Máster en Ingeniería de Sistemas y Control
UNED- Universidad Complutense de Madrid

INDOOR ROBOT MOBILE POSITIONING BY RANGE AND BEARING MEASUREMENTS

Ester Garrido Estrada
Advisor: Dr. David Moreno Salinas
2016-2017

September 2017

THIS PAGE INTENTIONALLY LEFT BLANK

Master	Systems and control engineering
Title	Indoor robot mobile positioning by range and bearing measurements
Project Type	Master thesis
Author	Ester Garrido Estrada
Advisor	Dr. David Moreno Salinas

THIS PAGE INTENTIONALLY LEFT BLANK

Autorización

Autorizamos a la Universidad Complutense y a la UNED a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firmado:

Ester Garrido Estrada

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGEMENTS

To my life-coach, my *sambo* Fernando: because I owe this thesis to you. Many thanks for your unfailing support and continuous back-up.

I would like to thank my thesis advisor Dr. David Moreno for their feedback, cooperation and of course, his patience teaching me everything I did not know. In addition I would like to express my gratitude to the staff of the National Distance Education University (UNED) for all the support.

A very special gratitude goes out to all my company colleagues for the help and support every day over this thesis development.

And finally, last but not least, my parents, Manuel and Julia, who always are there to help me out and show me the way ahead. Thanks both for trusting on my capabilities even when I think it is not going to be possible.

Thanks all for your encouragement!

Tack så mycket för din uppmuntran!

THIS PAGE INTENTIONALLY LEFT BLANK

PROJECT ABSTRACT

Mobile robot positioning is a fundamental problem in robotics applications, which has been investigated over years. Nowadays, the satellite navigation systems, such as the well-known Global Positioning System, can be used to solve this issue in a fast, inexpensive and easy way. However, because of the nature of these systems, an alternative solution needs to be found for specific environments, i.e., indoor, urban or underwater, where they are not available or reliable enough.

This project tackles with the problem about control and determination of indoor mobile robot positioning. Thus, multiple technologies, such as inertial navigation and ultrasonic sensors, joint with different positioning methods, for instance, relative and absolute estimations, are applied in order to assess which ones provide results closer to reality. In addition, different mathematical techniques, such as trilateration and triangulation, are implemented to solve accurately the location problem.

Several real-world experiments are done to demonstrate the functionality of the robot and the localization methods. The main differences between these tests are the mathematical techniques and the number of beacons used.

Finally, the results of the different experimental tests are compared between them as well as with some theoretical references, i.e., the Fisher Information Matrix and the Cramer-Rao Lower Bound. It is concluded that it is possible to reach quite accurate positioning, even though the low cost and non- reliable technologies that are used.

KEYWORDS

Mobile Robot, GPS, Positioning, Ultrasonic Sensors, Indoor, Triangulation, Trilateration, Least Squares Approaches

RESUMEN DEL PROYECTO

El posicionamiento de los robots móviles es hoy en día un problema fundamental en robótica que, a pesar de haber sido investigado durante años, aún está por resolver en determinadas situaciones y entornos.

Se ha demostrado que los sistemas de navegación por satélite, como el conocido GPS, pueden ser empleados para obtener la posición de un objeto móvil de forma rápida, sencilla y sin un coste elevado. Sin embargo, debido a la naturaleza de estos sistemas, éstos no se encuentran disponibles o no resultan fiables en todos los entornos, como por ejemplo en espacios interiores, bajo el agua o en zonas urbanas.

Este proyecto pretende resolver el problema de control y determinación de la posición de un robot móvil en un espacio interior. Para ello, se evalúan múltiples tecnologías como por ejemplo sensores inerciales y de ultrasonidos, junto con diferentes métodos de posicionamiento (estimaciones relativas y absolutas), con el objetivo de determinar la solución que proporciona los valores más cercanos al caso real. Además, se aplican algunas técnicas matemáticas como son la trilateración o la triangulación.

De este modo, se han llevado a cabo distintos experimentos reales para demostrar la funcionalidad del robot y de los métodos de localización seleccionados. Los experimentos se diferencian entre sí por el número de *beacons* empleados y las técnicas matemáticas utilizadas para obtener el posicionamiento del objeto.

Finalmente, los resultados obtenidos se comparan entre ellos y también con varias referencias teóricas como el límite inferior de Cramer-Rao o la matriz de información de Fisher, concluyendo que a pesar de emplear tecnologías de bajo coste y técnicas matemáticas no muy sofisticadas, es posible obtener buenos resultados de posicionamiento en espacios interiores.

PALABRAS CLAVE

Robot móvil, GPS, Posicionamiento, Ultrasonidos, Interior, Triangulación, Trilateración, Aproximación de Mínimos Cuadrados

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

1. INTRODUCTION

1.1. Background/ Motivation

The robotic developments based on mobile platforms have grown a lot over the last few years, becoming an essential part in many sectors such logistic, entertainment, medical service, transportation, agriculture or even security and defence. The *Sojourner* mobile robot used during a NASA mission to explore Mars in 1997 [36] and the Mobile Manipulation Robot *Momaro* applied to solve disaster-response tasks [37] are two good examples of how far the mobile robot technologies have been improved.

The main property of the mobile robot is to include any kind of mechanism of locomotion, which allows it to move freely in any environment within a pre-defined operating area. This implies that the robot mobile needs to know information about its surroundings, to react autonomously in the environment and to determine its position regarding to some inertial reference frame. Accordingly, sensorial systems have a crucial role, allowing mobile robots to successfully deal with any specific task.

Despite the technological progress of robot sensing systems in the last years, there are still many challenging problems related to navigation that make mobile robots not useful enough for some specific applications. The mobile robot positioning is one of these problems and currently an important research field.

Nowadays, the mobile robot outdoor positioning problem is mostly covered with satellite navigation systems, such as the well-known American Global Positioning System (GPS), the Russian GLONASS, the Chinese BeiDou or the European Galileo. The operating principle of these systems is based on pinpoint the target geo-spatial position, employing the information provided by constellations of Earth-orbiting satellites. Thus, the ground receiver gathers up the satellite broadcast signal, containing the satellite exact location and time, and computes the ground position by trilateration.

Nonetheless, the positioning problem is not entirely solved by satellite navigation systems due to these ones show some weaknesses related to the implementation framework and the relation cost/accuracy.

When it comes to urban, indoor or underwater environments, the satellite navigation systems cannot work properly due to the difficulty of acquiring the satellite signal by the receiver. Besides, the accuracy requirements are dissimilar between outdoor and indoor applications, being the indoor ones the most demanding. Likewise, most navigation solutions designed for indoor environments can also be used in open sky, although their capabilities are degraded, and vice versa.

Up to now, several solutions have been proposed differing in the type of technologies and the perceived data. However, some of them are high expensive and some of them are not accurate enough because of the application requirements.

1.2. Scope

This project examines several positioning techniques for mobile robot positioning in indoor scenarios. To this goal, an ad-hoc wheeled robot is developed using open-source Arduino electronics and a simple tracked chassis, which uses a couple of rubber tank treads. The reasons using the Arduino technologies are the low cost of the installation, the simple programming environment and the design flexibility.

On the one hand, it is assessed the relative positioning technique based on inertial sensors. The dead-reckoning is probably the most common and ancient positioning technique. However, it introduces systematic and non-systematic errors, which are accumulated over time and entail large inaccuracies, making this technique not suitable when the application time is longer than few minutes. However, it could be useful if having high-accurate and expensive inertial sensors instead. In this case, the inertial navigation analyzed is based on a combination of accelerometer and gyroscope, both sensors compatible with Arduino electronics.

On the other hand, it is appraised the absolute or reference-based positioning based on passive beacons. This is one of the most common technique in those environments where the GNNS-based positioning cannot be applied because lack of sight's line between satellites and the receiver. To carry out this project, it is used passive beacons and ultrasonic sensor modules. Besides, it is applied different mathematical methods based on Time of Arrival (TOA) and Angle of Arrival (AOA) measurements. Some mathematical algorithms are modified to deal with the sensors measurement uncertainty.

In addition, it is done multiple bi-dimensional experimental tests. These tests have different scenario configurations, which mean diverse number of passive beacons and ultrasonic sensors modules, and they also execute different positioning techniques in order to reach the project goals.

1.3. Goals

The present project has clearly four objectives. Firstly, it is intended to demonstrate the suitable use of low cost technology for the quite demanding geo-localization problem. Secondly, it is tested different positioning techniques such as triangulation and trilateration, and multiple mathematical algorithms. The results obtained are compared between them in order to identify pros and cons of each method. Thirdly, it is evaluated the effect of reducing the number of sensors and measurements performed. Finally, it is planned to compare the effectiveness performing range and bearing measurements in optimal and non-optimal positions.

1.4. Contents

This document is organized as follows:

Section 1. Introduction.

Section 2. System description. This section includes a brief description of the software and hardware of the mobile robot, including chassis properties, microprocessor and sensors and servomotors. Furthermore, it is also described the robot assembly and wiring diagram for the different designs tested.

Section 3. Robot modelling. It is described the mobile robot kinematic model obtained from experimental data.

Section 4. Methods for robot mobile positioning. Main relative and absolute positioning techniques are detailed in this section. Moreover, inertial sensors test results are compared with the kinematic model results.

Section 5. Mathematical approaches and problem formulation. This section describes the Least Squares approach used in 2D trilateration and the geometric circle intersection approach applied in 2D triangulation. The problem formulation is also detailed.

Section 6. Assumptions and limitations. This section summarizes the general assumptions and limitations identified along the project development.

Section 7. Simulated and experimental results. The main results got in the experimental test are detailed in this section. These results are divided into three different cases depending on the number of passive beacons and the mathematical method used to find the solution.

Section 8. Robot Development Problems. This section summarizes the main problems found during project development with the technologies used.

Section 9. Conclusions. This section details the project general conclusions and future work.

1.5. Acronyms and nomenclature

List of acronyms

AOA	Angle Of Arrival
CW	Counter Wise
CCW	Counter Clock Wise
CRLB	Cramer-Rao Lower Bound
DC	Direct Current
EEPROM	Electrically Erasable Programmable Read-Only Memory
FIM	Fisher Information Matrix

GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IDE	Integrated Development Environment
IO	Input/ Output
IT	Iteration
LLS	Linear Least Square
LOS	Line Of Sight
LSE	Least Squares Estimator
MEMS	Micro Electro Mechanical Sensor
MLS	Maximum Likelihood Estimators
MSE	Mean Square Error
NASA	National Aeronautics and Space Administration
OLS	Ordinary Least Squares
PWM	Pulse Width Modulation
RX	Receiver
SRAM	Static Random Access Memory
TDOA	Time Differences Of Arrival
TOA	Time Of Arrival
TST	Experimental TeST
TX	Transmitter

List of symbols

$\{I\}$	Inertial Reference Frame
$m \epsilon \mathbb{N}$	Scenario dimensions
$n \epsilon \mathbb{N}$	Number of beacons
$k \epsilon \mathbb{N}$	Number of positions where robot scans
x	x- Axis
y	y-Axis
O	Axes Origin
$\mathbb{R}$	Set of Real Numbers
$\mathbb{R}^m$	Vector with Real Entries of Dimension m
$\mathbb{R}^{mxn}$	Matrix with Real Entries of Dimension mxn
$\|\cdot\|$	Euclidean Norm
$E\{\cdot\}$	Expectation

$\delta(\,)$	Operator diag
$\nabla_{\mathbf{p}} \log\, \mathcal{L}(\boldsymbol{p}\|\boldsymbol{z})$	Gradient of the natural logarithm of the likelihood function
σ	Standard Deviation
σ^2	Variance
H	Capital Case symbols represents Matrices
$\boldsymbol{h}$	Lower Case Boldface symbols represents vectors
$\boldsymbol{p_i} \in \mathbb{R}^{\mathrm{m}}$	Robot coordinates
$P \in \mathbb{R}^{\mathrm{mxn}}$	Matrix of robot coordinates
$\boldsymbol{pb_i} \in \mathbb{R}^{\mathrm{m}}$	Beacon coordinates
$PB \in \mathbb{R}^{\mathrm{mxn}}$	Matrix of beacon coordinates
Δx	Robot linear displacement on x-axis
Δy	Robot linear displacement on y-axis
$\Delta\theta$	Robot angular displacement
$\overline{\boldsymbol{d}}$	Vector of measured squared ranges
d_i	Squared range measurement between beacon i^{th} and the robot
r_i	Range measurement between beacon i^{th} and the robot
$\boldsymbol{\varepsilon}$	Vector of range measurement errors
$A \in \mathbb{R}^{mxn}$	Matrix containing beacon positions
$\boldsymbol{b} \in \mathbb{R}^{n}$	Column vector with distance measurements
$\widehat{\boldsymbol{p}}_{\boldsymbol{L-LS}}$	Estimated robot position by LLS
A^T	Matrix Transpose
A^{-1}	Matrix Inverse
I	Identity Matrix
$\Sigma_{L-LS} \in \mathbb{R}^{mxm}$	Covariance matrix based on LLS estimation
$\mathbf{r}$	Column vector containing the diagonal entries of Σ_{L-LS}
$\mu_{L-LS} \in \mathbb{R}^{\mathrm{m}}$	Mean estimate error based on LLS estimation
ϕ_i	Bearing angle between the initial reference and the robot position $\boldsymbol{p_i}$
ϕ_{ij}	Bearing angle between robot positions $\boldsymbol{p_i}$ and $\boldsymbol{p_j}$
$\mathbf{c}(x_{ij}, y_{ij}) \in \mathbb{R}^{\mathrm{m}}$	Circle centre coordinates of circle passing through $\boldsymbol{p_i}$ and $\boldsymbol{p_j}$
R_{ij}	Radius of the circle passing through $\boldsymbol{p_i}$ and $\boldsymbol{p_j}$
T_{ij}	Cotangent of ϕ_{ij}
k_{ij}	Variable containing circle parameters (centre and radius)

THIS PAGE INTENTIONALLY LEFT BLANK

2. SYSTEM DESCRIPTION

This section provides a brief description about the hardware components and software used in this project. Furthermore, it is also mentioned how the mobile robot is assembled and the different robot designs developed, including a summary of the physical components, their interrelationship and the wiring diagram used on each case.

2.1. Hardware

2.1.1. Robot Chassis

The robot base used in this project is the Rover 5 from Dagu Electronics. This is a weightless tracked chassis made with rugged white plastic. It includes four traditional wheels, two on each side of the platform. Each wheel pair is connected to each other by a rubber tank tread.

The platform contains a couple of independent DC motors, TFK 280SC-21138-45, with 87:1 ratio gearboxes and without encoders. These DC motors can move the robot chassis in any indoor environment and allow it to reach a maximum speed of 0.277 m/s. Each DC motor encloses a red and black cable for motor power.

A unique quality of this robot platform is that its clearance can be adapted by modifying the angle at which the gearbox is mounted. The way of doing that is unscrewing the metal bracket inside the chassis, removing the gearbox and reinserting it in a desired angle, taking into account that the gearbox can rotate in 5 degrees increment. Consequently, the robot base height can vary regards to the floor from 0 to 0.038 m. In this project, it is used the robot flattened configuration, which means that the gearboxes mounted angle is 180° to each other.

Regarding the chassis physical description, it has an approximate weight of 0.720 kg without batteries, and its outer dimensions are 0.245 m x 0.225 m x 0.074 m (L x W x H), according to the robot flattened configuration mentioned above.

At the bottom of this chassis, it incorporates a couple of noise suppression coils and a battery holder designed for 6x AA batteries.

Finally, due to the Rover 5 chassis does not include a top plate, the electronic components are mounted in ad-hoc white plastic plate, which dimensions are 0.173 m x 0.086 m x 0.003 m (L x W x H). This top plate is fixed to the base using electronic tape because it can be easily fitted or removed.

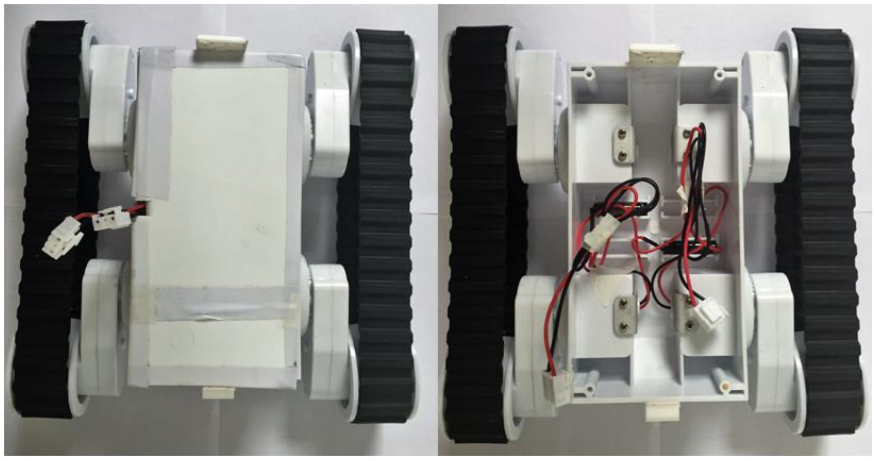

Figure 1. Rover 5 chassis appearance

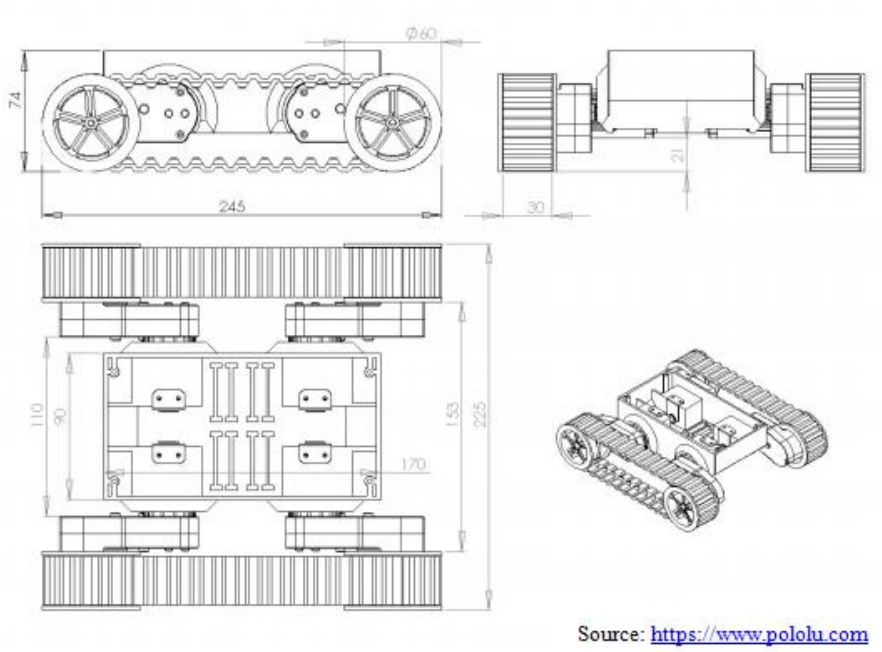

Figure 2. Rover 5 chassis dimensions

2.1.2. Robot Core

Arduino UNO microcontroller board is the Rover 5 core since all the software used to control the robot is implemented on it. This board computer is based on the high-performance ATmega328 microchip, which has 32 KB of flash memory (0.5 KB of them used by bootloader), 2KB of SRAM and 1KB of EEPROM. The Arduino programming code is saved in the flash memory.

According to the board recommended operating voltage range (7-12 V), the Arduino UNO is powered with 9V coming from 6x AA batteries. Additionally, this board can also be powered via USB with +5V DC, which may provide less than +5V DC to the pin and make the board unstable. Additionally, the Arduino UNO board has four power

pins and all of them are used in this project: (1) the input voltage, (2) +5V DC output voltage, (3) +3V DC output voltage, and finally, (4) the ground pins.

In reference to the Arduino UNO pins, this board has 14 digital and 6 analog pins. Meanwhile the digital ones can be configured as inputs/outputs pins and they run with +5V DC, the analog pins are only available as inputs.

Regarding communication, the Arduino UNO has several capabilities to communicate with a computer or another microcontroller. In this case, it is always used the serial communication, which uses the digital pins 0 (RX) and 1 (TX) of the Arduino UNO board.

2.1.2.1. Expansion Shields

Expansion shields are modular boards that plugged in on top of the Arduino UNO board to increase its capabilities. There are several types of shields to solve many different problems, so it is possible to combine these shields by mounting them on top each other.

It is utilized two Arduino shields in order to reduce the Arduino UNO board weaknesses. On the one hand, Arduino UNO board cannot control any type of motor; therefore, the Arduino Motor Shield is used to control the performances of the DC motors pair included at the Rover 5 chassis. This shield is directly plugged in on top of Arduino UNO board. Thus, there are pins that are always in use by the shield making possible to control the motor direction, the speed (PWM), start and stop and monitor the power provided to each motor. Arduino Motor Shield has a couple of channels to connect two motors or one stepper motor, and it can be powered directly with any external power source. In this case, the batteries (+9V DC) are connected to this motor shield directly. On the other hand, it is used the IO expansion shield v.5 from DF Robot to overcome the lack of digital and analog pins. In this case, this shield is plugged in on top of the previous shield and it is not directly powered.

Figure 3 shows the expansion shields applied in this project and how they are connected to the Arduino UNO board.

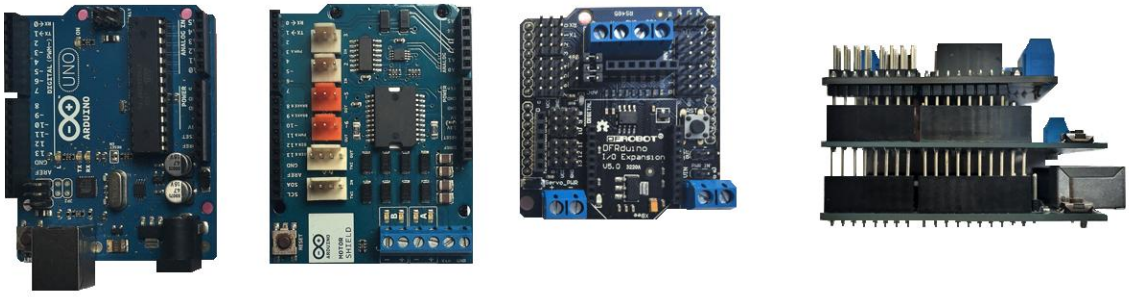

Figure 3. Rover 5 base board and expansion shields

2.1.3. Robot Distance Sensors

There are several types of distance sensors for range measurement and tracking, having all of them similar operating principle. Overall, this type of sensors consists of a transmitter sending out an electromagnetic radiation or electrostatic field, and a receiver that captures the signal back and analyses it.

The distance sensors most commonly used in robotics are the ultrasonic and the infrared (IR) sensors. They are also considered proximity sensors.

The sensor type chosen to carry out this project is the ultrasonic sensor. The reason for that is that its performance fits well with a set of obstacle types, while infrared sensors only work perfectly for specific materials [30]. Furthermore, ultrasonic sensors are appropriate for non-contact range measurements because they provide high accuracy and stable readings, regardless the different light conditions.

2.1.3.1. Ultrasonic sensors (HC-SR04)

Ultrasonic sensors generate high frequency sound wave that travels through the medium up to it bounces off with an obstacle. Then, the sound wave echo comes back until it is received by the ultrasonic sensor receiver.

As previously mentioned, among the ultrasonic sensors special features are their capability to detect several objects - regardless the object material and size-, their output is linear according to the distance between the sensor and the detected object and they can work in any conditions. Contrary, they could easily provide false-negative outputs, therefore, the scenario dimensions need to be well-limited, and it is required a high-density object surface to make the entire sound wave comes back to the sensor receiver.

To measure the distance between the sensor position and the object, i.e. the distance the sound wave has travelled, it is considered the sound has to go ahead and back, and the speed of sound, which depends on the type of medium and its temperature (see Ec.01). It is assumed that the speed of sound is approximately 340 m/s, considering sea level, temperature of 15 ℃ and 50% humidity.

$$d = \frac{Time\ x\ SpeedOfSound}{2} \qquad \text{Ec.01}$$

The ultrasonic sensor used in this project is the HC-SR04 from Itead Studio. This includes a transmitter, receiver and a control circuit. Its minimum range measurement is 0.02 m and the maximum 4 m. Furthermore, it provides a maximum accuracy of 3 mm along its entire measurement interval.

The HC-SR04 has four important pins: (1) voltage input (minimum power supply +5V DC), (2) trigger pulse (input), (3) echo pulse (output) and (4) ground.

Regarding the sensor operation principle, when the trigger pin receives a pulse higher than +5 V_{DC} during 10 us at least, the ultrasonic module sends eight high frequency sound waves, usually 40 kHz, and waits if there is a pulse signal back. If so, the echo

pulse width, which is proportional to the distance traveller, is set higher than +5 V_{DC}. It is considered the duration of the high output to compute the distance between the sensor and the object using Ec.01.

A couple of issues need to be considered at the time measurements. On the one hand, the object needs to have a smooth surface and be made with an appropriate material for ultrasound waves to be detected. On the other hand, the time between measurements has to be higher than 60 ms, to avoid trigger and echo signal overlapping.

