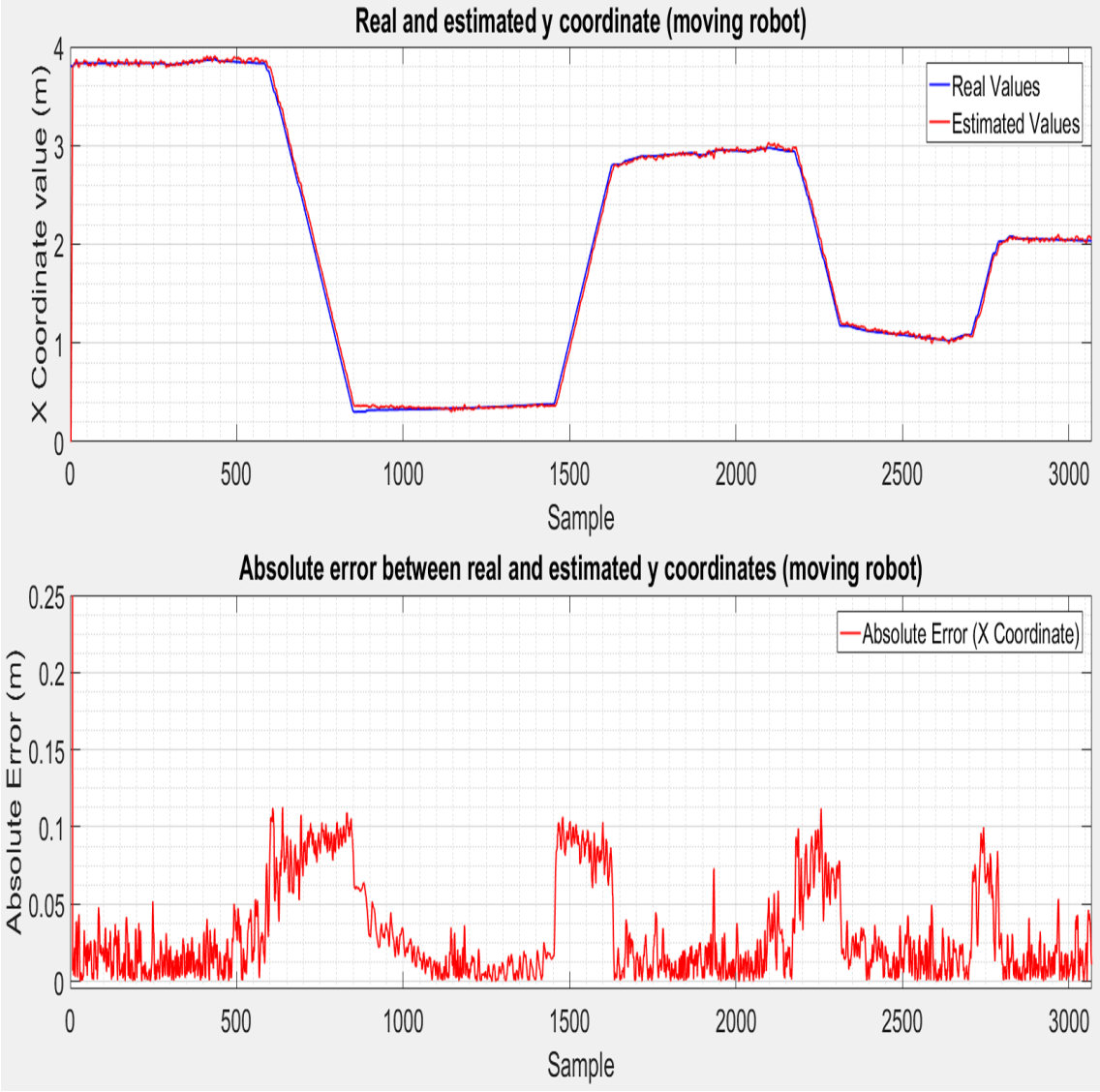


Figure 5.11: Real and estimated y coordinate data (moving robot).

5.2.2 Orientation

The second simulation aimed to evaluate the performance of the orientation estimate while the target is positioned on a fixed point. The target was rotated on its own axis, clockwise and counterclockwise. Figure 5.12 shows the data regarding the target orientation and the absolute error between the real and estimated values, with the maximum error detected being approximately 6.5º.