HC-SR04 sensor and its timing diagram are shown in Figure 4.

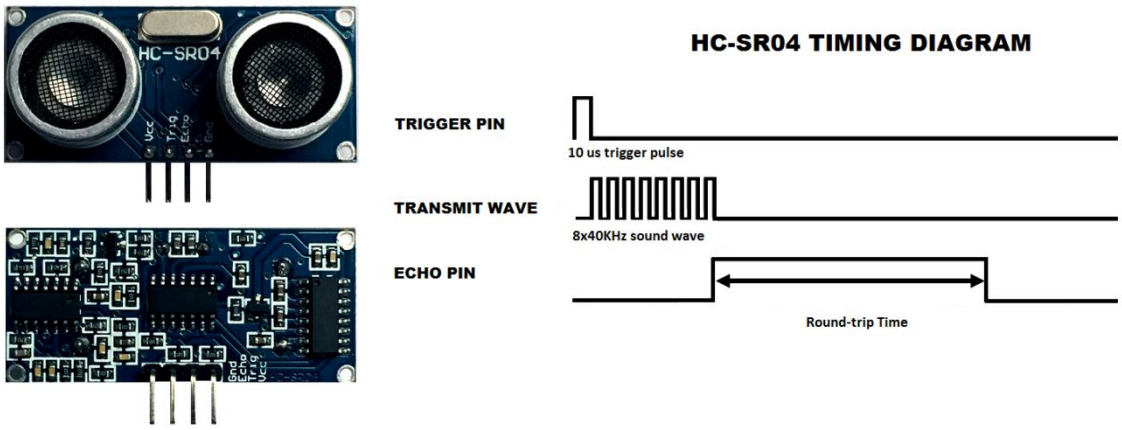

Figure 4. HC-SR04 appearance and timing diagram

A simple test is performed to estimate the standard deviation of these sensors. For this purpose, the sensor is located in front of the obstacle, approximately 0.3 m, and one hundred samples are collected. Then, it is computed the mean value, 0.2931 m, the variance, $\sigma^2 = 1.5886 \cdot 10^{-6}$ m^2, and the standard deviation, $\sigma = 1.2604 \cdot 10^{-3}$ m.

Lastly, it is important to note that two different robot designs are applied in this project. Each design involves different number of ultrasonic sensors. The first design involves three ultrasonic sensors, whereas the second one includes a single ultrasonic sensor mounted over a rotating base, which makes this design less costly.

2.1.4. Robot servomotor

A rotary actuator is needed to fulfil the second robot design requirements (see section 2.1.5.2). This is why it is utilized the micro analog servomotor SM-S2309S from SpringRC. This servomotor includes a plastic gear based on a servo plastic arm fixed using a standard screw. Then, the HC-SR04 ultrasonic module is positioned over the servo arm and fixed to it by a couple of plastic brackets. Consequently, the ultrasonic module can be positioned in several specific angular positions by sending different servo coded signals.

This servomotor can only rotate 180° clockwise (see A.06 Table 3). It requires to be powered with +5 V_{DC} and the maximum speed corresponding to this voltage is

approximately 0.09 sec/60°. Finally, the SM-S2309S is a small and light device, its weight is around 0.01 kg and its dimensions are 0.023 x 0.012 x 0.022 m.

Figure 5. Servomotor SM-S2309S

Again, it is performed a simple test to estimate the standard deviation of the angle provided by the servomotor. Initially, it is placed an obstacle at $97\,°$ regarding the robot orientation, which in this case is fixed to $0\,°$. Next, it is computed the upper and lower limit angle, i.e. the first angle from which the obstacle is detected and the last one. The final angle is obtained computing the mean of all angles within the interval identified. This process is repeated 100 times. Then, it is calculated the mean value, $97\,°$, the variance, $\sigma^2 = 1.4492\,°^2$, and the standard deviation, $\sigma = 1.2038\,°$.

2.1.5. Robot assembly

In this section it is described the robot assembly and the wiring diagram for both ultrasonic sensor designs mentioned in section 2.1.3.1.

Both robot designs have similar functionalities. Firstly, the robot can move forward, backward and perform CW or CCW turn, depending on the control signals. It has to be noted that the robot performs no large displacements. Secondly, the robot can move autonomously avoiding all the obstacles it finds on its way. In case it reaches an obstacle, the robot automatically turns CW according to a pre-defined angle. Finally, the designs are quite simple to assemble and disassemble, as well as to modify or improve.

It is important to highlight that both designs use same robot chassis, the Rover 5.

2.1.5.1. Design 1

As previously mentioned, design 1 includes three ultrasonic sensor modules, HC-SR04. One of them is fixed to the front of the Rover 5 chassis and the others two are fixed to each side of the base. These ultrasonic sensors keep stationary. Moreover, it is used a plastic bracket and electronic tape to fix the ultrasonic module to the Rover 5 chassis.

The Arduino UNO board and the extension shields are fixed to the top plate of the Rover 5 chassis with electric tape. It is used a breadboard to make the circuit flexible and not chaotic. This solderless board is fixed to the top plate chassis using electric tape. In addition, the batteries are also located at the top plate.

All the electronic components are connected to each other using jumper wires.

Figure 6 shows the installation appearance for the robot design 1.

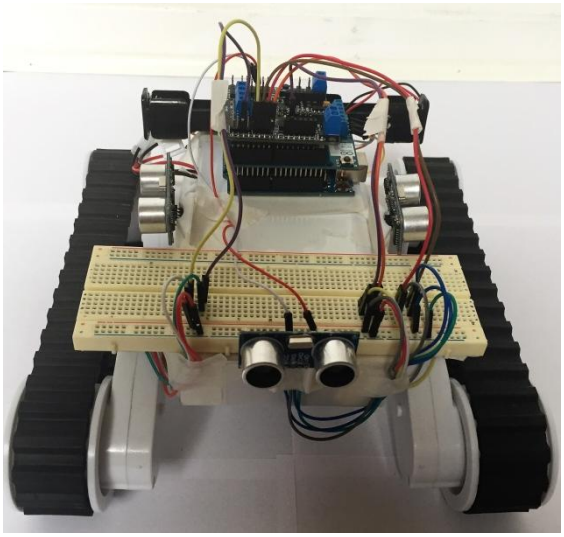

Figure 6. Robot design 1

Finally, a simple wiring diagram is shown in Figure 7.

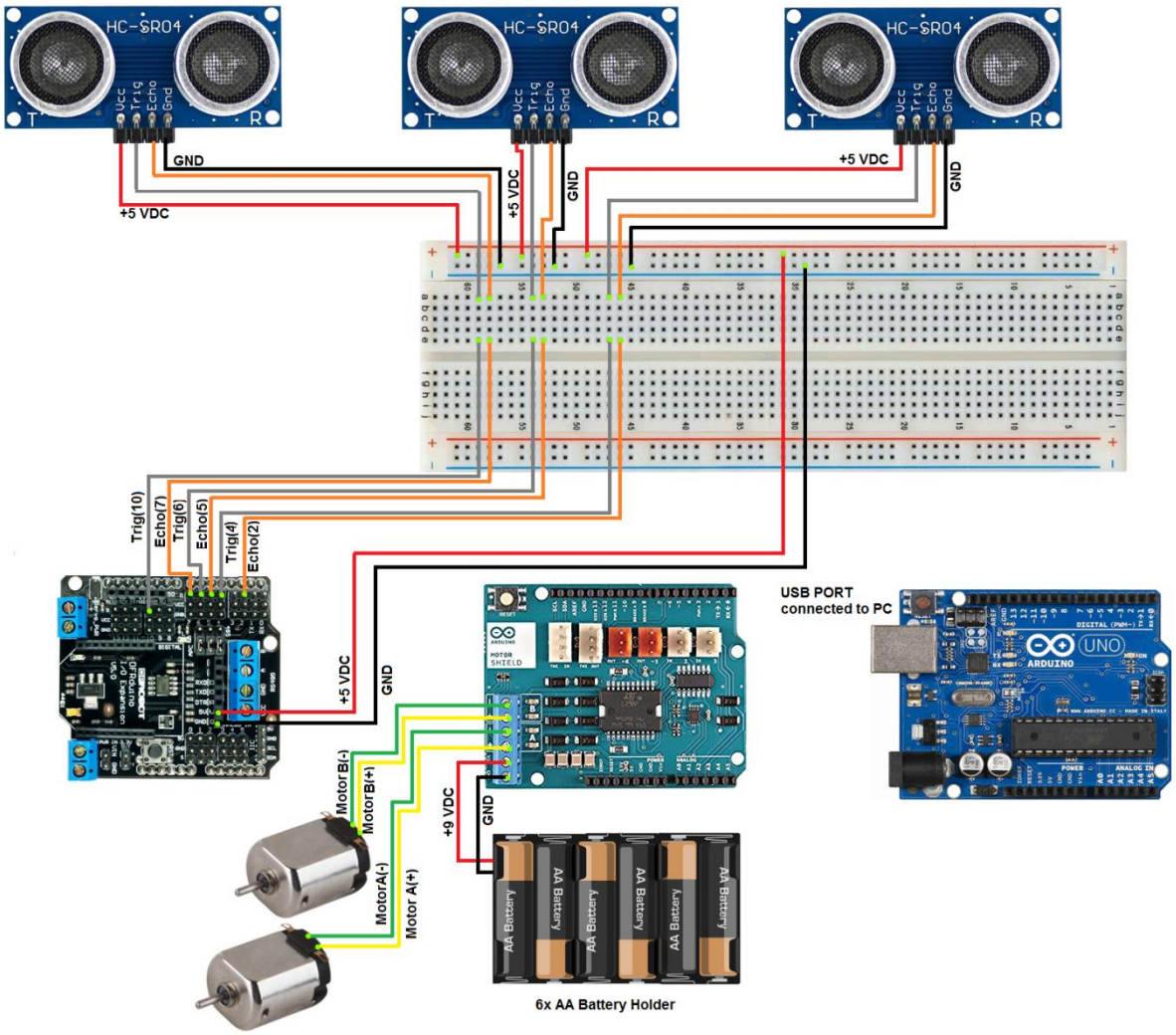

Figure 7. Robot design 1- Wiring diagram (not to scale)

Figure 7 demonstrates the connections between the external power source, the Arduino UNO board and the extension shields. Moreover, it also takes into account the relations between all components such as the ultrasonic modules.

2.1.5.2. Design 2

This design includes a unique ultrasonic sensor module, HC-SR04, mounted over a servo plastic arm, which means that this sensor rotates at the same time than the servomotor is rotating. The way of fixing this sensor to the servomotor arm is using a couple of plastic brackets. In this case, it is developed a raised area over the top chassis plate where it is placed the servomotor and the ultrasonic module. The reason for that is to avoid false-positive lectures because the ultrasonic sensor detects electric components or wiring.

Once again, the Arduino UNO board and the extension shields are fixed to the top plate of the Rover 5 chassis with electric tape. It is used a breadboard to make the circuit flexible and not chaotic. This solderless board is fixed to the top plate chassis using electric tape. In addition, the batteries are also located at the top plate.

All the electronic components are connected to each other using jumper wires.

Figure 8 shows the installation appearance for the robot design 2.

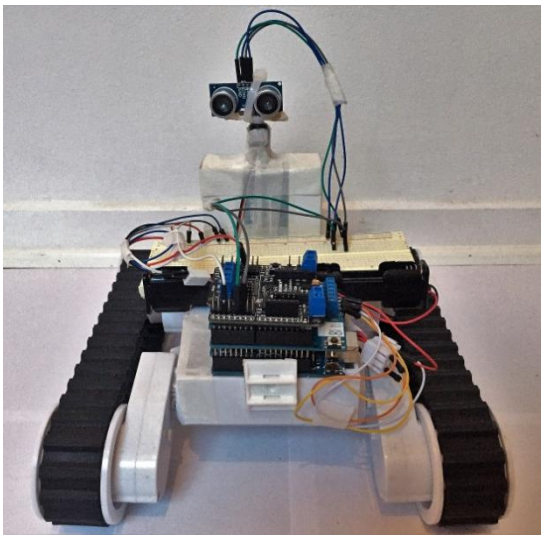

Figure 8. Robot design 2

Finally, a simple wiring diagram is shown in Figure 9.

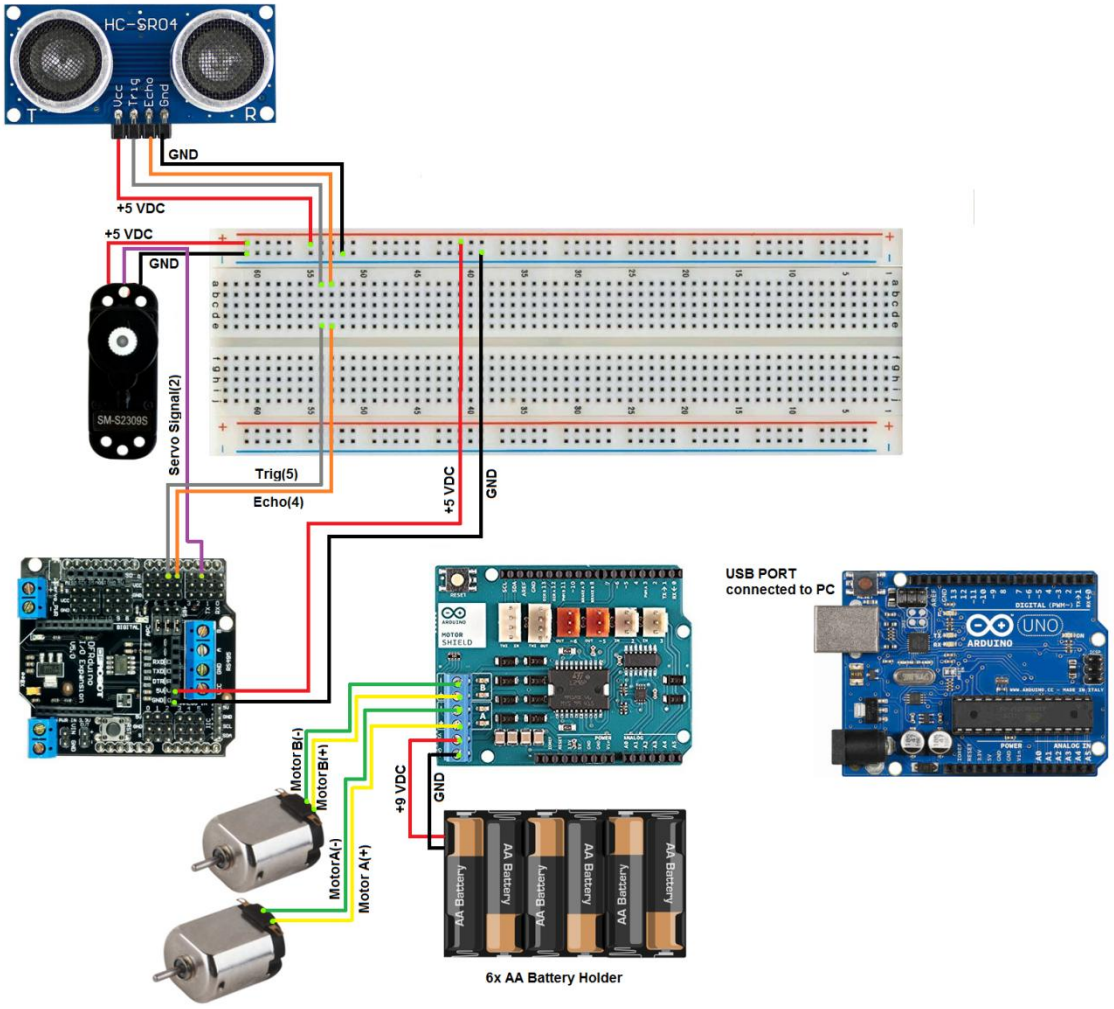

Figure 9. Robot design 2- Wiring diagram (not to scale)

2.2. Software

The Arduino Integrated Development Environment (IDE) or Arduino software is the main tool used to develop this project. The IDE includes an editor window for writing and edit the Arduino sketch. This sketch allows to get all the inputs from sensors and to define actions to be performed by the output sensors or devices. The program code is written in a simplified version of C++, which seems quite intuitive because it is mainly based on specific commands.

Additionally, the Arduino IDE has the verification and upload abilities. The first concerns the sketch verification, which means that the software errors are detected, if such exist. Then, it compiles the sketch into machine language to be loaded in the Arduino board. The second ability, upload, lets the communication between the laptop and the Arduino board over USB load the compiled program into the Arduino board memory.

The serial monitor window is also included in the Arduino IDE. This ensures the communication between the user and the running program on the Arduino board. For

that reason, it is a perfect tool for debugging the software and an easy way to get the test results.

Finally, this Arduino environment also includes few standard menus to get easy access to examples, libraries...

In this project, MATLAB R2016b and Microsoft EXCEL 2016 software's are used to help Arduino in some mathematical computations, for example, the kinematic modelling evaluation (see section 3.1).

THIS PAGE INTENTIONALLY LEFT BLANK

3. ROBOT MODELING

3.1. Introduction

System modelling is a key process where a real system problem is transformed into an equivalent mathematical problem, which helps us to interpret system functionalities and capabilities, and to predict how it will behave.

There are different types of models depending on how well-known the system is and also about modeling goals. This project is focus on building an empirical model of the Rover 5 platform (see section 2.1.1) to forecast and describe trends of the platform. This model is also called "Black-Box", based on the fact that the real internal working of the system is not known. Consequently, the core of the model is an unknown mathematical function gathered from measured data.

The robot dynamics modeling is considered outside the scope of this project; therefore, the empirical model of Rover 5 is mainly focused on the robot kinematics.

3.2. Kinematic model

Considering the system as a discrete-time control system with no large velocity changes, the kinematic model analysis is simplified by decoupling linear and angular motion (see A.01 Table 3).

Firstly, the DC motors outputs are evaluated for several power inputs within the range +6.5 V_{DC} to + 9 V_{DC}, assuming these are the maximum/minimum recommended voltages of the system operation. To begin with, the robot turns on itself approximately 360 degrees when it is powered with a specific voltage. Thus, the time is recorded per each power input selected, and the angular speed is calculated afterwards. Figure 10 shows angular motion response depending on power input for both DC motors.

Same procedure is applied for the linear motion evaluation, but in this case the robot moves linearly a length of 4 meters per each power input (see Figure 11).

The DC motors speed is controlled by the Pulse Width Modulation (PWM). The more voltage applied, the faster the motor will rotate and the wider the pulse will be (i.e.: the maximum voltage, +9 V_{DC}, corresponds to the maximum PWM (255), and maximum platform linear/angular speed).

Despite the fact that both DC motors shall have identical performances, a trivial dissimilarity exists between the two DC motors. This is probably due to some internal platform malfunction. As shown in Figure 10, motor A has lower angular speed than motor B for same power input. This difference is worst when high operating voltages apply. For simplicity, it is considered both DC motors have identical response making this difference easily corrected considering test results and reducing PWM from motor

B (see A.02 Table 3). In addition, the robot angular motion shows linear behaviour between +6.8 V_{DC} and 7.5 $V_{DC,}$ so it would be useful to keep the operation voltage within this range (see A.03 Table 3).

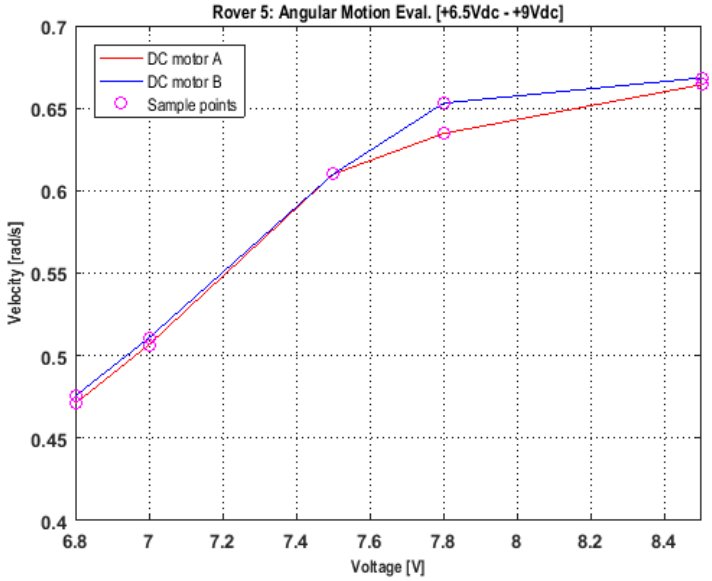

Figure 10. Rover 5: Angular motion vs. Power input

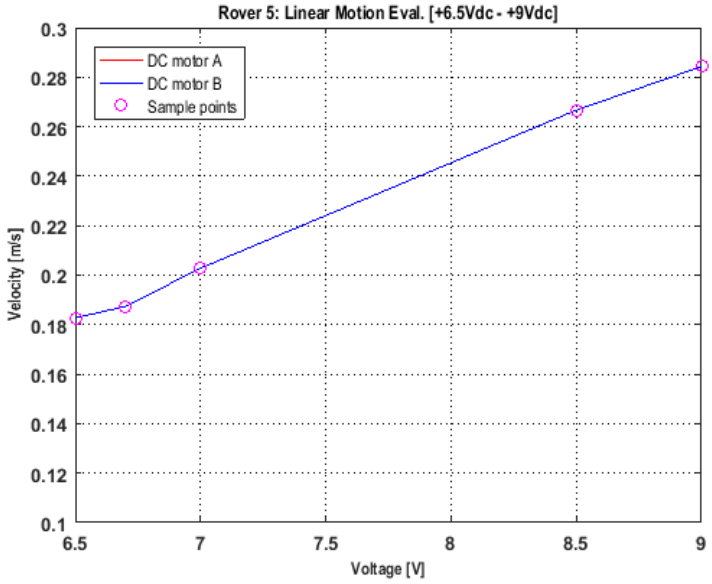

Figure 11. Rover 5: Linear motion evaluation vs. Power input

Secondly, several tests are done extracting the basic parameters to build a robot kinematic model for a specific operating voltage. In this case, the operating voltage selected is +7 V_{DC}, which means that the PWM is diminished to 198. The reason for this is the linear performance of DC motors for this input voltage, as is shown in Figure 10 and Figure 11 (see A.03 Table 3).

Several linear speed measurements are collected during 20 seconds considering +7 V_{DC} power input. These are taken more frequently during the first period where the system is more sensitive to voltages fluctuations. In the same manner, angular speed measurements are gathered during 12 seconds.

From the results obtained, it is performed a basic fitting analysis per each platform motion (see Figure 12 and Figure 13, respectively).

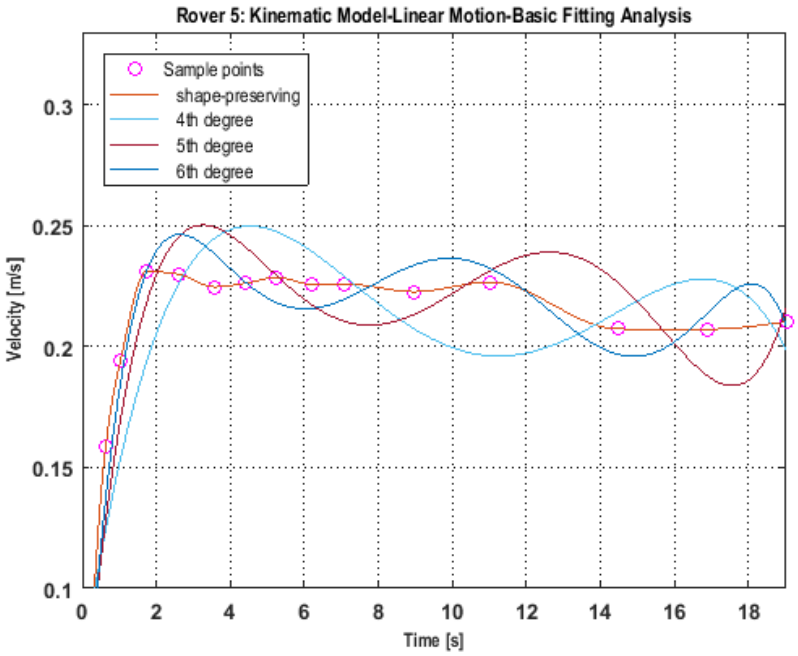

Figure 12. Rover 5: Linear motion- Basic Fitting analysis

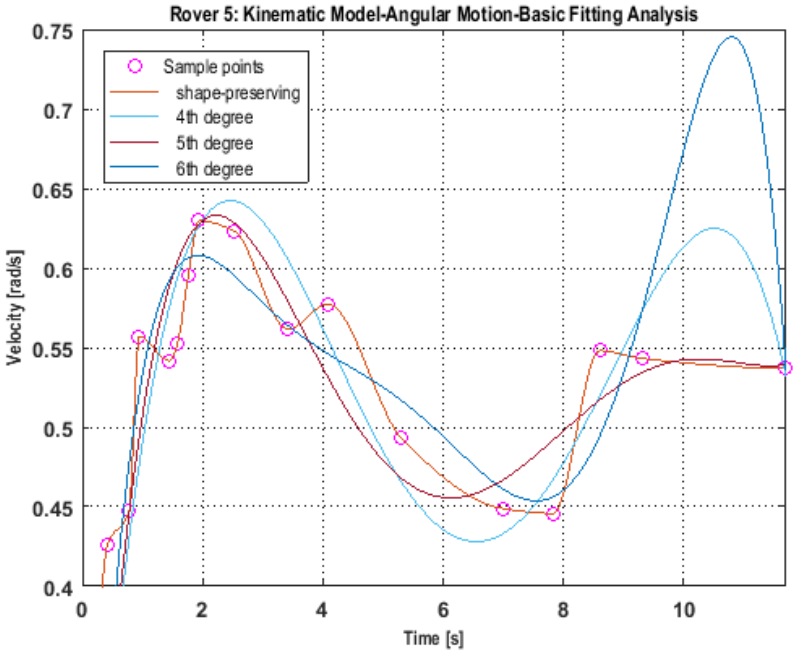

Figure 13. Rover 5: Angular motion- Basic Fitting analysis

Fromt the basic fitting analysis it is concluded that there is no relevant distinctness between 5th degree polynomial and 6th degree polynomial regarding the platform linear motion. However, this difference is more noticeable for angular motion. Besides, the shape-preserving piecewise cubic interpolation is probably the option that fits best in both cases.

In order to clear up the analysis, a 5th degree polynomial interpolation is selected for both, the linear motion (see Ec.02) and the angular motion (see Ec.03). The reason for this is that there is no attempt to large movements when it comes to an indoor robot with the mentioned performances (see section 2.1.5); therefore, the kinematic model derived by using a 5th degree polynomial could be useful enough.

Henceforward, Ec.02 and Ec.03 are used to determine robot position taking into account the time robot used whereas it has been moving.

$$v_{t_i} = 2.780e^{-7} \cdot t_i^5 - 1.99e^{-5} \cdot t_i^4 + 5.29e^{-4} \cdot t_i^3 - 6.30e^{-3} \cdot t_i^2 + 0.031 \cdot t + 0.179 \qquad \text{Ec.02}$$

$$\omega_{t_i} = 7.970e^{-5} \cdot t_i^5 - 2.996e^{-3} \cdot t_i^4 + 4.15e^{-2} \cdot t_i^3 - 0.254 \cdot t_i^2 + 0.638 \cdot t + 0.085 \qquad \text{Ec.03}$$

Finally, a couple of experiments are performed to verify how the kinematic model matches the real robot movement. First of all, it is compared the real and the estimated linear velocity for a while. The mobile robot performs a linear displacement of 4 meters during this time. Then, it is calculated the absolute error on each case. Figure 14 (Left) shows the kinematic absolute error for mobile robot linear motion. After, same analysis is done for the mobile robot angular motion (see Figure 14 (Right)).

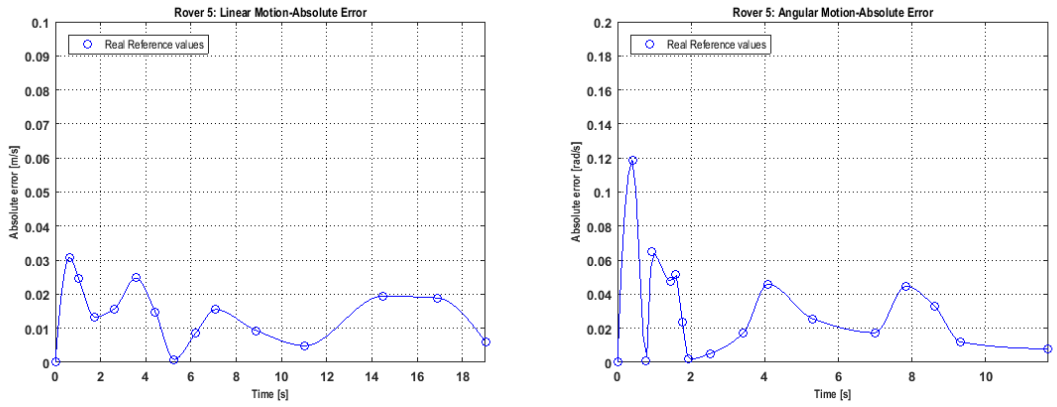

Figure 14. Rover 5: Kinematic model Abs. Error- Linear (Left) and Angular (Right) motion

According to Figure 14, the absolute error of the linear and angular velocity becomes more stable over time, presumably explained by the initial power input fluctuation as a result of battery wearing. The estimated linear speed mean error is 0.0138 m/s; the angular speed mean error is about 0.0305 rad/s. Angular motion estimation is more erratic, which is probably explained by a combination of the estimated kinematic model

for angular motion, as well as, the accuracy of the measurements considered as real reference.

To conclude, the mobile robot follows a pre-defined path combining counter wise (CW) and counter clockwise (CCW) pivoting turns, as well as linear motions. The kinematic model results are compared with the robot path estimated manually (see Figure 15). The robot trajectory duration is approximately 25 seconds and it is divided into eleven segments (see Table 1).

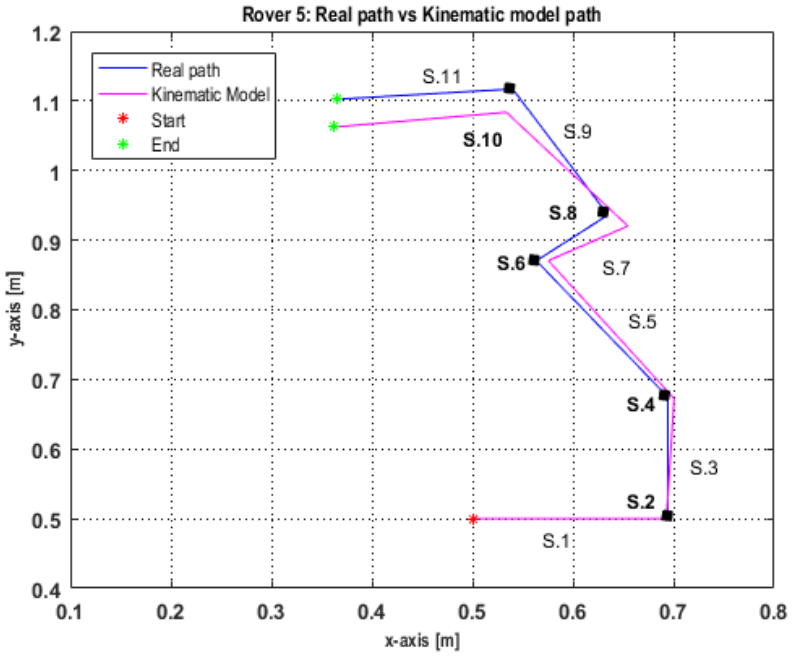

Figure 15. Rover 5: Kinematic model evaluation- Combined motion

Segment	*Type of Motion*	*Time [s]*
S.1	Lineal	0.950
S.2	CCW rotation	2.950
S.3	Lineal	0.900
S.4	CCW rotation	4.000
S.5	Lineal	1.200
S.6	CW rotation	1.400
S.7	Lineal	0.500
S.8	CCW rotation	8.000
S.9	Lineal	1.050
S.10	CCW rotation	2.950
S.11	Lineal	0.900

Table 1. Rover 5: Kinematic model evaluation- Segments description

As shown in Figure 15, the estimated robot path exposes compound errors, which are most notably when the robot turns; in fact, it is demonstrated that the absolute error between the real path and the kinematic model increases every time the robot turns. It

should also be noted that the y-axis error is higher than the x-axis error most part of the time, for example the absolute error at the end of the pre-defined path is 0.003 m on the x-axis, and 0.040 m on the y-axis.

Additionally, it has to be considered that the kinematic model is affected by some external sources such as battery wear or wheel slippage, which cannot be quantified.

Because of the accumulation of errors and the mentioned drawbacks, the kinematic model computed cannot be applied in long-term scenarios but it is assumed well-enough for this purpose. Consequently, the mobile robot linear displacement is limited to 2 m and the angular motion is restricted to 360 degrees (see A.05 Table 3). Notwithstanding, some robot position corrections could be applied by controlling the DC motors or correcting manually robot position during the test.

THIS PAGE INTENTIONALLY LEFT BLANK

4. METHODS FOR MOBILE TARGET POSITIONING

According to *Borenstein* [1], the existing techniques for determining indoor or outdoor target localization can be classified into two main categories: relative (local) and absolute (global). The main difference between them is that local positioning requires knowledge about previous target position to estimate the current position. Consequently, positioning errors can be easily mitigated by global techniques but not local.

While the numerous solutions for target positioning and technologies advances rapidly, it is really common to merge relative and absolute methods by using a Kalman filter, in order to get high accurate target position.

4.1. Relative Positioning

Relative positioning or dead-reckoning is the most common and ancient positioning technique. It mainly consists on applying a mathematical process to determine the current target location based on the information provided by on-board sensors over an elapsed time interval. The target previous position need to be known.

In general, relative positioning techniques inevitably introduce systematic and non-systematic errors, which are accumulated over time and entail large inaccuracies. For that reason, the relative positioning methods are generally inappropriate when the application time is longer than few minutes or the solution demands high accuracy.

Primarily, a couple of sub-methods are derived from relative positioning category: odometry and inertial navigation.

Odometry uses encoders to measure wheel revolution and steering orientation, usually coupled to the wheel axles or motor protection cover. It is a simple and inexpensive positioning method, which provides a short-term accurate solution. However, its major weakness is the unbounded accumulation of errors as a consequence of wheel base, imprecision in wheel circumference or wheel slippage.

Inertial navigation provides a self-contained navigation without any external reference, just being required processing sensors data with known initial position, velocity and course. Nowadays, the most basic implementation of the inertial navigation is a combination of accelerometers, which measure the mobile robot acceleration, and gyroscopes, that measure the angular rate. Integrating the accelerometers output twice makes possible to obtain the target position, whereas it is needed making the same just once with the gyroscope output to get target orientation. There are many types of accelerometers and gyroscopes with different performances and capabilities depending on sensor technology used [4]. However, these sensors are also submitted to large errors, becoming inadequate for long-term applications and really useful to provide short-time positioning (i.e.: course changes when robot is manoeuvring). Furthermore,

the way they are mounted on the platform contributes significantly to the output accuracy.

Unfortunately, the Rover 5 DC motors have no coupled encoders, hence, odometry positioning is ruled out. Accordingly, the relative positioning is computed using simple and low-cost inertial sensors compatible with Arduino UNO board (see section 2.1.2).

4.1.1. Arduino Inertial Sensors

This section includes the evaluation of a coupled of Arduino inertial sensors. On the one hand, it is assessed the triple axis MMA7361 accelerometer manufactured by Freescale. On the other hand, it is analyzed the dual axis T000062 gyroscope from Tinkerkit. Both inertial sensors are shown in Figure 16.

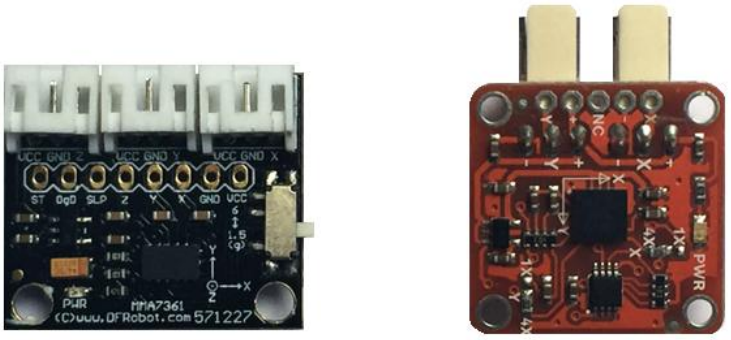

Figure 16. Arduino inertial sensors: MMA7361 (Left); T000062 (Right)

4.1.1.1. Accelerometer

The MMA7361 accelerometer is a small and lightweight accelerometer breakout, which has 3 JSL connectors to be easily plugged into the analog input pins of the Arduino I/O expansion board by using jumper wires. This accelerometer provides analog outputs on each axis (X, Y and Z) and it includes an internal low-pass filter. Moreover, it has an operating voltage range between $+3.3$ and $+8$ V_{DC}, which is set to 5 V_{DC} in this analysis.

This MEMS accelerometer has selectable sensitivity, (800 mV/g; 206 mV/g), depending on the measurement g-range selection. The working g-range can be easily modified using the switch included in the breakout board. A full-scale range implies more uncertain output, which means less accurate reading out. For that reason, it is recommended to apply low-scale range, if possible. Given the reasoning above, the g-range selected is ± 1.5 g, which corresponds with a high sensitivity, 800 mV/g.

One of the most common accelerometer errors is the 0g-offset or the difference between measured value and the true zero value. This error is also called bias error and it needs to be compensated before the sensor is used. When the accelerometer is steady, it is only earth gravity affecting it. It means that all axes shall measure 0-g except the one pointing to the direction of the earth gravity field, which shall measure 1 g.

Accelerometers are usually trimmed for 0g-offset in the factory, but in this case, it is computed and corrected every time the MMA7361 accelerometer is switched on. The way of doing this calibration is to keep the sensor in steady state position. Then, some output measurements are recorded. Finally, it is computed the outputs mean, which is the value subtracted from each analog output in order to get X=0 g, Y=0 g and Z=1 g.

Once the accelerometer output is corrected, the voltage is converted into g-forces taking into account the accelerometer sensitivity. To conclude, g-forces are transformed into common acceleration units, m/s^2, and a numerical integration method is applied twice to get the target position. As an example, different numerical integration methods - Trapezium, Midpoint and Simpson's rule- are applied to evaluate the accuracy of each one. After this assessment, it is concluded that in this case there is a trivial difference between the above mentioned numerical integration methods, this being about the order of 10^{-14} between the Trapezium and the Midpoint rule results, and around 10^{-17} between the Trapezium and the Simpson's rule results. For that reason, the simplest numerical integration method, Trapezium rule, is applied in this case.

Finally, a simple test is performed in order to compare the MMA7361 accelerometer results with the kinematic model (see section 3.1) and the real robot motion. To that end, the robot moves linearly during 8 seconds and its position is manually recorded every second.

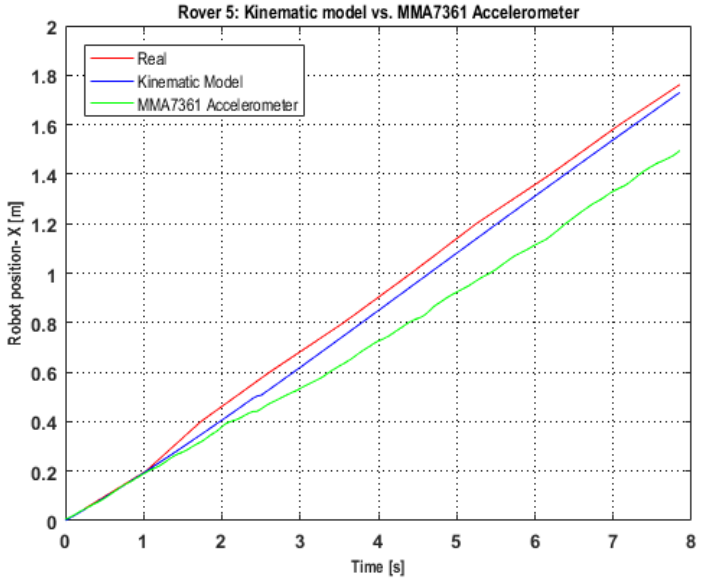

Figure 17. Rover 5: Kinematic model vs. MMA7361 Accelerometer

As can be seen in Figure 17, the accelerometer results are close to the model during first 2.5 seconds. After that, the position absolute error grows uncontrolled over time, this being higher than the absolute error obtained with the kinematic model.

4.1.1.2. Gyroscope

The T000062 gyroscope is a MEMS sensor that can measure angular rate along X-axis and Y-axis. However, in this case the interesting angular measurement is along Z-axis,

which means that the gyroscope needs to be mounted vertically. Both gyroscope analog outputs are connected to two different analog inputs of the Arduino I/O expansion board using jumper wires. The analog outputs of the gyroscope are positive when the mobile robot turns CCW.

The T000062 is a tiny and lightweight inertial sensor with an operating range between -0.3 V_{DC} and 6 V_{DC}. The working voltage sets to 5 V_{DC}. It has a full scale of $\pm 1500°$ and a sensitivity of 0.67 mV/°/s.

Again, the zero offset is a critical parameter and it is estimated in the same way it is computed for the accelerometer MMA7361 (see section 4.1.1.1). This 0-offset is calculated every time that the gyroscope is switched on, and subtracted from each gyroscope analog output. Finally, the angular velocity is calculated taking into account the sensor sensitivity and the robot orientation is estimated by numerical integration.

A simple test is performed in order to compare the real robot path, the kinematic model and the gyroscope results. To this end, the robot carries out a pivot turn and its position is manually recorded every second. From Figure 18 it is noted that the gyroscope results are closer to real data than the kinematic model during first 2.5 seconds. After this time, the gyroscope error increases in an uncontrolled manner, for instance the difference between the gyroscope and the real data is 37.655° at $t = 8$ s.

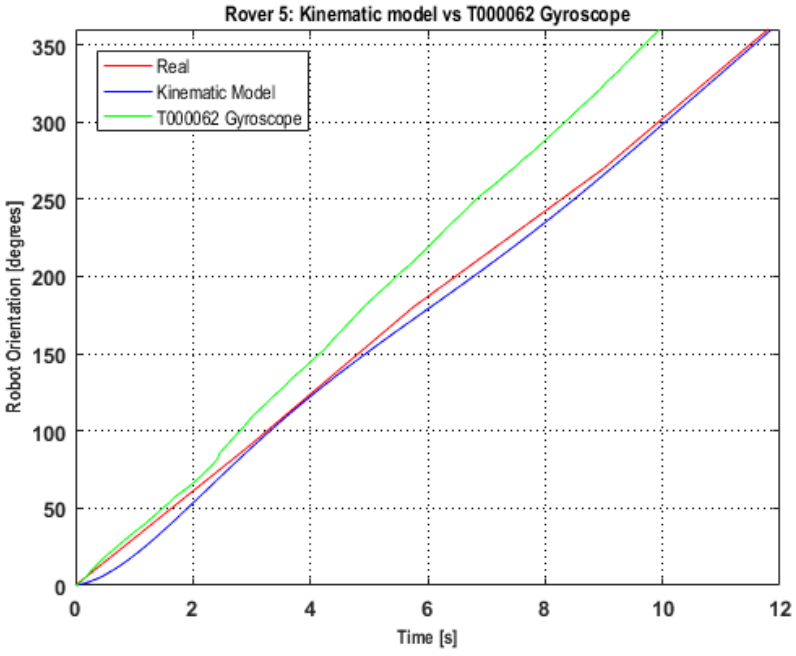

Figure 18. Rover 5: Kinematic model vs. T000062 gyroscope

To conclude, the mobile robot implements an arbitrary trajectory combining linear and angular motions during 18 seconds. The robot path is divided into nine segments (see Table 2). The results obtained are illustrated in Figure 19.

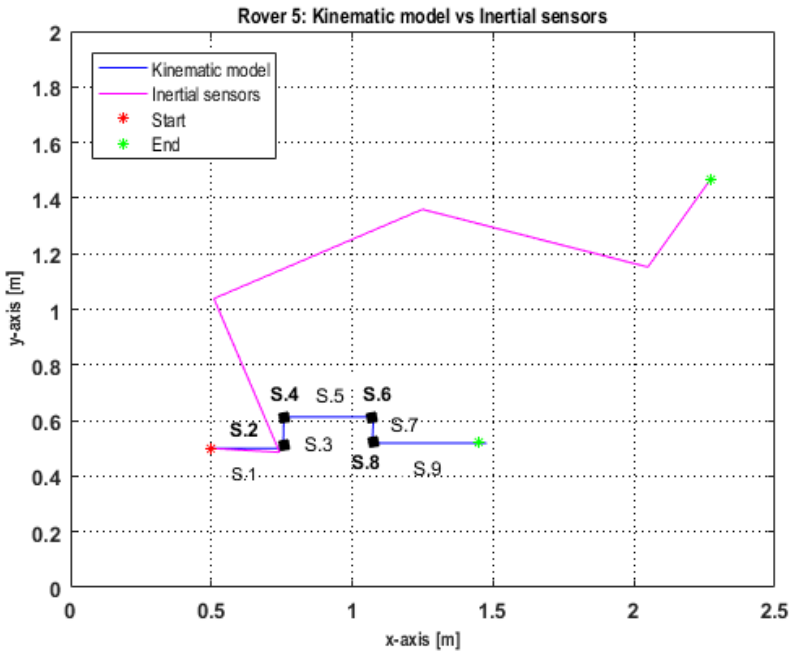

Figure 19. Kinematic model vs. inertial sensors

Segment	*Type of Motion*	*Time [s]*
S.1	Lineal	1.300
S.2	CCW rotation	3.000
S.3	Lineal	0.600
S.4	CW rotation	3.000
S.5	Lineal	1.600
S.6	CW rotation	3.000
S.7	Lineal	0.500
S.8	CCW rotation	3.000
S.9	Lineal	2.000

Table 2. Kinematic model vs. inertial sensors- Segments description

As seen in Figure 19, the inertial sensors error is so great that after few seconds the estimated mobile robot position is totally wrong, which is probably due to the inertial sensors error grow over time. Consequently, the inertial sensors are ruled out as a relative positioning method. Henceforward, the kinematic model is considered the real mobile robot reference.

4.2. Absolute Positioning

Absolute positioning – also known as reference-based positioning- arises from the need to prevent inaccurate position estimations because of cumulative errors. These positioning techniques rely on landmark navigation, active or passive beacons, electronic compasses, map matching or GNSS-based positioning, which is the case of

the Global Positioning System, one of the most accurate absolute positioning techniques.

4.2.1. Active or Passive Beacons

Absolute positioning based on active or passive beacons is one the most common technique in those environments where GNNS-based positioning cannot be applied, i.e., urban, underwater or indoor environments. This is a non-expensive solution and it offers high reliability results. Consequently, it is the absolute positioning technique used in this project.

Several passive beacons are ad-hoc designed and built. These consist of cardboard cylinders with smooth surface. The reason using cardboard is due to the perfect absorption of sound waves of this material (see section 2.1.5) [30]. Each beacon has unvarying diameter, 0.083 m, and high, 0.3 m. The cardboard thickness is around 0.003 m (see Figure 20).

Figure 20. Ad-hoc Beacons

Although there are multiple mathematical techniques used in active/passive beacon navigation [7], only a couple of them are implemented in this project: trilateration and triangulation. It is assumed only bi-dimensional scenarios.

4.2.1.1. Trilateration Technique

Trilateration is a mathematical technique that estimates the target location in a 2D scenario considering the intersection between three circles. Each of them has its centre in a known position and has a radius equal to the distance between the beacon and the target. Consequently, it is required to know at least three beacon positions and three distance measurements in order to make the 2D trilateration works properly.

There are several closely related ranging techniques to be used in trilateration. Some of the most common are Time of Arrival (TOA), Time Differences of Arrival (TDOA), or a combination thereof [9].

4.2.1.1.1. TOA/ TDOA

TOA and TDOA are high accurate ranging techniques based on signal travelling time, so the time sensed is converted into distance. The simplest way to do this is to multiply the time by the speed of propagation. In this project, it is assumed equal to the speed of sound (see section 2.1.3.1).

According to literature, the TOA is the instant in time when a signal arrives to the receiver. Then, if the speed of propagation in the medium and the time at which the signal is emitted are known, it may be possible to compute the total distance between the target and the beacon. Thus, one major disadvantage of TOA ranging technique is the time synchronization between the transmitter and the receiver.

Contrary, the TDOA uses the difference in time at which the signal sent by the transmitter arrives to multiple receivers. From this Δt, it is possible to compute the distance between receivers and deduce the target position. In this case, the time synchronization is trivial but the multi-path effects and the Line Of Sight (LOS) availability are even more critical points that need to be considered. Furthermore, TDOA technique requires one transmitter and at least two receivers to be applied in a bi-dimensional scenario.

For simplicity, it is applied the TOA ranging technique due to the ultrasonic SR-HC04 sensor has incorporated transmitter and receiver at the same module (see section 2.1.3.1).

4.2.1.2. Triangulation Technique

Triangulation computes the target pose (position and orientation) using the angular measurement from a reference point. In this case, it is considered at least a couple of angle direction lines formed by a radial line starting in a known position. The intersection of these two lines represents the target location and it can be obtained by doing some geometric computations. Unlike trilateration, only two positions and a couple of angle measurements must be known in order to estimate the target position by 2D triangulation. However, the accuracy increases if it is used more known positions.

The Angle Of Arrival (AOA) is the common ranging technique used in triangulation.

4.2.1.2.1. AOA

The AOA ranging technique is based on determining the angle of incidence of the signal when it arrives to the receiver. Then, different geometric computations can be done to estimate the target position. Some of the advantages of AOA are that the time synchronization is insignificant and it only needs two measurements even for 3D scenarios. In this case, the angle of arrival is related to the angle that the servomotor has moved when the beacon is detected (see section 2.1.4). Moreover, the reference point considered is the horizontal axis.