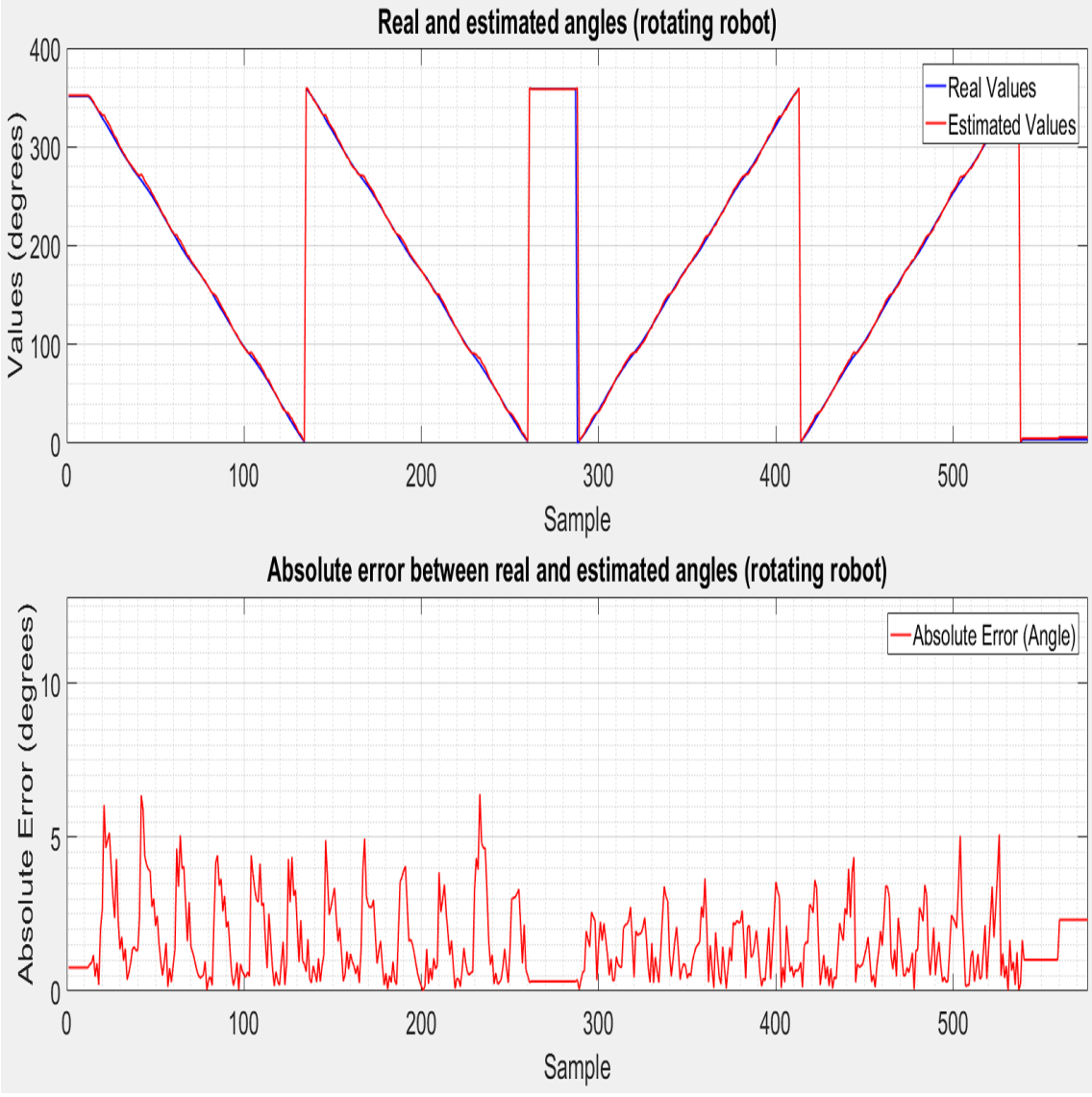


Figure 5.12: Real and estimated angle data (rotating robot).

During a complete rotation of the robot, six error peaks are detected. This behavior occurs due to the low resolution of the camera available in SimTwo.

Finally, to assess the behavior of the orientation estimate during the displacement and rotation of the target, in the third simulation the target was moved around the map as shown in figure 5.13, where the blue line represents the real coordinates obtained during the course, while the red line represents the estimated coordinates.