THIS PAGE INTENTIONALLY LEFT BLANK

5. MATHEMATICAL APPROACHES AND PROBLEM FORMULATION

The main problem of the mathematical techniques used to infer target position is that they are based on physical measurements, which are usually affected by non-systematic errors, i.e.: measurements noise, or even systematic errors. To successfully deal with this problem, an approximate solution that fits the equations as much as possible needs to be computed. Different mathematical approaches have been investigated and developed over time.

In this section, it is described the mathematical approaches used to solve the bi-dimensional single target positioning problem in this project.

5.1. Trilateration Approach- Least Squares Approach

Least Squares Estimator (LSE) and Maximum Likelihood Estimators (MLS) are two common approaches used to solve the range-only target positioning. To put it simply, the LSE goal is to minimize the squared errors between the predicted and real measurements by finding a solution that fits best the data obtained, whereas the MLE purpose is to estimate the parameters of a statistical model for which the measurement performed is most likely.

The MLE uses iterative methods such as Gradient or Newton-Raphson to get a convergence solution; therefore, it is required iterative operations including large matrices, which leads to high computational complexity.

Because of the basic technologies used in this project, the simplest way forward is to use the Least Squares Estimator, even though the MLE benefits surpass the advantages of the LSE, providing more accurate results with optimal properties such as smaller variances or approximate normal distributions.

5.1.1. Problem Formulation

A couple of cases are set-up to solve the single target positioning problem by trilateration technique. It should be emphasized that the same Least Square mathematical approach is used in both cases, so the main difference lies in how the problem is formulated and the number of beacons and sensors used.

So far, the trilateration problem has been raised as a technique requiring a minimum of three beacons and three distance measurements to work properly. However, it is intended to demonstrate that it is possible to diminish the quantity of beacons, as well as, the number of sensors. To that end, the robot has to move through pre-defined points performing distance measurements.

5.1.1.1. Single target positioning with multiple beacons

In this case, it is assumed a single robot with unknown location, and stationary beacons whose position are known with respect to a two dimensions ($m = 2$) inertial reference frame, $\{I\}$. This frame of reference is marked with the axes $\{x\}$ and $\{y\}$, and its origin, O, is located somewhere in an arbitrary point.

In this case, the robot position in $\{I\}$ is denoted by $\boldsymbol{p} = [x, y]$, while each beacon location is represented by $\boldsymbol{pb_i} = [x_i, y_i]$; $i = 1,2,3 \dots n$. The number of beacons is n.

Finally, the distance between the i^{th} beacon and the target is determined by $d_i = r_i^2 = \|\boldsymbol{p} - \boldsymbol{pb_i}\|^2$, where $\|\cdot\|$ indicates the norm. Figure 21 illustrates the first problem set-up and the main variables.

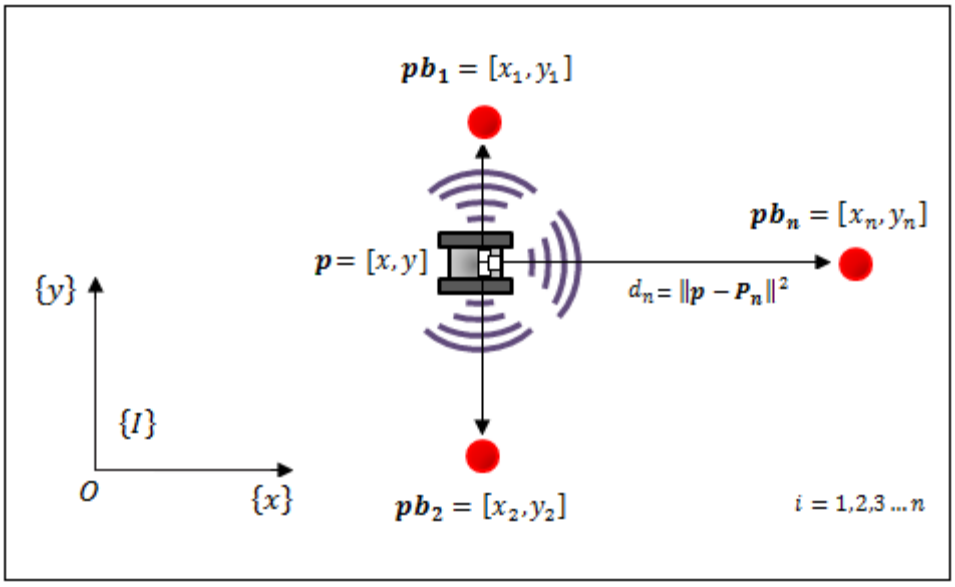

Figure 21. Problem set-up- 2D single target positioning using trilateration and multiple beacons

The kinematic model of the robot is described in section 3.1.

5.1.1.2. Single target positioning with a unique beacon

Again, it is considered a single robot located somewhere in the bi-dimensional frame of reference, $\{I\}$, detailed in section 5.1.1.1. In this case, the robot performs three range measurements in three different unknown positions, and there is only one stationary beacon which position is known with respect to $\{I\}$.

Then, the first robot position is denoted by $\boldsymbol{p_0} = [x_0, y_0]$, and the following positions by $\boldsymbol{p_i} = [x_{i-1} + \Delta x_i, y_{i-1} + \Delta y_i]$; $i = 1,2,3 \dots k$, where k is the total number of robot positions considered, and Δx_i and Δy_i denote the robot linear displacement between each position, which are dependent on robot orientation. The beacon location is represented by $\boldsymbol{pb} = [x, y]$.

The distance between a specific robot position and the only beacon is given by $d_i = r_i^2 = \|\boldsymbol{p_i} - \boldsymbol{pb}\|^2$, where $\|\cdot\|$ indicates the norm. The kinematic model of the robot is described in section 3.1.

Figure 22 shows the second problem set-up and the main variables.

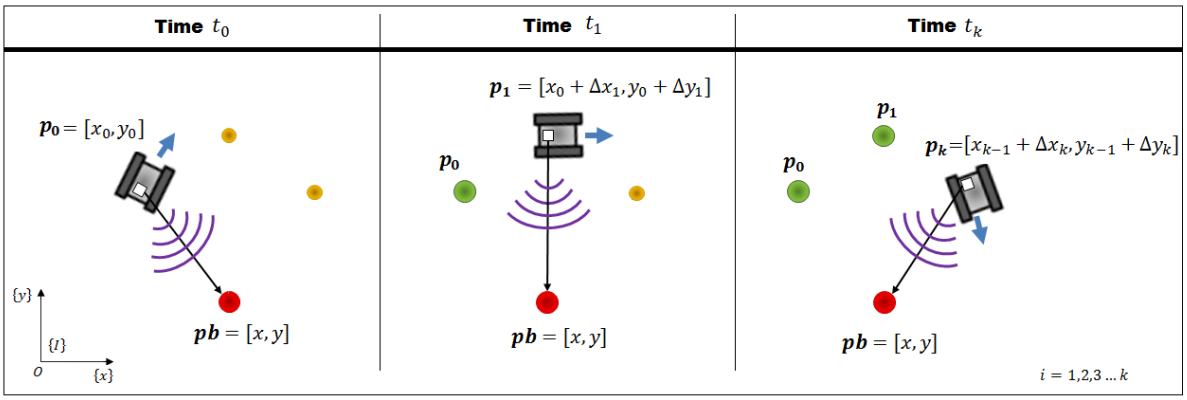

Figure 22. Problem set-up- 2D single target positioning using trilateration and one beacon

5.1.2. Ordinary Least Square (O-LS) or Linear Least Square (L-LS) Algorithm

For simplicity, the Ordinary Least Square (O-LS) or Linear Least Square (L-LS) algorithm is the one used to solve the over determined linear systems in this project. Henceforward, it is assumed the reader is familiar with Least Squares solutions; see for instance [15] and [16].

Following algorithm description assumes at least three known beacon positions and one robot location. However, same algorithm can be applied replacing beacon positions by robot scanning points and robot location by the beacon known position, in order to solve the mobile robot positioning problem with a unique beacon.

The vector of square range measurements is defined by

$$\bar{\boldsymbol{d}} = \boldsymbol{d} + \boldsymbol{\varepsilon} \ \in \mathbb{R}^{\mathrm{m}} \qquad \text{Ec.04}$$

where $\boldsymbol{\varepsilon}$ is a vector containing the measurements errors. If there is no measurement error, each entry of the square range vector can be written as $d_i = r_i^2$.

Then, considering the square range measurements, re-organizing some terms in Ec.04, and taking into account that it is looking for an approximate solution where the norm of the residual vector, $\|\boldsymbol{r}\|$, is minimized [14], the following linear equation can be derived

$$A \cdot \boldsymbol{p} = \boldsymbol{b} + \boldsymbol{\varepsilon} \qquad \text{Ec.05}$$

where $A \epsilon \mathbb{R}^{mxn}$ is a known matrix based on beacon known locations given by $\boldsymbol{pb_i} = [x_i, y_i]$; $i = 1,2,3 \ldots n$, $\boldsymbol{b} \epsilon \mathbb{R}^n$ is a column vector with the distance measurements and $\boldsymbol{p} \epsilon \mathbb{R}^{\mathrm{m}}$ is a vector containing the unknown robot position.

A, **b**, and $\boldsymbol{p}$ are given by Ec.06, Ec.07 and Ec.08, respectively.

$$A = \begin{bmatrix} 2\boldsymbol{P}_1^T & -1 \\ \vdots & \vdots \\ 2\boldsymbol{P}_n^T & -1 \end{bmatrix} \quad \text{Ec.06}$$

$$\boldsymbol{b} = \begin{bmatrix} \|2\boldsymbol{P_1}\|^2 - \bar{\boldsymbol{d}}_{\boldsymbol{1}} \\ \vdots \\ \|2\boldsymbol{P_n}\|^2 - \bar{\boldsymbol{d}}_{\boldsymbol{n}} \end{bmatrix} \quad \text{Ec.07}$$

$$\boldsymbol{p} = \begin{bmatrix} x \\ y \end{bmatrix} \quad \text{Ec.08}$$

The measurement errors are not zero mean error nor normally distributed. However, according to [10], they can be assumed as Gaussian error under a couple of realistic conditions. These conditions are related to the robot position, which should not be near to any of the beacons, and the distances magnitude, r_i, which needs to be much bigger than the standard deviation of the measurement errors, $r_i >> \sigma_i$.

For project analytical adaptability, it is assumed that the above conditions apply. Then, the measurement errors are supposed uncorrelated with the independent variables, having zero mean error, same variance and normally distribution (see A.08 Table 3). However, it is important not to forget that this assumption is non-realistic since the noise is distance dependent and, when the target is close to any beacon, it is required a special analysis.

Finally, the robot position can be estimated by Linear Least Square

$$\widehat{\boldsymbol{p}}_{\boldsymbol{L-LS}} = (A^T A)^{-1} A^T \cdot \boldsymbol{b} \quad \text{Ec.09}$$

where A^T is the transponse of matrix A. The Ec.09 can be simplified to Ec.10 when $n = 3$.

$$\widehat{\boldsymbol{p}}_{\boldsymbol{L-LS}} = (A)^{-1} \boldsymbol{b} \quad \text{Ec.10}$$

This mathematical development is entirely detailed and proved in [14].

5.1.2.1. Error Characterization of the estimated solution

One way of characterize the position estimation error related to the L-LS solution is to compute the mean error and covariance.

Based upon mathematical reasoning in [10], the covariance matrix, $\Sigma_{L-LS} \epsilon \mathbb{R}^{mxm}$, can be calculated using Ec.11, where I is the identity matrix.

$$\Sigma_{L-LS} = (A^T \cdot I \cdot A)^{-1} \quad \text{Ec.11}$$

The main diagonal entries of the covariance matrix are the variance of each component of the vector $\boldsymbol{p_{L-LS}}$, while the entries outside the main diagonal indicate the cross-correlation or estimations independency, i.e. the covariance between each component of the vector $\boldsymbol{p_{L-LS}}$. When it comes to an ideal case, the entries outside the main diagonal are zero, indicating that the measurements are performed in optimal positions and the information available is maximized. Consequently, the entries outside the main diagonal values need to be low in order to be close to an ideal case.

Additionally, the covariance matrix is always symmetric and it has positive determinant, $|\Sigma_{L-LS}| > 0$.

On the other hand, the mean estimate error, μ_{L-LS}, can be computed with Ec.12, where Σ_{L-LS} is the covariance matrix, I is the identity matrix and $\boldsymbol{r}$ is a column vector containing the main diagonal entries of the covariance matrix, Σ_{L-LS}. The vector $\boldsymbol{r}$ is calculated using the operator *diag*, $\delta(\Sigma_{L-LS})$. The mathematical development behind Ec.12 can be found in [10].

$$\mu_{L-LS} = \Sigma_{L-LS} \cdot A^T \cdot I \cdot \boldsymbol{r} \quad \text{Ec.12}$$

The mean estimate error is a vector of dimensions, $m = 2$, referring to x coordinate error and y coordinate error. The mean estimate error of each coordinate also has to be close to zero because it is assumed a zero mean Gaussian error (see A.08 Table 3).

5.1.2.2. Fisher Information and Cramer-Rao for Range-localization

According to the above described target positioning problem, the quality of the Linear Least Square solution can be generally measured by analyzing the Fisher Information Matrix (FIM) or the Cramer- Rao Lower Bound (CRLB).

As stated in [19], the CRLB establishes that under some regularity conditions, the variance reached by any unbiased estimator is at least as high as the inverse of the Fisher Information Matrix. This idea is represented in Ec.13, where Σ_{L-LS} is the covariance of the unbiased estimator. This CRLB statement is well-detailed and proved in [19].

$$\Sigma_{L-LS} \geq FIM(\boldsymbol{p})^{-1} = CRB(\boldsymbol{p}) \quad \text{Ec.13}$$

Given the above statement, any estimator which achieves CRLB limit, becomes a minimum-variance unbiased estimator, therefore, its mean square error is minimized.

The FIM catches the amount of information that the range measurements vector carries about the unknown robot position estimation. According to [18], let $\widehat{\boldsymbol{p}}(\boldsymbol{z})$ be any unbiased estimator of the unknown target position, $\boldsymbol{p}$, and $\boldsymbol{z}$ a set of measurements from where the unbiased estimator is obtained. In addition, let the likelihood function be defined as $\mathcal{L}(\boldsymbol{p}|\boldsymbol{z})$, which indicates the probability of obtaining the observation $\boldsymbol{z}$ given the target position, $\boldsymbol{p}$. Then, the FIM can be defined by Ec.14, where $E\{\cdot\}$ denotes the average operator and $\nabla_{\mathbf{p}} \log \mathcal{L}(\boldsymbol{p}|\boldsymbol{z})$ is the gradient of the natural logarithm of the likelihood function, also known as log-likelihood, with respect to the unknown robot position $\boldsymbol{p}$.

$$FIM(\boldsymbol{p}) = \mathcal{J}(\boldsymbol{p}) = E\left\{(\nabla_{\mathbf{p}} \log \mathcal{L}(\boldsymbol{p}|\boldsymbol{z}))\,(\nabla_{\mathbf{p}} \log \mathcal{L}(\boldsymbol{p}|\boldsymbol{z}))^T\right\} \qquad \text{Ec.14}$$

The complete mathematical explanation for Ec.14 is given in [18]. Additionally, the FIM mathematical proof for range-difference based localization in $\mathbb{R}^2$ [20] and for bearing-localization [21] are used for a better understanding of the mathematics developments behind Ec.14.

Accordingly, the FIM determinant can be calculated using Ec.15, where $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ are vectors in $\mathbb{R}^t$, considering t the number of sensors involved in the target positioning, and θ is the angle formed by these two vectors, $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$. The mathematical development to get Ec.15 is well detailed in [18].

$$|FIM| = \frac{1}{\sigma^4}\,|\boldsymbol{\mathcal{X}}|^2\,|\boldsymbol{\mathcal{Y}}|^2(1 - cos^2(\theta)) \qquad \text{Ec.15}$$

This FIM determinant can be used as an indicator of the performances of a given sensor configuration.

5.1.2.3. Optimal FIM determinant

Having considered the FIM and its determinant, it is also reasonable to look at the optimal FIM determinant, which allows us to determine the optimal measurement points, and subsequently to compute robot optimal trajectories.

The optimal measurement points can be got by maximizing the optimal FIM determinant, which implies to find the necessary conditions of optimality. This problem is solved step by step in [18].

Finally, Ec.16 can be used to compute the optimal FIM determinant, where n is the total number of known beacon positions.

$$\left|FIM_{opt}\right| = \frac{n^2}{4\sigma^4} \qquad \text{Ec.16}$$

Thus, this determinant sets an upper limit that can be reached with the n range measurements in a bi-dimensional scenario.

According to [29], the simplest optimal points are the ones defined by the polygon vertices, for instance the triangle polygon has three optimal points at $0°$, $120°$ and $240°$, which angles are formed by the vector connecting the beacon with robot position on each range measurement. However, these optimal points are maximum-energy estimations because they require the robot to move around the beacon. For that reason, it has been investigated a more efficient optimal solution from the robot energy consumption point of view, concluding that the symmetric positions can also be optimal points. These symmetric points are concentrated in a half-plane; therefore, the solution is obtained by taking the symmetrical from those ones from the polygon contained at this half-plane. For example, considering four scanning positions ($N = 4$), the symmetric points are located at $0°$, $45°$, $135°$ and $180°$. Again, each of the previous mentioned angles corresponds to the one formed by the vector connecting the beacon with robot position on each range measurement.

5.2. Triangulation Approach- Geometric Circle Intersection

As maintained by *Cohen and Koss* [22], the three-point triangulation problem can be solved by four different methods: Iterative Methods (e.g.: iterative search, Newton-Raphson method and so on), Geometric Circle Intersection, Geometric Triangulation and Multiple Beacon Triangulation.

Simply put, the first group is based on reaching the target position - if it exists- after multiple iterations, starting from a known point and applying trigonometric relations. The second method is anchored in trigonometric functions using the geometry of the beacons and the angles between the beacons from the target point of view. Regarding to the third method, it is founded on the intersection of two of the three circles passing through the beacons and the target. Finally, the fourth method uses more than three angle measurements, resulting in an over determined problem.

Although each above-mentioned method has its weaknesses, the sensitivity analysis performed in [22] concludes that the method of Geometric Circle Intersection is one of the most reliable and robust. Nevertheless, a major drawback of this method is the large errors gathered when the three beacons and the target lie on the same circle, or close circles.

The Geometric Circle Intersection method has been adapted -sometimes improved- and applied to many target positioning problems; see for example [23], [24], [25], [26], [27] and [28].

In this project, it is carried out the three object triangulation algorithm proposed in [23].

5.2.1. Problem Formulation

Let $\{I\}$ be the reference frame, which origin, O, is located somewhere in an arbitrary point and it is marked with the axes $\{x\}$ and $\{y\}$.

As mentioned previously, it is assumed a single robot performing three angle measurements in three different unknown positions, and one stationary beacon which position is known regarding $\{I\}$. This beacon location is defined by $\boldsymbol{pb} = [x, y]$.

Then, the first robot position is denoted by $\boldsymbol{p_0} = [x_0, y_0]$, and the following positions by $\boldsymbol{p_i} = [x_{i-1} + \Delta x_i, y_{i-1} + \Delta y_i]$; $i = 1,2 \dots k$, where Δx_i and Δy_i denote the robot linear displacement between each position, which are dependent on robot orientation, and k is the number of positions where robot scans.

The angles are measured CCW, considering the horizontal axis as an initial reference ($\phi = 0$). Consequently, the angles between this initial reference and the different robot positions are given by ϕ_i; $i = 1,2,3 \dots k.$, and the angle between two different robot positions, for instance $\boldsymbol{p_i}$ and $\boldsymbol{p_j}$, is defined by ϕ_{ij}.

The kinematic model of the robot is described in section 3.1.

Figure 23 shows the problem set-up and the main variables for the mobile robot positioning problem solved by triangulation.

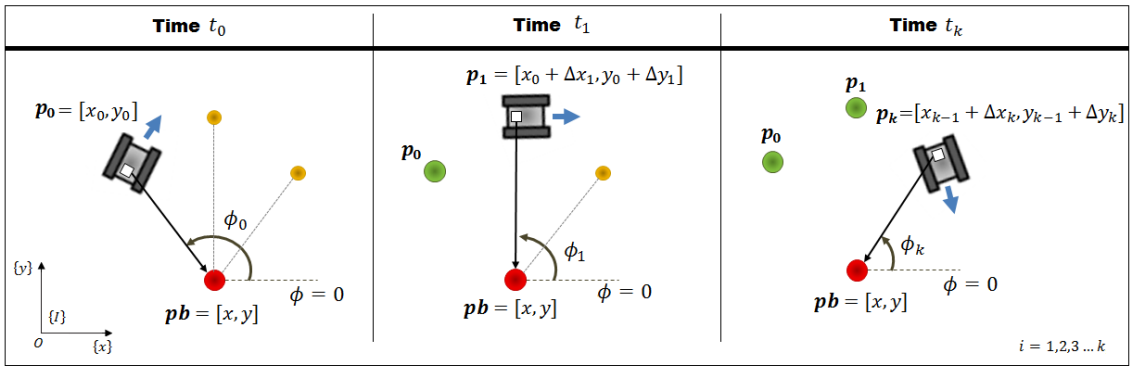

Figure 23. Problem set-up- 2D single target positioning using triangulation and one beacon

5.2.2. Geometric Circle Intersection- *ToTal* algorithm

The *ToTal* algorithm is based on the Geometric Circle Intersection method. However, the main difference between this and other similar algorithms is that *ToTal* uses the three circles parameters (radius, circle centre and power centre), while common methods use two circles parameters to compute the circle intersection. For that reason, *ToTal* algorithm introduces the notion of the power centre, which is the unique point of

equal power respect to the three circles. The *ToTal* algorithm also defines the power line of two circles, which is the line perpendicular to the line joining the circle centres and it concurs in the power centre. Thus, the problem consists of solving the intersection of all power lines, i.e. to solve a linear problem. According to the Geometric Circle Intersection drawbacks, the power centre always exists except when the circle centres are collinear or two of the three circles are equal.

The algorithm assumes that a measured angle can be associated to a given beacon, and the angle measurements from the beacons are taken separately and relatively to some reference angle. For simplicity, it is considered that all angles are measured CCW and the reference angle is the horizontal axis. Furthermore, it is also assumed Gaussian noise with zero mean.

It should be emphasized that following algorithm description assumes three known beacon positions, $n = 3$, (i, j and q) and one robot location. The entire mathematical development can be found in [23]. Nevertheless, same algorithm can be applied replacing beacon positions by robot scanning points and robot location by the beacon known position, in order to solve the mobile robot positioning problem with a unique beacon.

The *ToTal* algorithm divides the three object triangulation problem into two phases.

At the first phase, it is computed the circle parameters (radius, (R_{ij}), and circle centre coordinates, $\boldsymbol{c}(x_{ij}, y_{ij})$) of each of the circles passing through the robot unknown location, $\boldsymbol{p}$, and a couple of known beacon positions, for example, $\boldsymbol{pb_i}$ and $\boldsymbol{pb_j}$. Additionally, multiple simplifications are performed, including a translation ($'$) of the world coordinate frame into one of the known beacon positions.

Thus, the circle centre coordinates can be computed by

$$x'_{ij} = \left(x'_i + x'_j\right) + T_{ij}(y'_i - y'_j) \qquad \text{Ec.17}$$

$$y'_{ij} = \left(y'_i + y'_j\right) - T_{ij}(x'_i - x'_j)$$

where (x'_i, y'_i) and (x'_j, y'_j) are beacon i and beacon j coordinates at the new coordinate frame, and T_{ij} can be given by $T_{ij} = \cot(\phi_{ij})$.

In addition, the radius of this circle centre is derived from

$$R'^{\,2}_{ij} = \frac{\left(x'_i - x'_j\right)^2 + (y'_i - y'_j)^2}{4sin^2\phi_{ij}} \qquad \text{Ec.18}$$

According to [23], the power of any point, g, belonging to a circle, c, can be defined by Ec.19, where (x_c, y_c) are the circle centre coordinates and R is the circle radius.

$$\mathcal{P}_{c,g} = (x - x_c)^2 + (y - y_c)^2 - R^2 \quad \text{Ec.19}$$

Then, the power line of two circles can be obtained by equating the power of the points relatively to two different circles. For instance, the power line of c_{ij} and c_{jq} is given by

$$(x - x_{ij})^2 + (y - y_{ij})^2 - {R_{ij}}^2 = (x - x_{jq})^2 + (y - y_{jq})^2 - {R_{jq}}^2 \quad \text{Ec.19}$$

The second phase goal is to determine the circles intersection. Assuming a scenario with three known beacon positions, one robot location and the new coordinate frame, the three power lines will intersect in a specific point, which is the point to be determined. One way of doing this is to solve the linear system based on the three power lines (see Ec.20).

$$x'(x'_{ij} - x'_{jq}) + y'(y'_{ij} - y'_{jq}) = \frac{{x'}^2_{ij} + {y'}^2_{ij} - {R'}^2_{ij}}{2} - \frac{{x'}^2_{jq} + {y'}^2_{jq} - {R'}^2_{jq}}{2} \quad \text{Ec.20}$$

$$x'(x'_{jq} - x'_{qi}) + y'(y'_{jk} - y'_{ki}) = \frac{{x'}^2_{jq} + {y'}^2_{jq} - {R'}^2_{jq}}{2} - \frac{{x'}^2_{qi} + {y'}^2_{qi} - {R'}^2_{qi}}{2}$$

$$x'(x'_{qi} - x'_{ij}) + y'(y'_{qi} - y'_{ij}) = \frac{{x'}^2_{qi} + {y'}^2_{qi} - {R'}^2_{qi}}{2} - \frac{{x'}^2_{ij} + {y'}^2_{ij} - {R'}^2_{ij}}{2}$$

However, after some mathematical operations and simplifications, above linear system can be simplified to Ec.21, where it is omitted the ½ factor involved in the circle centre coordinates due to it cancels in the robot position coordinates. Moreover, it is also introduced the ${k'}_{ij}$ variable, which collects all circle parameters in a single variable. The variable ${k'}_{ij}$ can be computed using Ec.22, where $(x'_i y'_i)$ are the beacon coordinates relative to the new coordinates frame and T_{ij} is the cotangent of the bearing angle ϕ_{ij} between the beacons $\boldsymbol{pb_i}$ and $\boldsymbol{pb_j}$.

$$x'(x'_{ij} - x'_{jq}) + y'(y'_{ij} - y'_{jq}) = k'_{ij} - k'_{jq} \quad \text{Ec.21}$$

$$x'(x'_{jq} - x'_{qi}) + y'(y'_{jk} - y'_{ki}) = k'_{jq} - k'_{qi}$$

$$x'(x'_{qi} - x'_{ij}) + y'(y'_{qi} - y'_{ij}) = k'_{qi} - k'_{ij}$$

Any simple manner can be used to solve the above mentioned linear equations system. In this case, it is applied Cramer's rule.

$$k'_{ij} = x'_i x'_j + y'_i y'_j + T_{ij}(x'_j y'_i - x'_i y'_j) \quad \text{Ec.22}$$

In addition, it is possible to verify that the three circles are concurring if

$$\phi_{qi} = -\phi_{jq} - \phi_{ij} \quad \text{Ec.23}$$

5.2.2.1. Error Characterization of the estimated solution

According to the sensitivity analysis performed in [23], the deviation from the true robot position can be estimated by

$$\Delta x_i = \frac{1}{|D|}\Delta\phi g(\cdot) \quad \text{Ec.24}$$

$$\Delta y_i = \frac{1}{|D|}\Delta\phi h(\cdot) \quad \text{Ec.25}$$

where $g(\cdot)$ and $h(\cdot)$ are a function of all other parameters and D can be calculated by Ec.26.

$$D = (x'_{ij} - x'_{jk})(y'_{jk} - y'_{ki}) - (y'_{ij} - y'_{jk})(x'_{jk} - x'_{ki}) \quad \text{Ec.26}$$

Consequently, the inverse of the $|D|$ parameter can be used as a quality indicator of the robot position estimation. For that reason, it is interesting to keep $1/|D|$ value minimum, which means that the mean error of the position estimation is as low as possible and the robot scans are performed in the correct positions. The value of $1/|D|$ increases once the robot scanning points are closer to each other or the power lines trends to be parallel.

THIS PAGE INTENTIONALLY LEFT BLANK

6. ASSUMPTIONS AND LIMITATIONS

The main purpose of this section is to summarize all assumptions and limitations found during the development of this project. For each assumption or limitation, it is given a brief justification and the section where it is stated (see Table 3).

Ref.	*Section*	
A.01	3.1	*Linear and angular motions are decoupled. It means that mobile robot can move forward or rotate, but not both at the same time.* The reason for that is the system is considered as a discrete-time control system with no large speed changes. Furthermore, the kinematic model computation is simple when linear and angular motions are decoupled.
A.02	3.1	*Both DC motors provides same response for a specific power input.* In any ideal situation, both motors should have same performances. However, this is not reached in this case where motor A has lower angular speed than motor B. For simplicity, it is assumed same performances in both DC motors and the inequality is solved modifying the PWM.
A.03	3.1	*Operating voltage selected for all the experimental tests is +7 V_{DC}.* This is due to the angular and linear motion shows a linear behaviour within the range +6.8 V_{DC} to +7.5 V_{DC}, therefore, it is interesting to keep the operating voltage somewhere between these values.
A.04	7	*Mobile robot displacements are performed on flat and mostly level surface.* The reason for this assumption is that it is intended to simulate the robot operation as close as possible to real indoor environments.
A.05	3.1	*No large-displacements between measurements. Maximum linear motion 2 meters and angular motion less than 360 degrees.* This is one way to limit the operation scenario and, therefore, the ultrasonic sensors lectures. Furthermore, it is intended to keep the operation as close as possible to any real mobile robot indoor operation, which usually performs no large displacements.
A.06	2.1.4	*Servomotor rotation range is from 0 to 180 degrees (counter wise).* This is a servomotor model limitation but the experimental tests are adapted to this condition.
A.07	7	*The mobile robot cannot be closer than 0.100 m to any beacon.* The reasons for that distance value are (1) to be sure that the mobile robot has enough area to turn without hitting the beacon and (2) to fulfil assumption A.08 ($r_i >> \sigma_i$).
A.08	5.1.2	*The sensor measurement errors are considered Gaussian error, zero mean and normally distributed.* This assumption is made because the robot position is not near to any beacon and the range measurements are always larger than the standard deviation of the measurement errors, $r_i >> \sigma_i$, in all cases.

Table 3. General Assumptions and limitations

THIS PAGE INTENTIONALLY LEFT BLANK

7. SIMULATED AND EXPERIMENTAL RESULTS

As mentioned previously, trilateration and triangulation techniques are used to solve the three different bi-dimensional single target positioning cases considered in this project. These cases are (1) single target positioning using trilateration and multiple beacons, (2) single target positioning using trilateration and a unique beacon, and (3) single target positioning using triangulation and a unique beacon.

For each of the above-mentioned cases, it is done several experimental tests, which complexity increases gradually. The total number of tests is eight. It is assumed that the mobile robot displacements are performed on flat and mostly level surface (see A.04 Table 3).

Regarding the Rover 5 motion, it is used the robot kinematic model detailed in section 3.1 as a real reference.

The test scenario is defined specifically on each case and it varies depending on test requirements. The ultrasonic sensors area needs to be perfectly clear and restricted in order to avoid the detection of any other obstacle instead of the corresponding beacons (see A.05 Table 3). In case of false-positive beacons are still detected, the range measurements are limited in all tests. For that reason, it is assumed that distance between the robot and the beacon cannot be lower than d_{min}, which is always defined as 0.100 m (see A.07 Table 3). If so, the robot turns clock-wise automatically according to the robot functionalities (see section 2.1). Likewise, the distance between the robot and the beacon cannot be larger than d_{max}, which value is given by

$$d_{max} = \sqrt{(x_{max} - d_{min})^2 + (y_{max} - d_{min})^2} \qquad \text{Ec.27}$$

where x_{max} and y_{max} are the scenario dimensions x and y, respectively, and they are defined before each test.

Finally, the units of measurement are those defined by the international system (SI).

7.1. CASE 01: 2D Single Target Positioning Using Trilateration And Multiple Beacons

The experimental tests detailed in this section are based on a single robot positioning using trilateration and multiple beacons. In this case, the total number of tests performed is two. In both cases, the mobile robot includes three ultrasonic sensor modules, which are fixed to the robot chassis. So, the main difference between the tests is the number of beacons and if the robot remains stationary or not, meanwhile the test is being carried out. Table 4 contains a brief description of each test performed in case 01.

Test	*Beacon*			*Sensors*			*Rover 5 status when sensors are reading*
	Qty.	*Status*	*Type*	*Qty.*	*Status*	*Type*	
TST.01	3	Stationary	Passive beacon	3	Stationary	SR-HC04	Stationary
TST.02	9	Stationary	Passive beacon	3	Stationary	SR-HC04	In motion

Table 4. Case 01: Tests summary

7.1.1. Test 01 (TST.01): Range Measurements with Stationary Robot

7.1.1.1. TST.01 Description

The motivation to perform TST.01 is threefold: (1) to be familiar with HC-SR04 ultrasonic module operating mode, (2) to verify trilateration algorithm problems due to measurement errors and (3) to get used to Linear Least Square algorithm by executing a simple case with stationary robot.

The scenario dimensions are limited to $2.000\ x\ 2.000\ \mathrm{m}^2$. The reason for this is that the ultrasonic sensors area needs to be perfectly clear and restricted in order to avoid the detection of any other obstacle instead of the beacons. The origin of the inertial frame is located at $(0,0)$, which corresponds to centre-left of the scenario previously defined. Thus, three stationary beacons are placed arbitrarily in the above mentioned area, which positions are $B1(0.350, 0.300)$, $B2(0.350, -0.450)$ and $B3(1.010, -0.150)$ (see Figure 24). In this case, the robot Rover 5 platform keeps stationary in an unknown location $\mathrm{p}(\mathrm{x}, \mathrm{y})$ during the measurements.

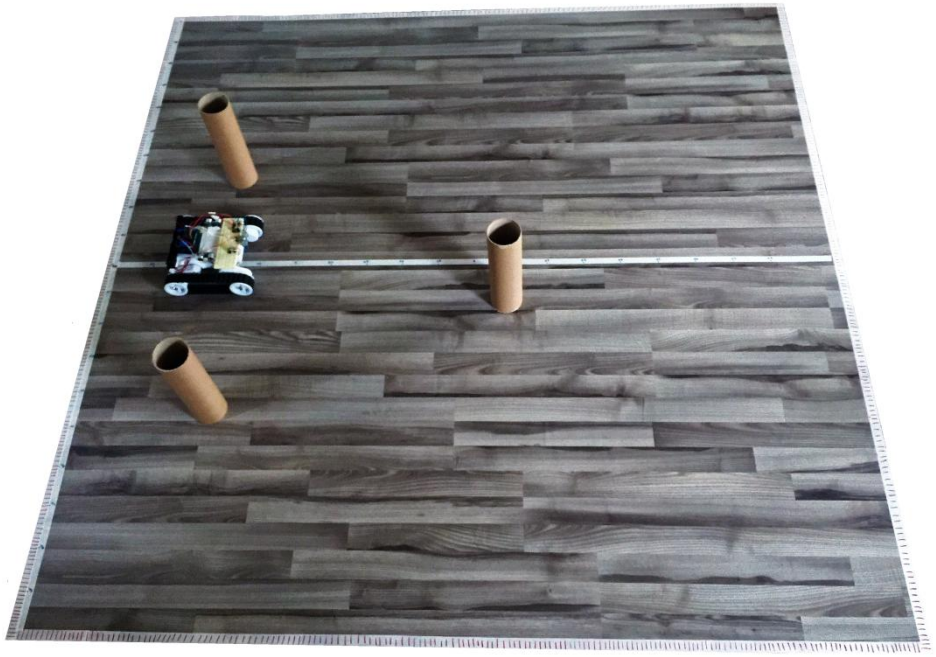

Figure 24. TST.01- Test set-up1

In order to carry out the TST.01, three HC-SR04 ultrasonic modules are fixed to the robot platform (see Figure 6). These will provide the range measurements d_1, d_2 and d_3.

7.1.1.2. TST.01 Algorithm

In this section it is detailed the algorithm used to carry out the test 01.

Algorithm TST.01

Given three stationary beacons ($N = 3$) with known coordinates, B1(x_1, y_1), B2(x_2, y_2) and B3(x_3, y_3):

1) Compute the three sensors range measurements (d_1, d_2, d_3) using the TOA ranging technique. It is considered the ultrasound waves travel at the speed of sound, approximately $340\ m/s$ or $0.00034\ m/\mu s$.

$$d_1 = \frac{t_1 \cdot c}{2};\quad d_2 = \frac{t_2 \cdot c}{2};\quad d_3 = \frac{t_3 \cdot c}{2}$$

2) Verify distance measurements according to the scenario dimensions and minimum distance allowed to any beacon (eliminate false-positive):

$$d_{min} \le d_1 \le d_{max};\quad d_{min} \le d_2 \le d_{max};\quad d_{min} \le d_3 \le d_{max}$$

3) Compute *b* matrix:

$$b = \begin{bmatrix} d_1^2 - x_1^2 - y_1^2 - d_3^2 + x_3^2 + y_3^2 \\ d_2^2 - x_2^2 - y_2^2 - d_3^2 + x_3^2 + y_3^2 \end{bmatrix}$$

4) Compute *A* matrix:

$$A = -2 \cdot \begin{bmatrix} x_1 - x_3 & y_1 - y_3 \\ x_2 - x_3 & y_2 - y_3 \end{bmatrix}$$

5) Compute the inverse of the *A* matrix:

$$A^{-1} = \frac{1}{|A|} \cdot \begin{bmatrix} y_2 - y_3 & -(y_1 - y_3) \\ -(x_2 - x_3) & x_1 - x_3 \end{bmatrix}$$

6) Multiply A^{-1} and *b* matrices to get robot position:

$$p = A^{-1} b$$

7.1.1.3. TST.01 Results

After performing the test, the three distances obtained with the ultrasonic sensors are $d_1 = 0.370$ m, $d_2 = 0.440$ m and $d_3 = 0.380$ m.

Figure 25 shows the beacons location and the three circles raised from distance measurements and beacons position to carry out the trilateration technique. As expected, the three circles never intersect each other and only an area can be defined where the

robot could be located. It confirms that the range-only target positioning needs to be solved using any mathematical approach.

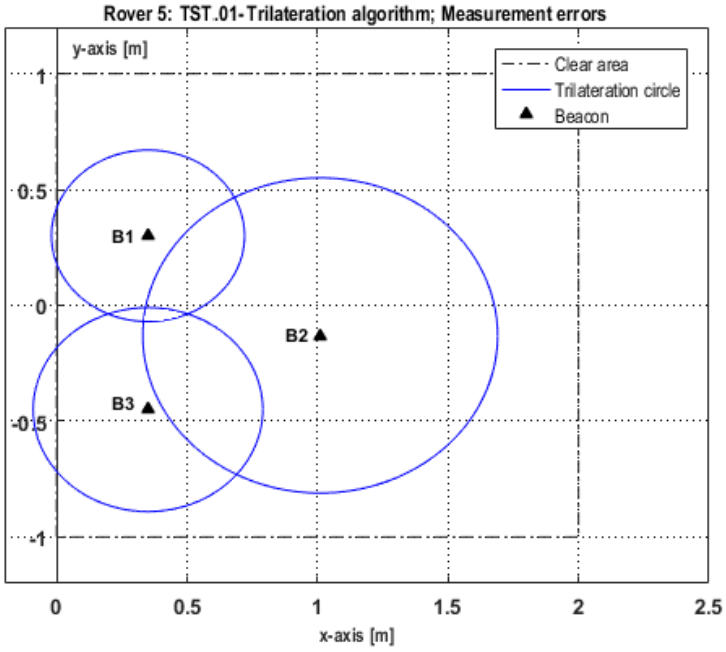

Figure 25. TST.01- Trilateration algorithm and range measurement errors

The robot position is finally collected with the Linear Least Square algorithm, concluding that there is barely any difference between the computed robot position, $p(0.354, -0.037)$, and the real robot position estimated manually $p(0.300, 0)$. In this case, the absolute error is 0.054 m on x-axis and 0.037 m on y-axis.

7.1.1.4. TST.01 Conclusions

Due to the range measurements noise, it is needed to solve trilateration problem using any mathematical approach, such as Least Squares.

The test results show that there is a few centimetres deviation in position estimation on both axes, which matches perfectly with the performances of the ultrasonic sensors used and the expectations of the algorithm applied.

7.1.2. Test 02 (TST.02): Range Measurements with Moving Robot

7.1.2.1. TST.02 Description

The main goal of TST.02 is to verify the robot capabilities computing its position while it travels from a pre-defined initial point to a pre-defined final point using trilateration technique.

The scenario dimensions are limited to $2.000\ x\ 2.000$ m^2, and it is assumed that TST.02 uses same robot sensor configuration than the one used in TST.01. The origin of the inertial frame is located at $(0,0)$, which corresponds to centre-left of the scenario previously defined.

Initially, the robot starts its trajectory in $p_o(0,0)$ (see Figure 26). Then, it moves linearly until the distance detected to any beacon is lower or equal than 0.1 meters. If so, the robot rotates $90°$ (approximately 3 seconds) CW or CCW depending on which sensor detects the beacon closer. For example, if sensor (1) detects the beacon, the robot will rotate $90°$ degrees CW. Conversely, if sensor (3) detects the beacon, the robot will rotate $90°$ CCW. Finally, if the sensor front-pointing (2) detects the beacon, the robot will also turn $90°$ CW. The robot aimed position is $p_f(1.400, -0.600)$.

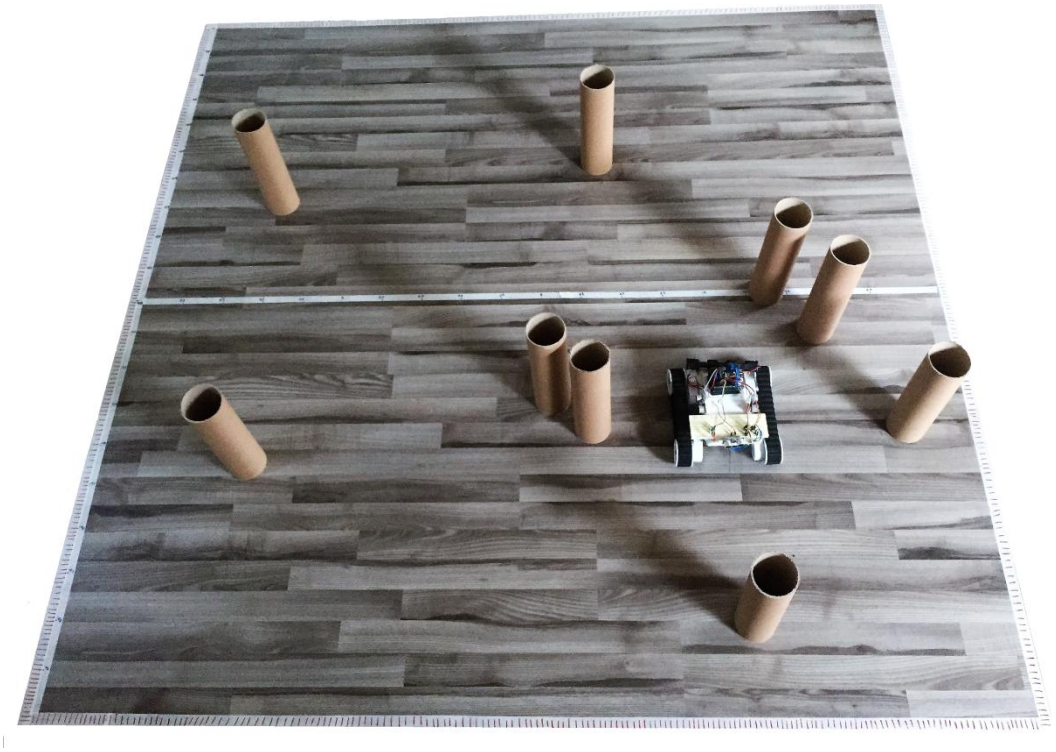

Figure 26. TST.02- Test set-up

In this test, nine stationary beacons are arbitrarily located according to the scenario dimensions. The beacons positions are given in Table 5. The Rover 5 estimates three range measurements d_1, d_2 and d_3 to the three closer beacons. Then, these distances are used to estimate the robot position. Besides, the robot keeps its motion while the measurements are taking place. If there are no available distance measurements, i.e. any beacon can be reached by none of the three sensors; the robot continues its linear displacement normally and it is not performed any position estimation.

$\boldsymbol{B_i(x_i\ [\mathrm{m}], y_i\ [\mathrm{m}])}$								
$\boldsymbol{B_1}$	0.350	0.300	$\boldsymbol{B_4}$	1.050	0.420	$\boldsymbol{B_7}$	1.150	−0.380
$\boldsymbol{B_2}$	0.350	−0.450	$\boldsymbol{B_5}$	1.550	0	$\boldsymbol{B_8}$	1.450	−0.800
$\boldsymbol{B_3}$	1.010	−0.300	$\boldsymbol{B_6}$	1.650	−0.150	$\boldsymbol{B_9}$	1.800	−0.380

Table 5. TST.02- Arbitrary beacon positions

7.1.2.2. TST.02 Algorithm

In this section it is detailed the algorithm used to carry out the test 02.

Algorithm TST.02

Given nine stationary beacons with known coordinates, $B_i(x_i, y_i)$; $i = 1,2, \dots N$; $N = 9$:

1) Compute the three sensors range measurements (d_1, d_2, d_3) using the TOA ranging technique. It is considered the ultrasound waves travel at the speed of sound, approximately $340\ m/s$ or $0.00034\ m/\mu s$.

$$d_1 = \frac{t_1 \cdot c}{2};\quad d_2 = \frac{t_2 \cdot c}{2};\quad d_3 = \frac{t_3 \cdot c}{2}$$

2) Verify distance measurements according to the scenario dimensions and the minimum distance allowed to any beacon (eliminate false-positive):

$$d_{min} < d_1 \leq d_{max};\quad d_{min} < d_2 \leq d_{max};\quad d_{min} < d_3 \leq d_{max}$$

3) Estimate robot position using the robot kinematic model:

$$p_{mod} = (x_{mod}, y_{mod})$$

4) Compute the vector including all distances between the estimated robot position and each beacon position:

$$d_{est_i} = \sqrt{(x_{mod} - x_i)^2 + (y_{mod} - y_i)^2}\ ;\ i = 1,2,3 \dots N$$

5) Correlate each estimated distance with its corresponding beacon position using known beacon locations.

6) Select the three beacons positions with lower estimated distances (it is assumed they are the closer beacons):

$$d_1, B_i(x_i, y_i);\quad d_2, B_j(x_j, y_j);\quad d_3, B_k(x_k, y_k)$$

7) Compute b matrix:

$$b = \begin{bmatrix} d_1^2 - x_i^2 - y_i^2 - d_3^2 + x_k^2 + y_k^2 \\ d_2^2 - x_j^2 - y_j^2 - d_3^2 + x_k^2 + y_k^2 \end{bmatrix}$$

8) Compute A matrix:

$$A = -2 \cdot \begin{bmatrix} x_i - x_k & y_i - y_k \\ x_j - x_k & y_j - y_k \end{bmatrix}$$

9) Compute the inverse of the A matrix:

$$A^{-1} = \frac{1}{|A|} \cdot \begin{bmatrix} y_j - y_k & -(y_i - y_k) \\ -(x_j - x_k) & x_i - x_k \end{bmatrix}$$

10) Multiply A^{-1} and b matrices to get robot position:

$$p = A^{-1} b$$

7.1.2.3. TST.02 Results

The robot pre-defined start point is $p(0,0)$ and the final point is $p(1.400,-0.600)$. The robot uses the three closer beacons to determine its own position between these two points. Additionally, the robot kinematic model is used to estimate the robot real position while it is moving. In this case, the ultrasonic sensors rate measurements is one measurement every 0.18 s.

Figure 27 shows the robot estimated positions according to the different beacons located at this scenario. Thus, for example, the first three beacons used to estimate the robot position are B1, B2 and B3, and then it uses B3, B4 and B5, and so on. From Figure 27, it is easy to differentiate the areas where the robot is unable to estimate its position, which is due to the lack of sight between the beacon and the ultrasonic sensor mounted with a specific orientation in the robot chassis.

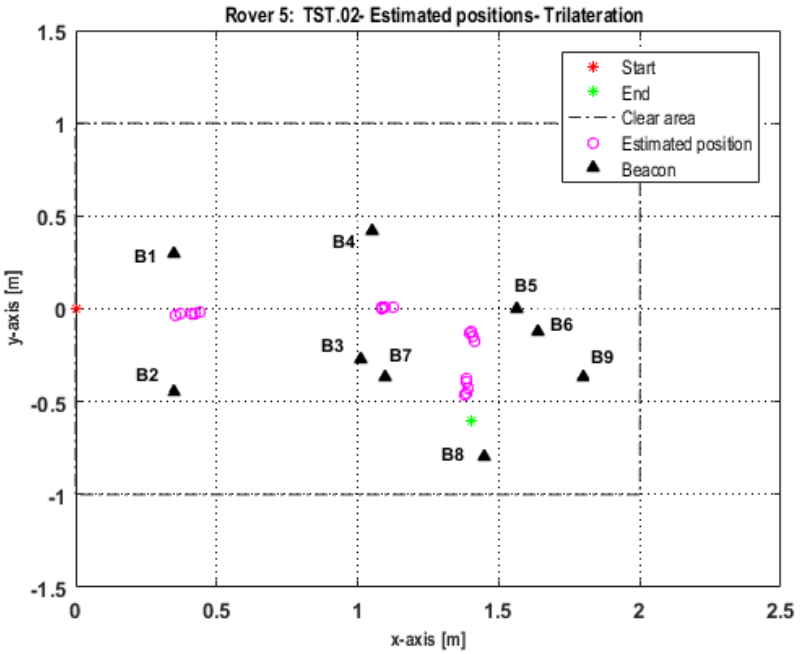

Figure 27. TST.02- Estimated robot trajectory using trilateration technique

The mobile robot trajectory obtained by the kinematic model is compared with the trajectory estimated using located beacons and trilateration in Figure 28. The estimated path is obtained by linear interpolation taking into account the points collected.