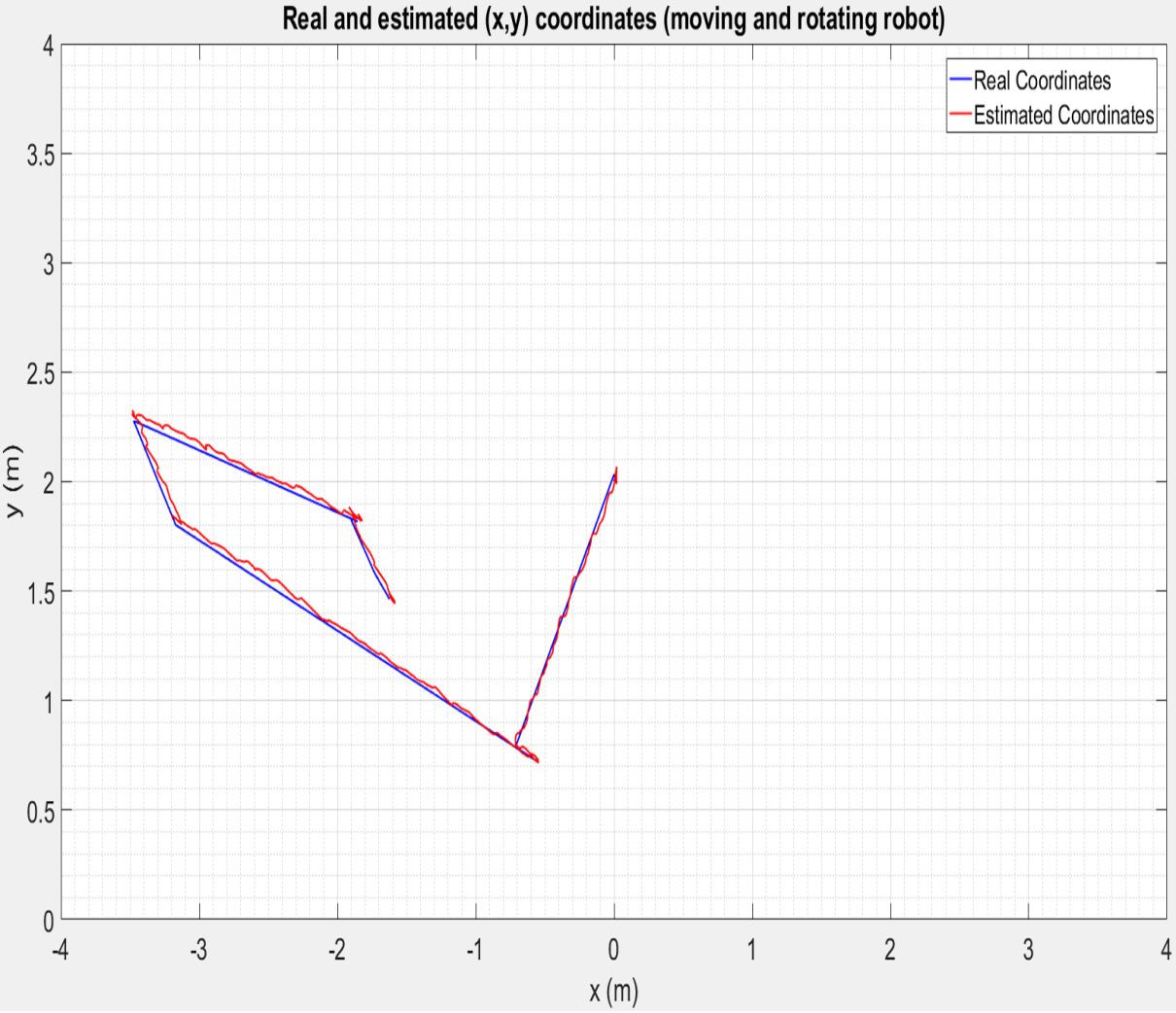


Figure 5.13: Real and estimated (x,y) coordinates data (moving and rotating robot).

Figure 5.14 shows the data regarding the orientation of the target during the course shown above. In addition, the absolute error between actual and estimated values is also

shown.

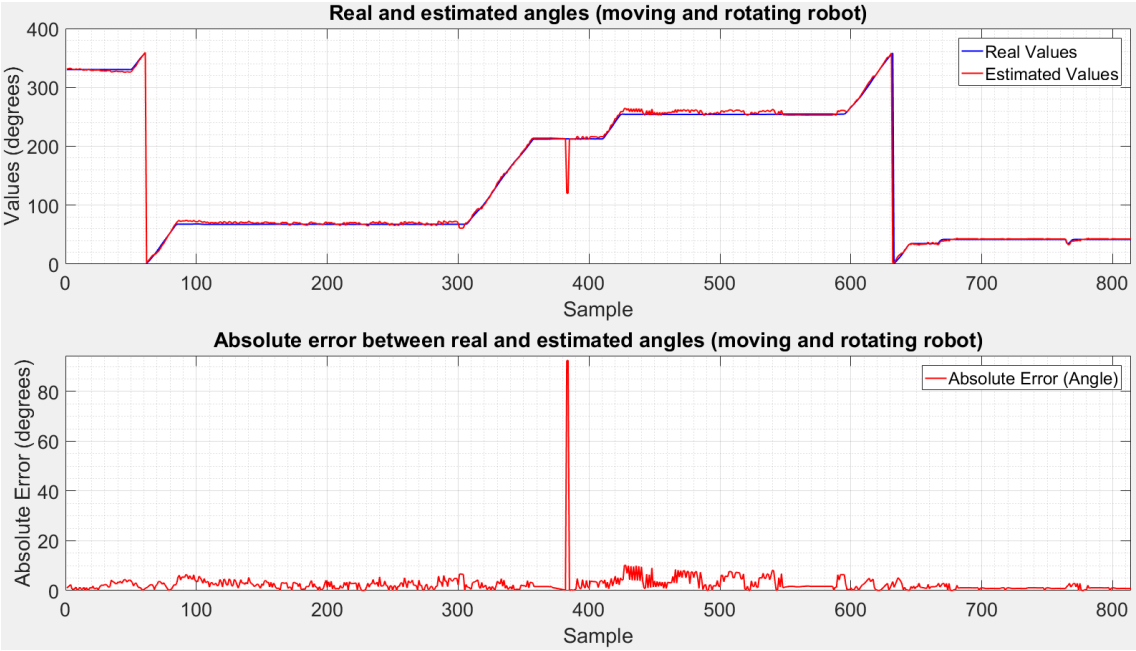


Figure 5.14: Real and estimated angle data (moving and rotating robot).

As it is possible to notice, during the displacement of the target, the error in the orientation tends to increase, presenting error values greater than that of figure 5.12, and, at a certain point, a peak of approximately 92º of error was detected. Despite the error introduced in the system due to the low resolution of the camera, this peak is probably due to a eventual shadow projected on the robot's wheels, which has already been observed in other opportunities during the development of the proposed system. Shadows present in the scene can greatly interfere in the orientation estimation, because as the target detection is made from shades of gray, any object with shadow can be interpreted as part of the target.

Chapter 6

Conclusions

6.1 Summary and Conclusions

Ground-truth systems are extremely useful in the field of robotics, allowing to monitor the movement of robots with high precision, but they generally have a high cost, which can be a limitation to several projects. The main objective of this work was to develop a low-cost ground-truth system capable of determining the location and orientation of mobile robots through the use of a laser sensor and a camera. The system was developed using a realistic SimTwo simulator, which allowed the implementation and testing of the proposed methods. The developed solution can be applied to several types of terrestrial robots, since the methodology used is based on a cylindrical target with colored markers that can be fixed on the robot.

Initially, the methodology to be used in the creation of the proposed system was conceived. A cylindrical target was designed, with the body in a single color and six colored markers equally distributed around its axis.

The proposed system was divided into two subsystems: the laser sensor and the camera. Initially, the laser sensor subsystem was developed, with the ability to cover the entire scene in a two-dimensional way and determine the target coordinates. To discard points referring to objects other than the target, an acceptance zone was created, where

only the points contained within the target's movement area are considered, excluding all points referring to the scene walls. The resulting data was processed in order to be able to detect the edges of the target and, through mathematical relations, determine the coordinates of its center.

In the next step, the camera subsystem was developed, with the ability to track the target at any point in the scene and determine its orientation. Using the coordinates previously determined by the laser sensor, the angle that the motor that controls the camera must move is calculated so that the target is in the center of its field of view. To avoid undesired oscillations during motor positioning, a dead zone has been created that can tolerate a certain difference between the current camera angle and the target angle. Finally, an image processing algorithm based on the color description of the RGB model was created, enabling the estimation of the target's orientation based on mathematical relationships extracted from its physical structure.

The system was validated through simulation, with actual data (provided by the simulator) of the robot's location and orientation compared with the estimated data through the developed system. Through the tests carried out it was verified that the developed system presents an average absolute error of 9 mm on the X axis, 16 mm on the Y axis and 2.63 degrees in the angle estimate for a static target.