In this case, the maximum absolute error is 0.025 m on x-axis and 0.037 m on y-axis.

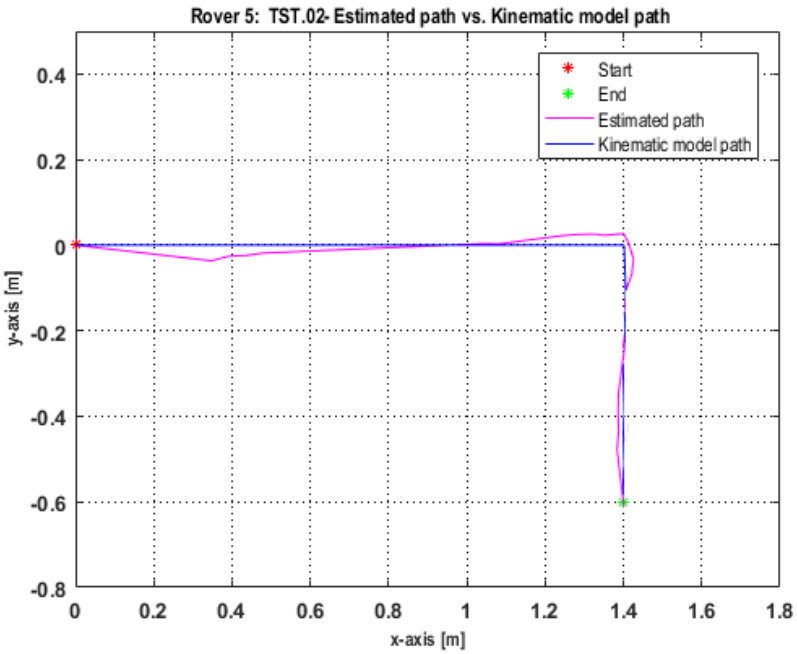

Figure 28. TST.02- Kinematic model vs. Estimated mobile robot trajectory

Finally, the covariance and the mean error of the estimation are computed. Table 6 contains the numeric results obtained.

Beacons	*Cov matrix* $\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}\cdot 10^{-2}$	*Mean error*	
		$x \cdot 10^{-2}$	$y \cdot 10^{-2}$
$\boldsymbol{B_1, B_3, B_2}$	$\begin{pmatrix}0.468 & 0.040\\ 0.040 & 0.005\end{pmatrix}$	0.002	2.750
$\boldsymbol{B_4, B_5, B_3}$	$\begin{pmatrix}0.280 & 0.023\\ 0.023 & 0.002\end{pmatrix}$	0.003	0.689
$\boldsymbol{B_6, B_7, B_3}$	$\begin{pmatrix}0.001 & -0.002\\ -0.002 & 0.255\end{pmatrix}$	0.638	0.004
$\boldsymbol{B_9, B_8, B_7}$	$\begin{pmatrix}0.004 & -0.054\\ -0.054 & 1.140\end{pmatrix}$	1.010	0.003

Table 6. TST.02- Covariance and mean error results

7.1.2.4. TST.02 Conclusions

Comparing the model trajectory and the estimated path, it is concluded that there is no large difference between both trajectories.

As can be seen in Table 6, the mean error varies depending on the robot motion. For example, the mean error on x-axis is almost negligible while the robot is moving linearly through x-axis without any displacement registered on y-axis. By contrast, the mean error on y-axis is insignificant while the robot is moving linearly through y-axis without any displacement registered on x-axis. Likewise, the x and y estimations are quite independent because the values of the entries outside the main diagonal of the covariance matrix are close to zero on each measurement.

7.2. CASE 02: 2D Single Target Positioning Using Trilateration And A Single Beacon

In this case, it is carried out four different tests considering a single beacon and solving the single robot positioning problem by trilateration. The mobile robot is equipped with a unique ultrasonic sensor module, which is placed over a rotating platform (see Figure 8). Hence, the robot scans its environment from diverse positions. The main difference between the tests is position where the robot scans its surroundings. Table 7 contains a brief description of each test performed in case 02.

Test	*Scanning positions*	*Sensors*			*Rover 5 status when sensor is reading*
		Qty.	*Status*	*Type*	
TST.03	Arbitrary	1	Rotating	SR-HC04	Stationary
TST.04	Optimal	1	Rotating	SR-HC04	Stationary
TST.05	Arbitrary + Optimal	1	Rotating	SR-HC04	Stationary
TST.06	Arbitrary + Optimal	1	Rotating	SR-HC04	Stationary

Table 7. Case 02: Tests summary

7.2.1. Test 03 (TST.03): Range Measurements at Arbitrary Positions

7.2.1.1. TST.03 Description

There are several goals performing TST.03. Firstly, it is attempted to verify that trilateration technique can provide suitable results using a single beacon and unique ultrasonic sensor module, which is able to scan the robot surroundings. Secondly, it is expected to investigate how the trilateration results are affected by the position where the range measurements take place. Finally, it is evaluated how the results improve when the number of range measurements increases.

As above mentioned, it is used the robot design 2 in this case. According to this configuration, it is always computed one distance measurement per each degree scanned.

The scenario dimensions are limited to $1.800\,x\,1.800$ m^2, in order to avoid false-positive outputs from the ultrasonic sensors. The origin of the inertial frame is located at $(0,0)$, which corresponds to the lower-right corner of the scenario previously defined. A stationary beacon is placed arbitrarily in this area, which position is $B(0.400, 0.700)$. For simplicity, the y-axis is reversed due to the servomotor operating mode.

In this case, the Rover 5 is positioned somewhere taking into account the above mentioned scenario dimensions. Then, the robot keeps stationary while it performs a scan of its environment. After each scan, the robot performs an angular motion and afterwards a linear motion. Consequently, the robot orientation and position are modified before next scan. Both motions are previously defined.

7.2.1.2. TST.03 Algorithm

In this section it is detailed the algorithm used to carry out the test 03.

Algorithm TST.03

Given a single stationary beacon with known coordinates, $B(x_b, y_b)$:

1) Make the robot carries out a 180° CW scan at its current position, $p_i(x_i, y_i)$.
2) Compute the range measurement (d_k) using the TOA ranging technique. It is considered the ultrasound waves travel at the speed of sound, approximately 340 m/s or 0.00034 $m/\mu s$.

$$d_k = \frac{t_k \cdot c}{2}; \quad k = 1,2,3 \dots M; \quad M = 180$$

3) Verify distance measurements according to the scenario dimensions and the minimum distance allowed to any beacon (eliminate false-positive):

$$d_{min} < d_k \leq d_{max}$$

4) Identify the interval of interesting distances according to the effectual angle of the ultrasonic sensor. The result is a vector, $\boldsymbol{dm}$, including few distance measurements:

$$\{Int_o, Int_f\}; \ \boldsymbol{dm} = (d_{Int_o} : d_{Int_f})$$

where Int_o is the lower limit and Int_f is the upper limit of the above mentioned interval. Accordingly, the initial and final range measurements of the $\boldsymbol{dm}$ vector are d_{Int_o}, and d_{Int_f}, respectively

5) Compute the arithmetic mean of the $\boldsymbol{dm}$ vector:

$$\overline{dm} = \frac{\sum_{d=1}^{n} d_d}{n}$$

where n is the number of range measurements between the upper and lower interval limits

6) Make the robot performs a pre-defined angular and linear displacement:

$$\{\Delta\theta, \Delta x, \Delta y\}$$

7) Estimate new robot position considering its previous position and robot displacements:

$$p_i\{x_{i-1} + \Delta x_i \, , y_{i-1} + \Delta y_i\}$$

8) Before (9), repeat from (1) to (8) until $i = N$; where N is the total number of scans that need to be performed.

9) Compute b matrix:

$$b = \begin{bmatrix} d_i^2 - \Delta x_i^2 + 2\Delta x_i x_b - \Delta y_i^2 + 2\Delta y_i y_b - d_N^2 + \Delta x_N^2 - 2\Delta x_N x_b + \Delta y_N^2 - 2\Delta y_N y_b \\ \vdots \\ d_{N-1}^2 - \Delta x_{N-1}^2 + 2\Delta x_{N-1} x_b - \Delta y_{N-1}^2 + 2\Delta y_{N-1} y_b - d_N^2 + \Delta x_N^2 - 2\Delta x_N x_b + \Delta y_N^2 - 2\Delta y_N y_b \end{bmatrix};$$

$$i = 1,2,3 \dots N$$

10) Compute A matrix:

$$A = -2 \cdot \begin{bmatrix} \Delta x_i - \Delta x_N & \Delta y_i - \Delta y_N \\ \vdots & \vdots \\ \Delta x_{N-1} - \Delta x_N & \Delta y_{N-1} - \Delta y_N \end{bmatrix}; i = 1,2,3 \dots N$$

11) Compute the inverse and the transpose of the A matrix.
12) Multiply matrices to get robot position:

$$p = (A^T A)^{-1} A^T \cdot b$$

7.2.1.3. TST.03 Results

The results of TST.03 are divided into four different samples depending on the number of range measurements (N) taken into account.

7.2.1.3.1. TST.03- Sample A: N=3

Initially, it is evaluated a simple situation where the robot only performs linear displacements after each scan, i.e., $\Delta x = 0.3$ m and $\Delta y = 0$ m. It means that the scanning points are collinear.

After performing the test, it is concluded that trilateration technique cannot be applied when the scanning points are collinear because indeterminate matrices are obtained during the trilateration process. Thus, the experimental result is ambiguous. However, this is not the case when same problem is solved theoretically.

Henceforward, it is assumed that the Rover 5 is always carrying out an angular motion followed by a linear motion after each scan; just to be sure that the scanning points are not collinear.

For the reason above mentioned, same test is repeated considering that the robot rotates 30° CW, and then, it moves forward 0.193 m after each scan. This test is repeated three times (IT1, IT2 and IT3).

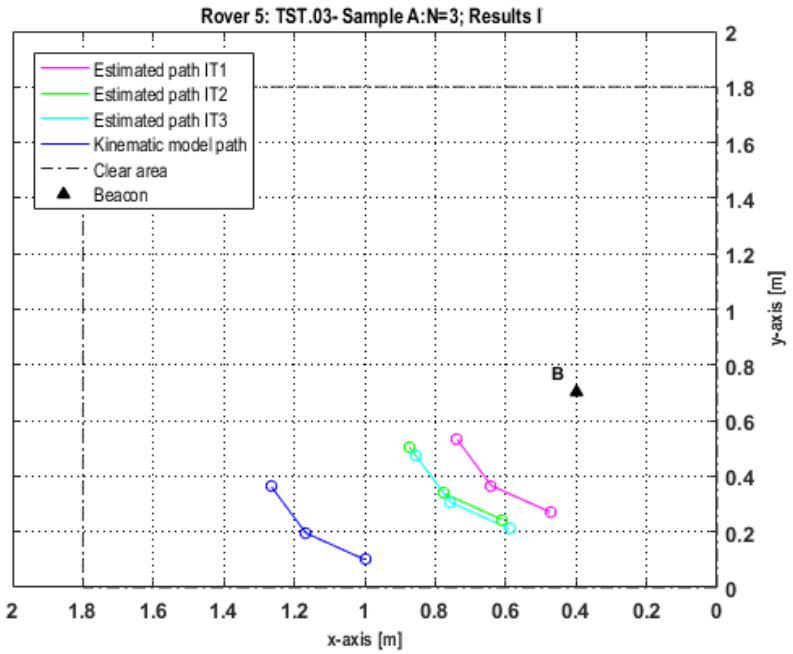

Figure 29. TST.03- Sample A: N=3; Results I

According to Figure 29, there is a large difference between the estimated and the real mobile robot position. Thus, the absolute error is 0.443 m on x-axis, and 0.142 m on y-axis, concluding that these greater differences can be due to the number of scans performed, but also because the scanning points are quite close to each other. The absolute error is computed considering all the values collected on each iteration.

In addition, the error on x-axis is higher than on y-axis, which is probably due to the positions where the measurements are taking place, i.e., only measurements in half-plane.

Finally, it is repeated same test considering three non-collinear points with mobile robot larger displacements (≥ 0.283 m). In this case, the robot rotates 65.890° CW and moves 0.680 m after each scan. This test is repeated three times (IT1, IT2 and IT3, respectively).

As seen in Figure 30, the results are greatly improved by increasing robot displacements between the scanning points. However, the absolute errors on x axis (0.184 m) and on y axis (0.141 m) are still quite large, which is probably due to the number of scanning points is rather limited. Furthermore, the difference between the results obtained on each iteration is trivial.

This absolute error is compared with the one obtained with three stationary beacons and three ultrasonic modules (see section 7.1.1), concluding that the estimated position absolute error has almost quadrupled with one single beacon and a unique ultrasonic sensor module.

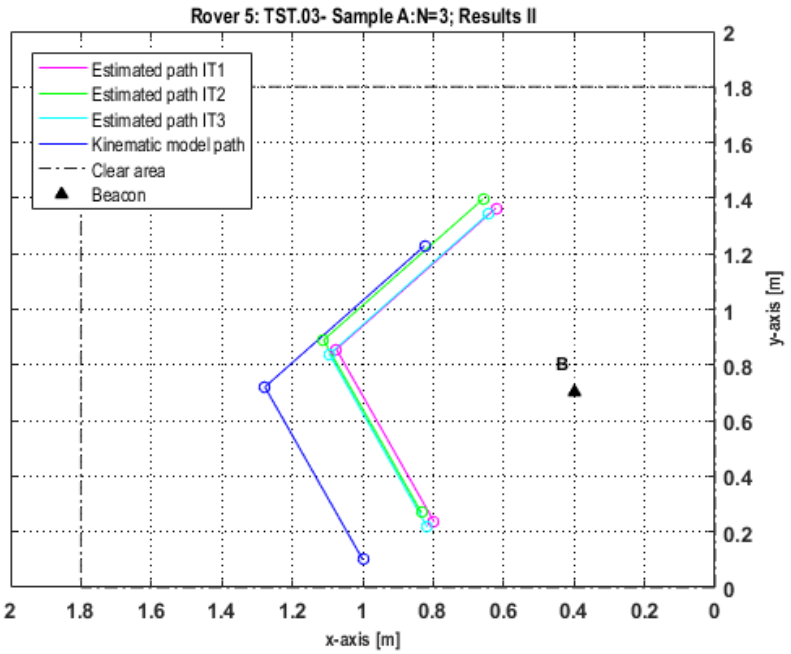

Figure 30. TST.03- Sample A: N=3; Results II

7.2.1.3.2. TST.03- Sample B: N=4

In this case, it is considered four different range measurements ($N = 4$). Furthermore, it is taken into account that the scanning points cannot be collinear, and also, that the robot displacements have to be ≥ 0.283 m.

Consequently, it is assumed that Rover 5 rotates $65.890°$ CW, and then, it moves linearly 0.650 m after each scan. This test is repeated twice (see Figure 31).

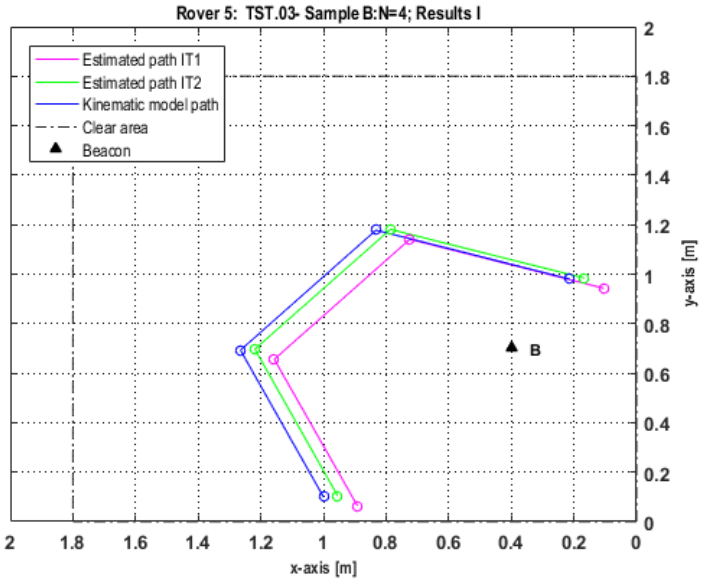

Figure 31. TST.03- Sample B: N=4; Results I

According to Figure 31, the absolute error is 0.046 m on x-axis and 0.017 m on y-axis. In this case, the estimated position absolute error is lower than the one obtained with three stationary beacons and three ultrasonic modules.

7.2.1.3.3. TST.03- Sample C: N=5

This experiment is equal to the one performed in the previous section, but it is considered an additional range measurement; therefore, the total number of scanning points is $N = 5$.

Again, the Rover 5 rotates 65.890° CW and moves linearly 0.650 m after each scan. This test is repeated a couple of times, and the results are shown in Figure 32.

In this case, the absolute error is 0.016 m on x-axis and 0.004 m on y-axis.

7.2.1.3.4. TST.03- Sample D: N=6

In sample D, it is considered six different range measurements ($N = 6$). After each scanning, the Rover 5 rotates 59.587° CW and then, it moves linearly 0.650 m. Again, the test is performed a couple of times, and the results are shown in Figure 33.

After, it is computed that absolute error on both axes, concluding that it is 0.017 m on x- axis and 0.007 m on y-axis.

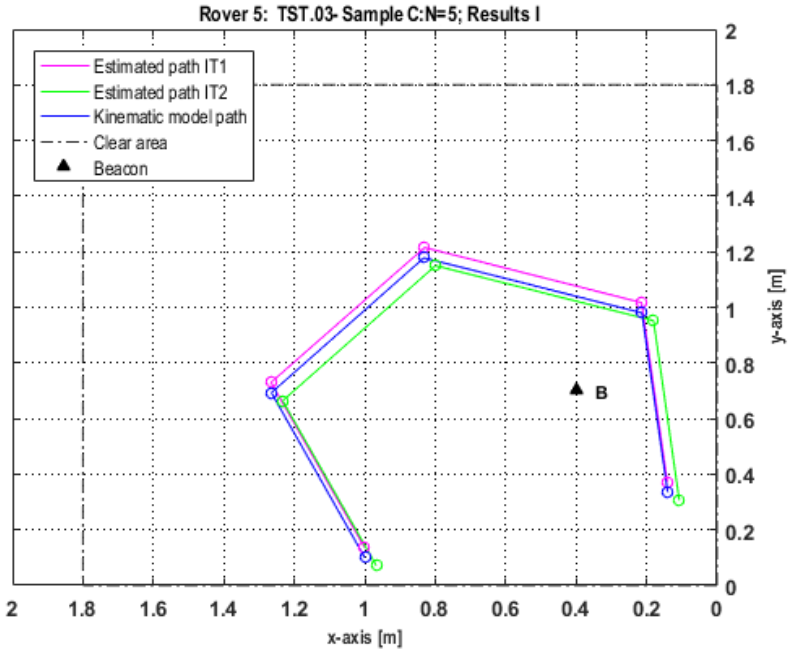

Figure 32. TST.03- Sample C: N=5; Results I

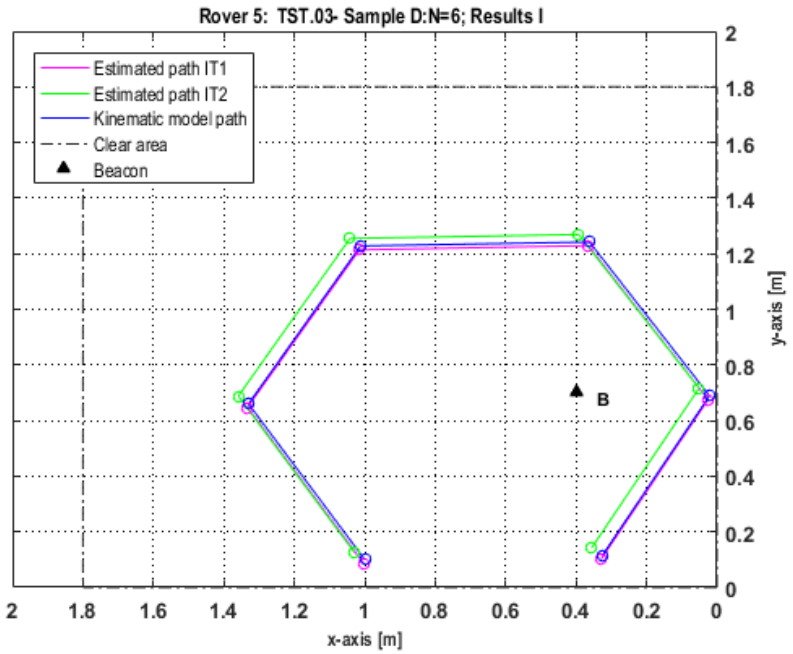

Figure 33. TST.03- Sample D: N=6; Results I

As seen in Figure 34, the absolute error decreases on both axes when the number of scans increases. Besides, it is also computed the Fisher Information Matrix and its determinant for each range measurement performed (see Table 8).

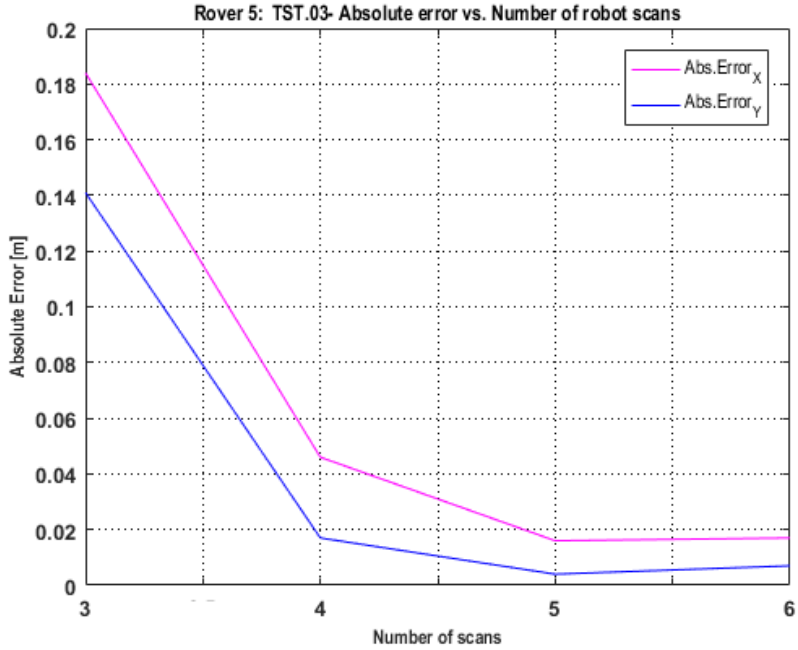

Figure 34. TST.03- Absolute error vs. Number of robot scans

		$N=3$	$N=4$	$N=5$	$N=6$
Experimental	FIM (10^6)	$\begin{pmatrix} 1.258 & -0.022 \\ -0.022 & 0.630 \end{pmatrix}$	$\begin{pmatrix} 1.421 & 0.297 \\ 0.297 & 1.097 \end{pmatrix}$	$\begin{pmatrix} 1.843 & -0.030 \\ -0.030 & 1.304 \end{pmatrix}$	$\begin{pmatrix} 1.719 & -0.078 \\ -0.078 & 2.058 \end{pmatrix}$
	$\lvert FIM \rvert$	$7.925 \cdot 10^{11}$	$1.471 \cdot 10^{12}$	$2.403 \cdot 10^{12}$	$3.532 \cdot 10^{12}$
Theoretical	FIM (10^6)	$\begin{pmatrix} 0.944 & 0 \\ 0 & 0.944 \end{pmatrix}$	$\begin{pmatrix} 1.259 & 0 \\ 0 & 1.259 \end{pmatrix}$	$\begin{pmatrix} 1.574 & 0 \\ 0 & 1.574 \end{pmatrix}$	$\begin{pmatrix} 1.888 & 0 \\ 0 & 1.888 \end{pmatrix}$
	$\lvert FIM \rvert$	$8.916 \cdot 10^{11}$	$1.585 \cdot 10^{12}$	$2.476 \cdot 10^{12}$	$3.566 \cdot 10^{12}$
|FIM| Normalized		0.888	0.928	0.971	0.989

Table 8. TST.03- Sample A,B, C& D: Covariance matrix, FIM and FIM determinant

Finally, the test is repeated with six scanning points but combining different robot displacements and diverse robot rotations between each scan. In case D.1, it is considered constant linear displacement, 0.650 m, and several angular displacements, whereas in case D.2 it is not considered constant linear motion neither angular motion. Table 9 contains all the linear and angular motions for both cases.

The results are shown in Figure 35-Left and Figure 35-Right, respectively.