The results obtained for the location of the target are satisfactory, being comparable to those achieved by modern commercial systems (as shown in subsection 2.2.1). The results obtained for the target orientation are promising, however they could be better if the camera available in the simulator had a higher resolution. In addition, as SimTwo currently allows only one lighting source in the scene, shadows may appear, compromising image processing. As the camera was recently added to the simulator, new updates with improvements at these points are expected in the future.

6.2 Future Works

During the development of this work, some points were identified that would make it possible to improve the results of the proposed system, such as:

- Use the HSV color space (hue, saturation, value) instead of RGB. When the color description plays a crucial role, the HSV model is advantageous in comparison to RGB, as it is more robust regarding changes in the external lighting. In the HSV space, the "hue" component suffers little to no influence from changes in lighting (such as occasional shadows), however in the RGB space a change in lighting can cause major changes in all component. Thus, the HSV model is preferable in real situations;
- Use a higher resolution camera;
- Add other lighting sources to the scene;
- Due to computational limitations, different image processing techniques (such as segmentation, edge detection, etc.) could not be explored. Thus, tests are suggested for future work.

Finally, it is suggested the real implementation (in hardware) of the ground-truth system developed, which was not possible in this work due to the restrictions caused by the pandemic of COVID-19.

Bibliography

[1] C. Ilas, "Electronic sensing technologies for autonomous ground vehicles: A review," in *2013 8TH INTERNATIONAL SYMPOSIUM ON ADVANCED TOPICS IN ELECTRICAL ENGINEERING (ATEE)*, May 2013, pp. 1–6. DOI: 10.1109/ATEE.2013.6563528.

[2] N. F. B. T. Nunes, "Sistema visual para ground truth de sistemas autónomos," Ph.D. dissertation, Instituto Politécnico do Porto. Instituto Superior de Engenharia do Porto, 2012.

[3] C. D. Metcalf, R. Robinson, A. J. Malpass, T. P. Bogle, T. A. Dell, C. Harris, and S. H. Demain, "Markerless motion capture and measurement of hand kinematics: Validation and application to home-based upper limb rehabilitation," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 8, pp. 2184–2192, 2013.

[4] M. Santos Pessoa de Melo, J. Gomes da Silva Neto, P. Jorge Lima da Silva, J. M. X. Natario Teixeira, and V. Teichrieb, "Analysis and comparison of robotics 3d simulators," in *2019 21st Symposium on Virtual and Augmented Reality (SVR)*, 2019, pp. 242–251.

[5] J. Lima, "Construção de um modelo realista e controlo de um robô humanóide," Ph.D. dissertation, Faculdade de Engenharia da Universidade do Porto, 2009.

[6] J. Gonçalves, "Modelação e simulação realista de sistemas no domınio da robótica móvel," Ph.D. dissertation, Faculdade de Engenharia da Universidade do Porto, 2009.

[7] *Simtwo*, Available in: `https://github.com/P33a/SimTwo`, "Accessed in: 2020-09-17 at 17:00 p.m.".

[8] *Broadcom*, Available in: `https://www.broadcom.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[9] *Adafruit*, Available in: `www.adafruit.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[10] *Banner*, Available in: `www.bannerengineering.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[11] H. D. Young and R. A. Freedman, *Física IV: Ótica e Física Moderna*, 12th ed. São Paulo: Pearson Education do Brasil, 2008, ISBN: 978-85-88639-35-5.

[12] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*. Berlin, Heidelberg: Springer-Verlag, 2007, ISBN: 354023957X.

[13] R. Siegwart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*. Scituate, MA, USA: Bradford Company, 2004, ISBN: 026219502X.

[14] B. Steder, M. Ruhnke, R. Kümmerle, and W. Burgard, "Maximum likelihood remission calibration for groups of heterogeneous laser scanners," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 2078–2083. DOI: `10.1109/ICRA.2015.7139472`.

[15] J. Kim and W. Chung, "Localization of a mobile robot using a laser range finder in a glass-walled environment," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 6, pp. 3616–3627, Jun. 2016.

[16] *Sick sensor intelligence*, Available in: `www.sick.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[17] S. Wagner, N. Fet, M. Handte, and P. J. Marrón, "An approach for hybrid indoor/outdoor navigation," in *2017 International Conference on Intelligent Environments (IE)*, 2017, pp. 36–43. DOI: `10.1109/IE.2017.22`.

[18] *U-blox*, Available in: `https://www.u-blox.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[19] F. Jimenez, *Intelligent Vehicles: Enabling technologies and future developments*. Butterworth-Heinemann, 2017.