	Case D.1		*Case D.2*	
	$\Delta\theta$ *[degrees]*	$(\Delta x, \Delta y)$ *[m]*	$\Delta\theta$ *[degrees]*	$(\Delta x, \Delta y)$ *[m]*
1→2	83	$(0.079, 0.645)$	50	$(0.418, 0.498)$
2→3	147	$(-0.545, 0.354)$	95	$(-0.026, 0.777)$
3→4	200	$(-0.611, -0.222)$	140	$(-0.598, 0.501)$
4→5	240	$(-0.325, -0.563)$	180	$(-0.228, 0)$
5→6	250	$(-0.222, -0.611)$	110	$(-0.222, 0.611)$

Table 9. TST.03- Robot linear and angular displacements for N=6

As can be seen in Figure 35, there are not significant differences between these two cases. Besides, the absolute error keeps low even when it is applied different angular and linear motions. In sample D.1, the absolute error is 0.027 m on x-axis and 0.009 m on y-axis. Regarding the sample D.2, this is 0.036 m on x axis and 0.013 m on y axis. The reason why the error is higher in sample D.2 is probably due to the robot trajectory, which is around the stationary beacon in sample D.1 and somewhere else in sample D.2.

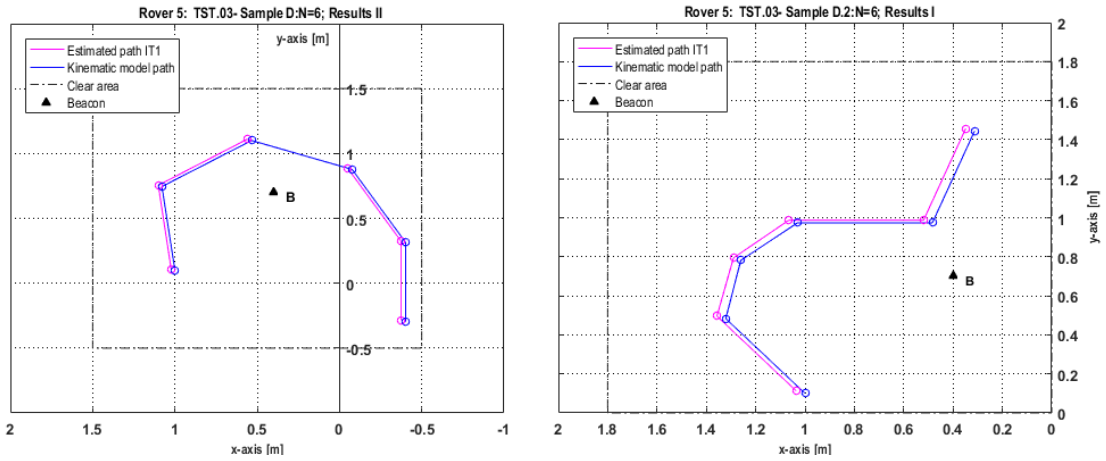

Figure 35. TST.03- Sample D.1 (Left) and Sample D.2 (Right) results

7.2.1.4. TST.03 Conclusions

As confirmed in previous sections, the estimated position varies between iterations although the robot performs same motions, which is something expected when it comes to physical measurements. This is owing to the measurements noise and it can be attenuated if the scans are performed in optimal points or points closer to these ones. In any case, these differences are not great and it is used the mean value for any subsequent computation.

In addition, it is verified that trilateration technique cannot be applied when the scanning points are collinear. It implies that the robot needs to vary its orientation and position before each scanning. Furthermore, the estimation position error improves when the robot performs vast angular and linear motions between each scan.

Lastly, it is also noted that the absolute error in both axes decreases when the number of scans increases. However, unlike in the theoretical developments, there is any trivial improvement from a specific number of range measurements and forward. For instance, looking at Figure 34, the absolute error is almost not reduced from $N = 5$ to $N = 6$; therefore, the benefits of performing six scans are not making up for the robot work load and estimation time. Moreover, the fact that the absolute error on x-axis is higher than on y-axis, it is related to where the scanning points are regarding the reference axes considered.

7.2.2. Test 04 (TST.04): Range Measurements at Optimal Positions

7.2.2.1. TST.04 Description

Main goals of this test are (1) to evaluate how the scans performed in optimal positions benefit the outcomes, (2) analyse the results depending on different types of optimal points, (3) compare the results obtained when the number of scans varies, (4)

characterize the estimation errors and (5) verify the estimation quality comparing the results with the Cramer-Rao Lower Bound (CRLB).

The Rover 5 sensor configuration is the same than the one used in section 7.2.1. According to this configuration, it is always computed one distance measurement per each degree scanned.

The scenario dimensions are limited to $1.400\,x\,1.400$ m^2, in order to avoid false-positive detected by the ultrasonic sensors. The origin of the inertial frame is located at $(0,0)$, which corresponds to the lower-right corner of the scenario previously defined. A stationary beacon is placed arbitrarily in this area, which position is $B(0.400, 0.600)$. For simplicity, the y-axis is reversed due to the servomotor operating mode.

In this case, the Rover 5 is positioned at the first optimal position identified. Then, the robot keeps stationary while it performs a scan of its environment. After each scan, the robot moves to the next optimal position.

7.2.2.2. TST.04 Algorithm

The TST.03 algorithm detailed in section 7.2.1 is also applicable for TST.04.

7.2.2.3. TST.04 Results

Recalling the information provided in section 5.1.2.3, there are different ways of getting the optimal points. In this project, a couple of them are analysed. On the one hand, it is evaluated the simplest optimal points defined by the polygon vertices. For instance, if the number of range measurements selected is $N = 3$, the optimal points are those located at $0°$, $120°$ and $240°$ regarding the stationary beacon. On the other hand, it is assessed the optimal positions solution based on the symmetric points, which is more efficient solution from the robot minimum-energy consumption point of view.

7.2.2.3.1. TST.04- Sample A: Optimal positions- Polygon vertices

In this case, it is performed several test with different number of scans (N), starting from $N = 3$ up to $N = 6$. Furthermore, the test is repeated once (1 IT).

Initially, the robot is placed at $(0.700, 0.400)$ and oriented $0°$. Then, it rotates CW and moves forward linearly according to a pre-defined values. Table 10 summarizes robot linear and angular motions depending on the number of scans.

	N = 3		N = 4		N = 5		N = 6	
	(Δx, Δy) [m]	Δθ [°]	(Δx, Δy) [m]	Δθ [°]	(Δx, Δy) [m]	Δθ [°]	(Δx, Δy) [m]	Δθ [°]
1→2	(−0.30,0.52)	120	(0,0.60)	90	(0.18,0.57)	72	(0.30,0.52)	60
2→3	(−0.30, −0.52)	120	(−0.6,0)	90	(−0.48,0.35)	72	(−0.30,0.52)	60
3→4			(0, −0.6)	90	(−0.48, −0.35)	72	(−0.60,0)	60
4→5					(0.18, −0.57)	72	(−0.30, −0.52)	60
5→6							(0.30, −0.52)	60

Table 10. TST.04- Sample A: Robot linear and angular motions

Figure 36 shows a comparison between robot estimated path and the trajectory computed using the robot kinematic model. At first glance, the results are quite accurate just using the minimum number of scans ($N = 3$). The absolute error in the estimated path is even lower than the one obtained in section 7.1.1.

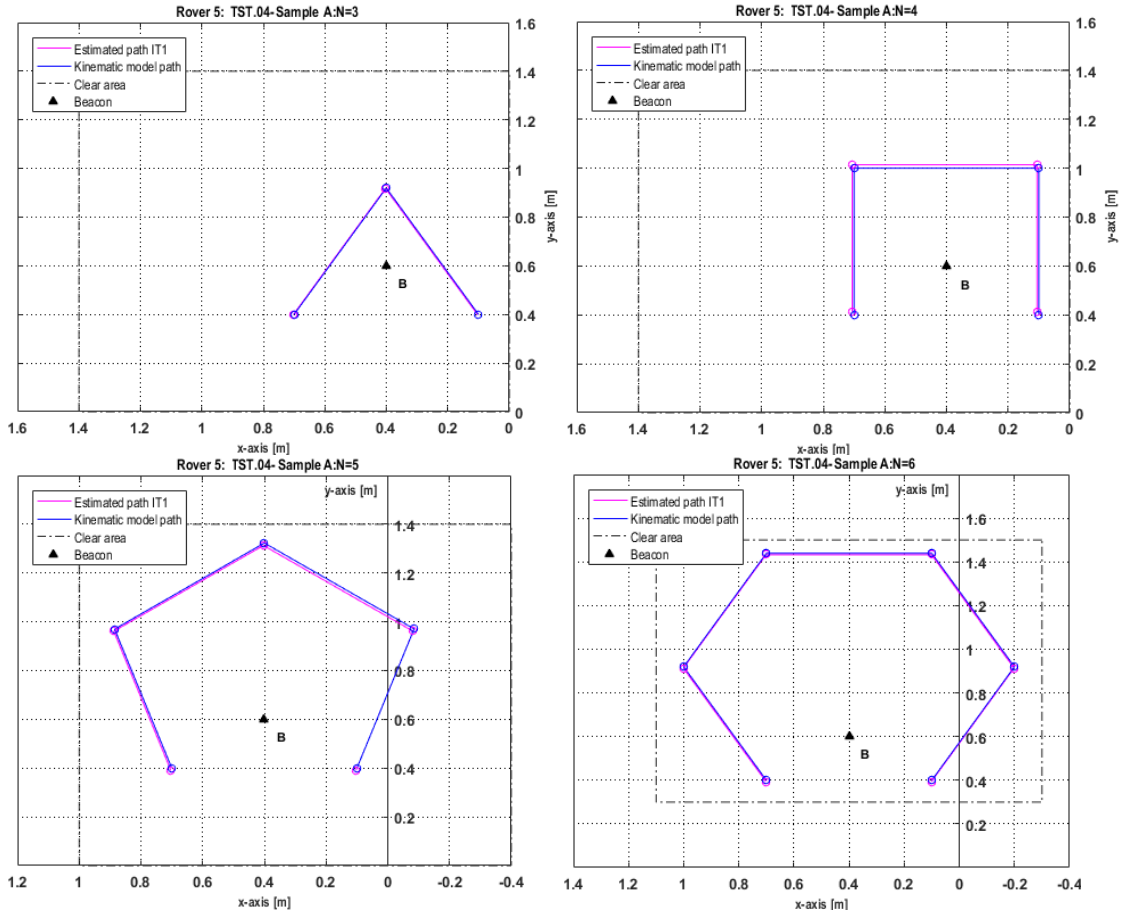

Figure 36. TST.04- Sample A: Estimated path vs. Kinematic model path for N=3, 4, 5 and 6

According to Figure 36, if the number of scans increases, the outcomes will improve. It means that the entries outside the main diagonal of the covariance matrix are closer to zero and the variances of x and y axes - main diagonal values- are smaller on each new measurement. Lastly, it is compared the experimental and theoretical results using the Fisher Information Matrix and its determinant (see Table 11).

As can be seen in Figure 37, the FIM determinant value becomes approximately stable after a concrete number of range measurements. This value is compared with the Cramer Rao Lower Bound to determine the quality of the estimation. In addition, the normalized FIM components are depicted in Figure 38. The results are normalized for a better understanding.

		$N=3$	$N=4$	$N=5$	$N=6$
Experimental	Cov matrix $\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}$ (10^{-6})	$\begin{pmatrix}1.15 & -0.005\\ -0.005 & 0.98\end{pmatrix}$	$\begin{pmatrix}0.745 & 0.006\\ 0.006 & 0.851\end{pmatrix}$	$\begin{pmatrix}0.604 & -0.0008\\ -0.0008 & 0.671\end{pmatrix}$	$\begin{pmatrix}0.503 & 0.0006\\ 0.0006 & 0.559\end{pmatrix}$
	FIM (10^6)	$\begin{pmatrix}0.868 & 0.004\\ 0.004 & 1.020\end{pmatrix}$	$\begin{pmatrix}1.343 & -0.009\\ -0.009 & 1.175\end{pmatrix}$	$\begin{pmatrix}1.657 & 0.002\\ 0.002 & 1.491\end{pmatrix}$	$\begin{pmatrix}1.988 & -0.002\\ -0.002 & 1.788\end{pmatrix}$
	$\lvert FIM\rvert$	$8.858\cdot 10^{11}$	$1.580\cdot 10^{12}$	$2.470\cdot 10^{12}$	$3.560\cdot 10^{12}$
Theoretical	FIM (10^6)	$\begin{pmatrix}0.944 & 0\\ 0 & 0.944\end{pmatrix}$	$\begin{pmatrix}1.259 & 0\\ 0 & 1.259\end{pmatrix}$	$\begin{pmatrix}1.574 & 0\\ 0 & 1.574\end{pmatrix}$	$\begin{pmatrix}1.888 & 0\\ 0 & 1.888\end{pmatrix}$
	$\lvert FIM\rvert$	$8.916\cdot 10^{11}$	$1.585\cdot 10^{12}$	$2.476\cdot 10^{12}$	$3.566\cdot 10^{12}$
|FIM| Normalized		0.993	0.995	0.997	0.997

Table 11. TST.04- Sample A: Covariance matrix, FIM and FIM determinant

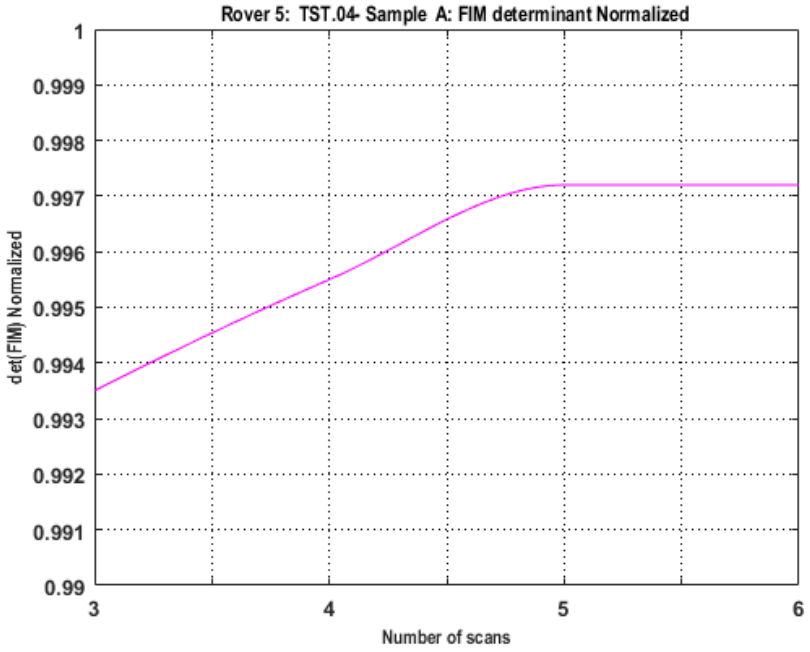

Figure 37. TST.04- Sample A: Determinant of normalized FIM

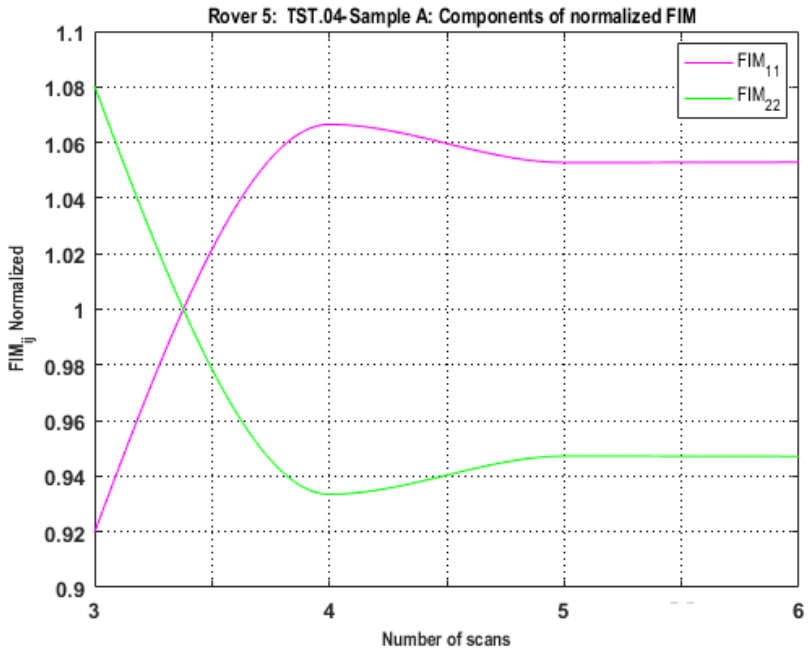

Figure 38. TST.04- Sample A: Main diagonal components of normalized FIM

7.2.2.3.2. TST.04- Sample A: Conclusions

According to TST.04 outcomes, it is possible to collect accurate positioning results despite the number of range measurements, only if they are performed in optimal points. For example, taking into account $N = 3$ and assuming that the scans are performed in optimal positions based on polygon vertices, the absolute error is 0.003 m on x-axis and 0.001 m on y-axis. This result can be compared with the absolute error obtained for the double number of scans ($N = 6$) made in arbitrary locations, which is 0.017 m on x-axis and 0.007 m on y-axis. Consequently, it is concluded that the absolute error is higher in the second case ($N = 6$), even though it is performed the double number of range measurements but in arbitrary points.

Additionally, the absolute error gathered for $N = 3$ is compared with the absolute error obtained with three beacons and three ultrasonic modules (see section 7.1.1) (0.054 m on x-axis and 0.037 m on y-axis), concluding that once again the absolute error on x-axis is higher than on y-axis, but it has been reduced on both axes even though it is used a single stationary beacon and one ultrasonic sensor module.

This can also be seen at the covariance matrix. When the range measurements are taken from optimal points, the entries outside the main diagonal are close to cero and the values of the main diagonal elements are really low. It indicates that the x and y variances –regarding a mean value- are small and the measurements are quite independent.

Additionally, the FIM determinant value becomes more or less stable at 0.997 for five scans ($N = 5$). Comparing this value with the Cramer Rao Lower Bound, it is

concluded that the estimation performed at the optimal point is not away from the ideal case, which is equal to 1. Therefore, it is figured out that the methodology adopted try to maximize the FIM determinant, which means that it is an effective method to get the robot position in spite of the performances of the technologies used to execute the range measurements.

Lastly, it is noted that the FIM matrix diagonal components become stable around a fix value, whereas the components outside the main diagonal are approximately zero, which is consistent with the theoretical findings.

7.2.2.3.3. TST.04- Sample B: Optimal positions- Symmetric points

In this case, a couple of tests are done based on mobile robot range measurements at symmetric points (see section 5.1.2.3).

Initially, the robot is placed at $(0.700, 0.400)$ and 0° oriented. The tests are performed considering $N = 3$ and $N = 4$, and it is made one iteration per test (see Figure 39). Finally, it is calculated the covariance matrix, the FIM and the FIM determinant.

Table 12 contains robot linear and angular displacements according to the number of scans and the symmetric positions, which are 0°, 60° and 120° for $N = 3$, and 0°, 45°, 135° and 180° for $N = 4$. Each of the above mentioned angles corresponds to the one formed by the vector connecting the beacon with robot position on each range measurement.

	N = 3		N = 4	
	$(\Delta x, \Delta y)$ [m]	$\Delta\theta$ [°]	$(\Delta x, \Delta y)$ [m]	$\Delta\theta$ [°]
1→2	(0.30,0.52)	60	$(0.42, 0.42)$	45
2→3	$(-0.30, 0.52)$	60	$(-0.42, 0.42)$	90
3→4			$(-0.6,0)$	45

Table 12. TST.04- Case B: Robot linear and angular motions

According to Figure 39, the estimated position absolute error increases with regard to the results obtained at the previous section (see Figure 36).

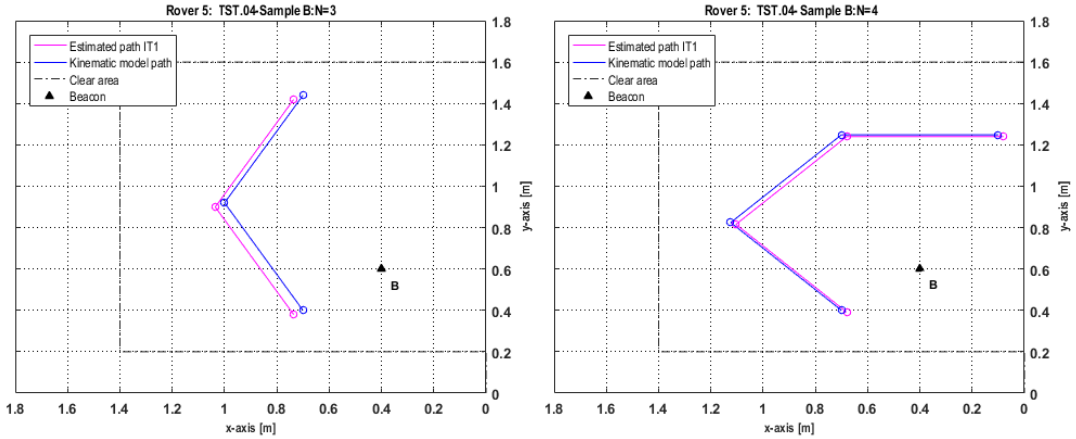

Figure 39. TST.04- Sample B: Covariance matrix, FIM and FIM determinant

In addition, Table 13 contains the Covariance matrix, FIM and |FIM| numerical results.

		$N = 3$	$N = 4$
Experimental	Cov matrix $\begin{pmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{pmatrix}$ (10^{-6})	$\begin{pmatrix}1.040 & 0.198\\ 0.198 & 1.161\end{pmatrix}$	$\begin{pmatrix}1.208 & 0.145\\ 0.145 & 1.311\end{pmatrix}$
	FIM (10^6)	$\begin{pmatrix}0.997 & -0.170\\ -0.170 & 0.890\end{pmatrix}$	$\begin{pmatrix}1.207 & 0.146\\ 0.146 & 1.310\end{pmatrix}$
	$\lvert FIM\rvert$	$8.598 \cdot 10^{11}$	$1.561 \cdot 10^{12}$
Theoretical	FIM (10^6)	$\begin{pmatrix}0.944 & 0\\ 0 & 0.944\end{pmatrix}$	$\begin{pmatrix}1.259 & 0\\ 0 & 1.259\end{pmatrix}$
	$\lvert FIM\rvert$	$8.916 \cdot 10^{11}$	$1.585 \cdot 10^{12}$
|FIM| Normalized		0.964	0.985

Table 13. TST.04- Sample B: Robot linear and angular motions

7.2.2.3.4. TST.04- Sample B: Conclusions

The energy required to estimate the robot position is minimized when the mobile robot performs the range measurement in symmetric points. However, it has been observed that the absolute error of the estimated position is affected. For instance, considering $N = 3$ and symmetric points, the absolute error is 0.036 m on x-axis and 0.020 m on y-axis, whereas this is 0.003 m on x-axis and 0.001 m on y-axis when it is assumed $N = 3$ optimal positions based on polygon vertices. In any case, the absolute error obtained is still lower than the one got with three arbitrary positions or using three stationary beacons and three ultrasonic sensor modules.

When it comes to the estimation quality, the FIM determinant indicates that the estimation quality has been degraded using optimal positions based on symmetric points instead of polygon vertices.

To sum up, the robot position estimation using optimal positions based on symmetric points requires less energy and time because the robot does not need to move around the stationary beacon. Conversely, the results are less accurate than the ones obtained with optimal positions based on polygon vertices.

7.2.2.4. TST.04 Conclusions

See section 7.2.2.3.2 and section 7.2.2.3.4.

7.2.3. Test 05 (TST.05): Iterative Estimation with Non-Optimal and Optimal Range Measurement Positions

7.2.3.1. TST.05 Description

The main goal of this test is to estimate the robot location with a stable error, combining range measurements performed in optimal and non-optimal positions.

It is used same robot sensor configuration than in TST.04 (see Figure 8). Consequently, the Rover 5 includes a single HC-SR04 ultrasonic module mounted over a rotary platform, which is moved by the SM-S2309S servomotor. Then, it is always computed one distance measurement per each degree scanned, up to 180°.

In this case, the scenario dimensions are limited to $1.800\ x\ 2.500$ m^2. The origin of the inertial frame is located at $(0,0)$, which corresponds to the centre-right of the scenario previously defined. A stationary beacon is placed arbitrarily in this area, which position is $B(1.000, 0.600)$. For simplicity, the y-axis is reversed due to the servomotor operating mode.

To perform the test, the robot is positioned somewhere taking into account the above mentioned scenario dimensions. After that, the robot performs three scans in three arbitrary non-collinear points, and it estimates its position.

Once the initial robot position is known –even though it has a degree of uncertainty-, it is planned a new trajectory through N optimal points. Then, Rover 5 performs N scans at the pre-defined optimal points and estimates again its position. This procedure is repeated several times until the position absolute error is stable.

7.2.3.2. TST.05 Algorithm

In this section it is detailed the algorithm used to carry out the test 05.