[20] F. Caron, E. Duflos, D. Pomorski, and P. Vanheeghe, "Gps/imu data fusion using multisensor kalman filtering: Introduction of contextual aspects," *Information fusion*, vol. 7, no. 2, pp. 221–230, 2006.

[21] *Berryimu*, Available in: `ozzmaker.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[22] D. Becker, F. Thiele, O. Sawade, and I. Radusch, "Cost-effective camera based ground truth for indoor localization," in *2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, IEEE, 2015, pp. 885–890.

[23] C. Fu, C. Mertz, and J. M. Dolan, "Lidar and monocular camera fusion: On-road depth completion for autonomous driving," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 273–278.

[24] C. H. Tong and T. D. Barfoot, "A self-calibrating 3d ground-truth localization system using retroreflective landmarks," in *2011 IEEE International Conference on Robotics and Automation*, IEEE, 2011, pp. 3601–3606.

[25] R. Marchant, P. Guerrero, and J. Ruiz-del-Solar, "A portable ground-truth system based on a laser sensor," in *Robot Soccer World Cup*, Springer, 2011, pp. 234–245.

[26] P. Khandelwal and P. Stone, "A low cost ground truth detection system for robocup using the kinect," in *Robot Soccer World Cup*, Springer, 2011, pp. 515–527.

[27] L. Piardi, J. Lima, and P. Costa, "Development of a ground truth localization system for wheeled mobile robots in indoor environments based on laser range-finder for low-cost systems.," in *ICINCO (2)*, 2018, pp. 351–358.

[28] *Qualisys*, Available in: `www.qualisys.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[29] *Optitrack*, Available in: `https://optitrack.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[30] *Hokuyo*, Available in: `www.hokuyo-aut.jp`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[31] *Oxts*, Available in: `www.oxts.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[32] *Gazebo, robot simulation made easy*, Available in: `http://gazebosim.org`, "Accessed in: 2020-09-10 at 11:00 p.m.".

[33] *Ros*, Available in: `https://www.ros.org/`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[34] *Coppelia robotics*, Available in: `www.coppeliarobotics.com`, "Accessed in: 2020-09-11 at 00:00 a.m.".

[35] L. Nogueira, "Comparative analysis between gazebo and v-rep robotic simulators," *Seminario Interno de Cognicao Artificial-SICA*, vol. 2014, no. 5, 2014.

[36] L. Pitonakova, M. Giuliani, A. Pipe, and A. Winfield, "Feature and performance comparison of the v-rep, gazebo and argos robot simulators," in *Annual Conference Towards Autonomous Robotic Systems*, Springer, 2018, pp. 357–368.

[37] *Webots*, Available in: `www.cyberbotics.com`, "Accessed in: 2020-09-17 at 2:00 a.m.".

[38] *Inesc tec*, Available in: `https://www.inesctec.pt/`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[39] C. Paulo, G. José, L. José, and M. Paulo, "Simtwo realistic simulator: A tool for the development and validation of robot software," *Theory and Applications of Mathematics & Computer Science*, vol. 1, no. 1, pp. 17–33, 2011.

[40] A. R. C. d. Andrade, "Sistema robótico autónomo para ambientes de terapia de iodo," Ph.D. dissertation, 2015.

[41] G. Amaral and P. Costa, "Simtwo as a simulation environment for flight robot dynamics evaluation," *U. Porto Journal of Engineering*, vol. 1, no. 1, pp. 80–88, 2015.

[42] A. F. de Domingues *et al.*, "Ferramenta para desenvolvimento de inteligência em jogos simulados em ambiente simtwo," 2019.

[43] *Glscene for delphi c++ builder*, Available in: `http://glscene.sourceforge.net/wikka/`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[44] *Open dynamics engine*, Available in: `https://www.ode.org/`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[45] *Pascal script for delphi*, Available in: `https://www.remobjects.com/ps.aspx`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[46] *Synedit*, Available in: `https://github.com/SynEdit/SynEdit`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[47] *Omnixml*, Available in: `https://github.com/mremec/omnixml`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[48] *Rx library for delphi*, Available in: `https://sourceforge.net/projects/rxlib/`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[49] *Sick sensor intelligence*, Available in: `www.sick.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

[50] S. Ozdemir, *Principles of data science*. Packt Publishing Ltd, 2016.

[51] *Desmos*, Available in: `https://www.desmos.com`, "Accessed in: 2020-09-18 at 2:00 a.m.".

Appendix A

Project - GitHub

Below is the GitHub link with the developed work:

`https://github.com/Soaares1/Ground-Truth`