Algorithm TST.05

Given a single stationary beacon with known coordinates, $B(x_b, y_b)$:

1) Make the robot carries out a 180° CW scan at its current position, $p_i(x_i, y_i)$.
2) Compute the range measurement (d_k) using the TOA ranging technique. It is considered the ultrasound waves travel at the speed of sound, approximately $340\ m/s$ or $0.00034\ m/\mu s$.

$$d_k = \frac{t_k \cdot c}{2};\quad k = 1,2,3 \dots M;\quad M = 180$$

3) Verify distance measurements according to the scenario dimensions and the minimum distance allowed to any beacon (eliminate false-positive):

$$d_{min} < d_k \leq d_{max}$$

4) Identify the interval of interesting distances according to the effectual angle of the ultrasonic sensor. The result is a vector, $\boldsymbol{dm}$, including few distance measurements:

$$\{Int_o, Int_f\};\ \boldsymbol{dm} = (d_{Int_o} : d_{Int_f})$$

5) Compute the arithmetic mean of the $\boldsymbol{dm}$ vector:

$$dist_i = \overline{dm} = \frac{\sum_{d=1}^{n} d_d}{n}$$

6) Make the robot performs a pre-defined angular and linear displacement:

$$\{\Delta\theta, \Delta x, \Delta y\}$$

7) Estimate new robot position considering previous position and the robot displacement:

$$p_i\{x_{i-1} + \Delta x_i\ , y_{i-1} + \Delta y_i\}$$

8) Before (9), repeat from (1) to (8) until $i = N$; where $N = 3$ is the total number of scans performed in non-optimal positions.

9) Compute b matrix:

$$b = \begin{bmatrix} d_1^2 - \Delta x_1^2 + 2\Delta x_1 x_b - \Delta y_1^2 + 2\Delta y_1 y_b - d_3^2 + \Delta x_3^2 - 2\Delta x_3 x_b + \Delta y_3^2 - 2\Delta y_3 y_b \\ d_2^2 - \Delta x_2^2 + 2\Delta x_2 x_b - \Delta y_2^2 + 2\Delta y_2 y_b - d_3^2 + \Delta x_3^2 - 2\Delta x_3 x_b + \Delta y_3^2 - 2\Delta y_3 y_b \end{bmatrix}$$

10) Compute A matrix:

$$A = -2 \cdot \begin{bmatrix} \Delta x_1 - \Delta x_3 & \Delta y_1 - \Delta y_3 \\ \Delta x_2 - \Delta x_3 & \Delta y_2 - \Delta y_3 \end{bmatrix}$$

11) Multiply matrices to get robot position:

$$p_i = (A^T A)^{-1} A^T \cdot b$$

---

12) Plan a new robot trajectory according to its estimated position p_i and depending on the number of range measurements in optimal positions.
13) Repeat from (1) to (7) until $j = K$; where K is the total number of scans performed in optimal positions.
14) Compute b matrix:

$$b = \begin{bmatrix} d_j^2 - \Delta x_j^2 + 2\Delta x_j x_b - \Delta y_j^2 + 2\Delta y_i y_b - d_N^2 + \Delta x_N^2 - 2\Delta x_N x_b + \Delta y_N^2 - 2\Delta y_N y_b \\ \vdots \\ d_{N-1}^2 - \Delta x_{N-1}^2 + 2\Delta x_{N-1} x_b - \Delta y_{N-1}^2 + 2\Delta y_{N-1} y_b - d_N^2 + \Delta x_N^2 - 2\Delta x_N x_b + \Delta y_N^2 - 2\Delta y_N y_b \end{bmatrix};$$

$$j = 1,2,3 \dots K$$

15) Compute A matrix:

$$A = -2 \cdot \begin{bmatrix} \Delta x_j - \Delta x_N & \Delta y_j - \Delta y_N \\ \vdots & \vdots \\ \Delta x_{N-1} - \Delta x_N & \Delta y_{N-1} - \Delta y_N \end{bmatrix}; j = 1,2,3 \dots K$$

16) Compute the inverse and the transpose of the A matrix.
17) Multiply matrices to get the robot position:

$$p_j = (A^T A)^{-1} A^T \cdot b$$

18) Repeat from (12) until the estimated position absolute error becomes stable.

7.2.3.3. TST.05 Results

The initial robot position estimated from three arbitrary scanning points is $(0.423, 0.167)$. Figure 40 shows the robot estimated path and the same path computed using the kinematic robot model (see section 3.1).

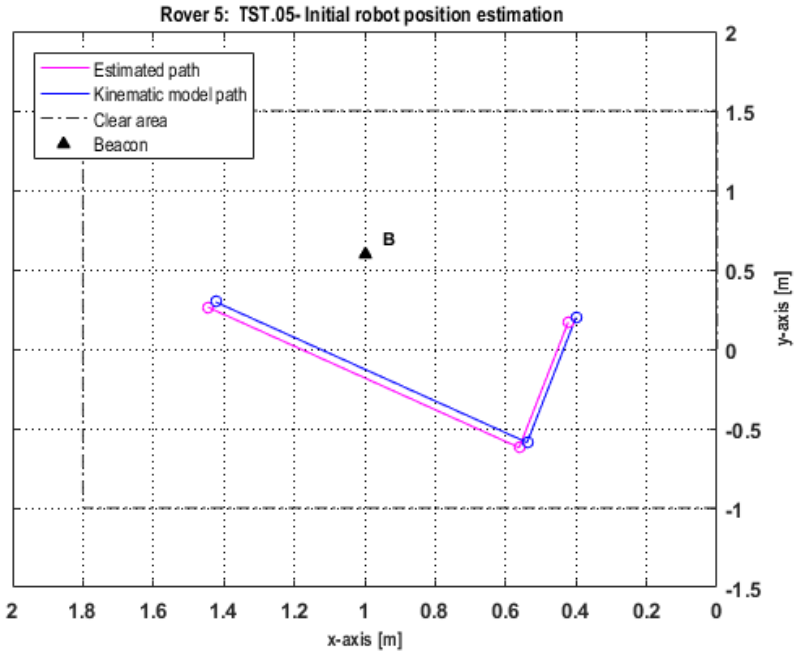

Figure 40. TST.05- Initial estimated path vs. Initial kinematic model path

As can be deduced from Figure 40, the absolute error is 0.023 m on x-axis and 0.033 m on y-axis. This absolute error is lower than the one obtained in the similar case analysed in section 7.2.1. The reason for this is that the arbitrary scan positions are probably closer to the optimal points in this case. Moreover, it is possible to verify that the absolute errors on both axes are also affected. Contrary to previous tests, the absolute error on y-axis is higher than on x-axis, which is possibly due to the scanning points are located in positive x half-plane in TST.05, whereas they are located positive y half-plane in the previous tests.

7.2.3.3.1. TST.05 - Sample A: N=3

In this case, it is computed a new robot position considering $N = 3$ optimal scanning points, and taking into account the initial robot position estimated previously. It is performed seven iterations and the Δx, Δy and $\Delta\theta$ performed by the robot in every iteration are summarized in Table 14. Additionally, Figure 41 shows the estimated path and the robot real trajectory obtained using the robot kinematic model.

	1→2			*2→3*		
	Δx [m]	$\Delta y\ [m]$	$\Delta\theta\ [°]$	$\Delta x\ [m]$	$\Delta y\ [m]$	$\Delta\theta\ [°]$
$IT1$	−0.450	0.779	120	−0.450	−0.779	120
$IT2$	0.450	−0.779	120	0.450	0.779	120
$IT3$	−0.450	0.779	120	−0.450	−0.779	120
$IT4$	0.450	−0.779	120	0.450	0.779	120
$IT5$	−0.450	0.779	120	−0.450	−0.779	120
$IT6$	0.450	−0.779	120	0.450	0.779	120
$IT7$	−0.450	0.779	120	−0.450	−0.779	120

Table 14. TST.05- Sample A: N=3; Robot linear and angular displacements

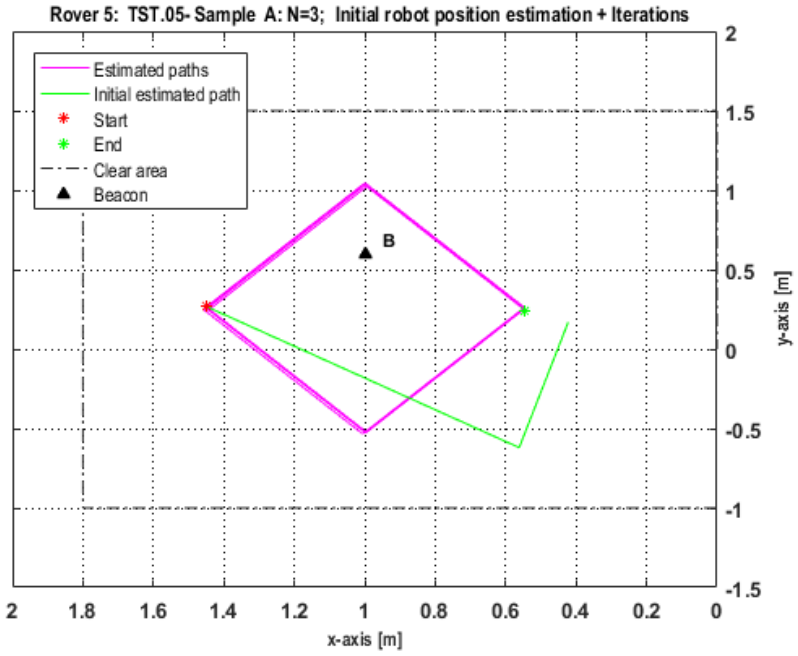

Figure 41. TST.05- Sample A: N=3; Initial estimated path and optimal measurements (7 IT)

As shown in Figure 41, initially it is considered three different optimal points separated 120°, which are equivalent to an upper equilateral triangle. Then, it is considered same symmetric positions, also separated 120° and forming lower equilateral triangle. The reason for that is to make the robot trajectory as simple as possible according to robot motion capabilities.

For each of the abovementioned iterations, it is calculated the absolute error on both axes (see Figure 42).

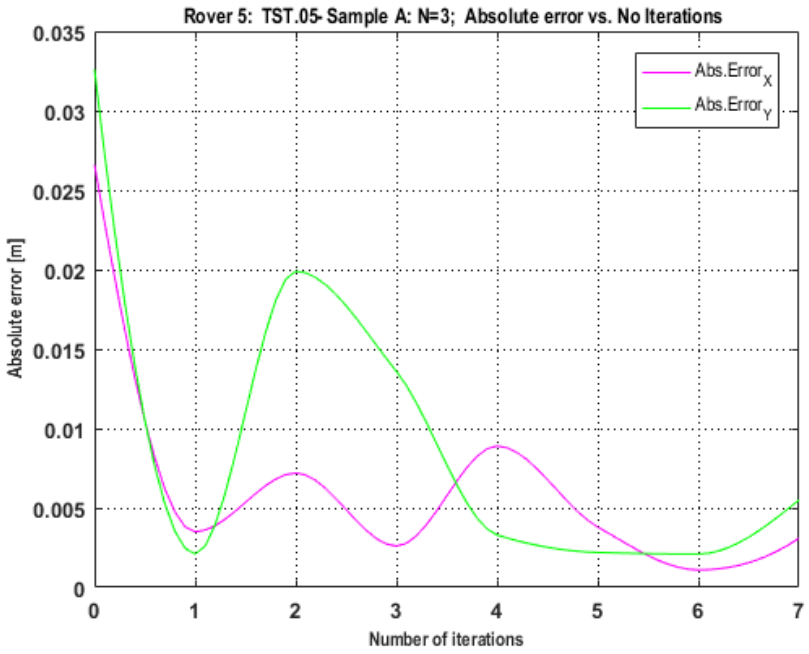

Figure 42. TST.05- Sample A: N=3; XY axes Absolute error

As can be deduced from Figure 42, the absolute error is higher on x- axis than on y-axis. It is possibly due to the uncertainty of the initial robot position. In this case, $N = 3$, the absolute error on y-axis comes to be stable after 4 iterations, and the stabilized value is around 0.003 m. However, the absolute error on x-axis never becomes stable.

7.2.3.3.2. TST.05 – Sample B: N=4

It is performed a second test considering same initial robot position and $N = 4$ range measurements performed in optimal positions. Once again, it is made seven iterations. The mobile robot linear and angular robot motions are gathered in Table 15.

	1→2			*2→3*			*3→4*		
	Δx *[m]*	Δy *[m]*	$\Delta\theta$ *[°]*	Δx *[m]*	Δy *[m]*	$\Delta\theta$ *[°]*	Δx *[m]*	Δy *[m]*	$\Delta\theta$ *[°]*
$IT1$	0.000	0.800	90	−0.800	0.000	90	0.000	−0.800	90
$IT2$	0.000	−0.800	90	0.800	0.000	90	0.000	0.800	90
$IT3$	0.000	0.800	90	−0.800	0.000	90	0.000	−0.800	90
$IT4$	0.000	−0.800	90	0.800	0.000	90	0.000	0.800	90
$IT5$	0.000	0.800	90	−0.800	0.000	90	0.000	−0.800	90
$IT6$	0.000	−0.800	90	0.800	0.000	90	0.000	0.800	90
$IT7$	0.000	0.800	90	−0.800	0.000	90	0.000	−0.800	90

Table 15. TST.05- Sample B: N=4; Robot linear and angular displacements

The estimated robot path and the absolute error on both axes are shown in Figure 43 and Figure 44, respectively.

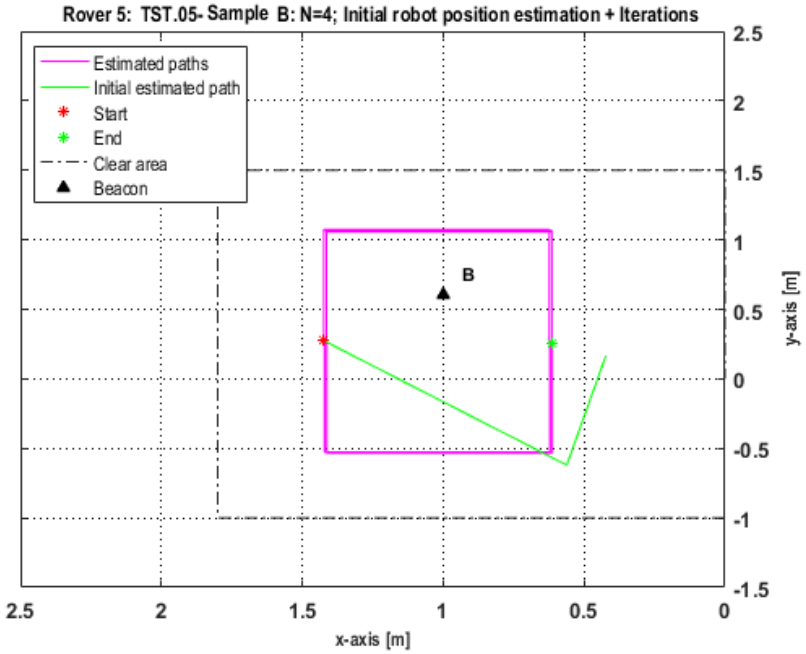

Figure 43. TST.05- Sample B: N=4; Initial estimated and optimal measurements (7 IT)

The robot trajectory is based on four optimal points, which angle between the beacon and the robot position is increasing 90° each time (see Figure 43). Consequently, the robot performs a trajectory consisting of a square and then, it moves through same symmetric positions, also separated 90°, and forming lower square. The reason for that is to make the robot trajectory as simple as possible according to robot motion capabilities.

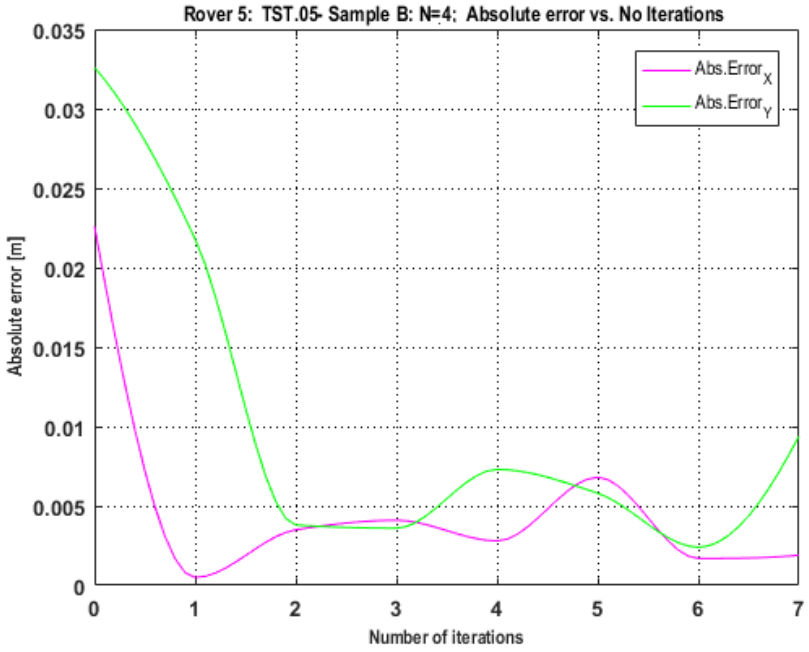

Figure 44. TST.05- Sample B: N=4; XY axes Absolute error

From Figure 44, it is deduced that the absolute error becomes stable after two iterations,

which means that it is required less iterations to get a stable absolute error compare to the previous results (see Figure 42). Additionally, the stabilized value is approximately 0.0028 m on x-axis and 0.005 m on y-axis.

7.2.3.3.3. TST.05 - Sample C: N=5

In this case, same test is repeated considering identical initial robot estimated position and a total number of scans equal to five ($N = 5$). As in previous tests, it is also carried out seven iterations. In Table 16 it is collected all the displacements performed by the mobile robot and in Figure 45 it is possible to compare the estimate robot path and the real robot path computed using the kinematic robot model.

	1→2			*2→3*			*3→4*			*4→5*		
	Δx $[m]$	Δy $[m]$	$\Delta\theta$ $[°]$	Δx $[m]$	Δy $[m]$	$\Delta\theta$ $[°]$	Δx $[m]$	Δy $[m]$	$\Delta\theta$ $[°]$	Δx $[m]$	Δy $[m]$	$\Delta\theta$ $[°]$
$IT1$	0.247	0.761	72	−0.647	0.470	72	−0.647	−0.472	72	0.247	−0.761	72
$IT2$	−0.247	−0.761	72	0.647	−0.470	72	0.647	0.472	72	−0.247	0.761	72
$IT3$	0.247	0.761	72	−0.647	0.470	72	−0.647	−0.472	72	0.247	−0.761	72
$IT4$	−0.247	−0.761	72	0.647	−0.470	72	0.647	0.472	72	−0.247	0.761	72
$IT5$	0.247	0.761	72	−0.647	0.470	72	−0.647	−0.472	72	0.247	−0.761	72
$IT6$	−0.247	−0.761	72	0.647	−0.470	72	0.647	0.472	72	−0.247	0.761	72
$IT7$	0.247	0.761	72	−0.647	0.470	72	−0.647	−0.472	72	0.247	−0.761	72

Table 16. TST.05- Sample C: N=5; Robot linear and angular displacements

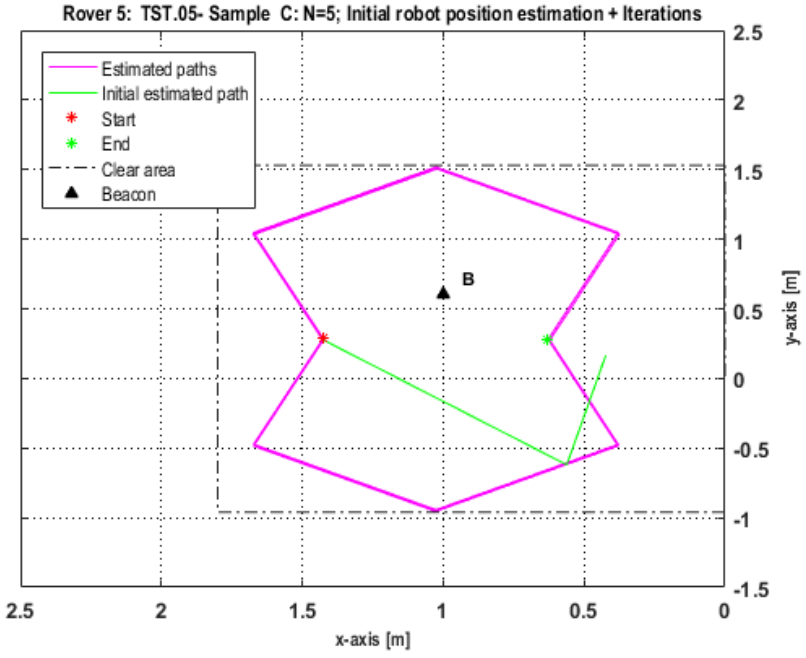

Figure 45. TST.05- Sample C: N=5; Initial estimated path and optimal measurements (7 IT)

Again, the robot carries out a trajectory based on five optimal points followed by same symmetric trajectory. In this case, the angle between each optimal position increases 72°; therefore, the robot follows and upper pentagon trajectory and a lower pentagon trajectory. The reason for that is to make the robot path as simple as possible according to robot motion capabilities.

Lastly, it is also computed the absolute error in both axes for all the iterations performed (see Figure 46), concluding that for $N = 5$, the absolute error turns into a stable value after a couple of iterations. The stabilized value is approximately $2.4 \cdot 10^{-3}$ m on x-axis and $7.4 \cdot 10^{-3}$ on y-axis.

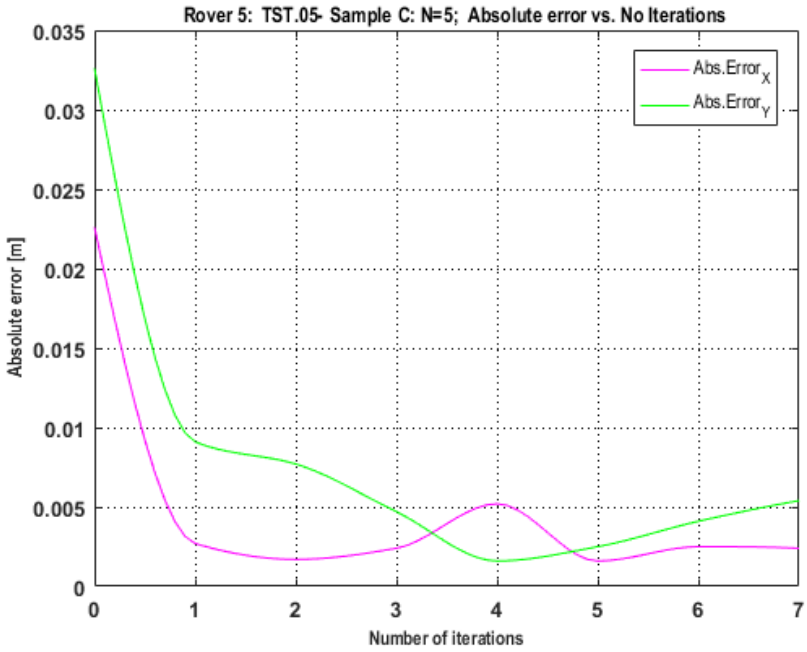

Figure 46. TST.05- Sample C: XY axes Absolute error

7.2.3.4. TST.05 Conclusions

According to the previous results, it is deduced that the estimated position absolute error decreases and becomes stable early when the number of scans increases. For example, for $N = 3$, the absolute error on x-axis is 0.003 m, whereas this is 0.0024 m when $N = 5$. Nevertheless, if the number of scans increases the absolute error on y-axis also increases. This result is the opposite of what is expected, and it is probably due to the location of the optimal points selected.

Furthermore, it is also noted that there is no large results improvement between $N = 4$ and $N = 5$, which means that from a specific number of range measurements and forward the benefit is trivial compared to the time required to estimate the robot position.

Finally, it is verified that the reason of getting absolute error higher in one axis than in other is probably due to the distribution of the optimal points regarding the reference axes selected.

7.2.4. Test 06 (TST.06): Re-planning of Optimal Points

7.2.4.1. TST.06 Description

The aim of TST.06 is to estimate the robot location with a stable error from an uncertain robot estimated position and re-planning the optimal robot trajectory after each range measurement.

This test requires same robot sensor configuration than the one used in TST.05.

In this case, the scenario dimensions are limited to $2.500\ x\ 2.500$ m^2, in order to avoid false-positive detected by the ultrasonic sensors. The origin of the inertial frame is located at $(0,0)$, which corresponds to the centre-right of the scenario previously defined. A stationary beacon is placed arbitrarily in this area, which position is $B(1.000,\ 0.600)$. For simplicity, the y-axis is reversed due to the servomotor operating mode.

Initially, it is estimated the robot position using three range measurements got from three range measurements in arbitrary points. The result is an uncertain robot position that it is used as a reference for the robot trajectory planning. This is same procedure than the one performed in TST.05.

Once the initial robot position is known, it is planned a new trajectory through $N = 3$ optimal points. When the robot reaches the first optimal point and performs the scan, it is estimated a new robot trajectory considering the new range measurement and two previous measurements. This procedure is repeated up to seven times.

7.2.4.2. TST.06 Algorithm

In this section it is detailed the algorithm used to carry out the test 06.

Algorithm TST.06

Given a single stationary beacon with known coordinates, $B(x_b, y_b)$:

1) Make the robot carries out a 180° CW scan at its current position, $p_i(x_i, y_i)$.
2) Compute the range measurement (d_k) using the TOA ranging technique. It is considered the ultrasound waves travel at the speed of sound, approximately $340\ m/s$ or $0.00034\ m/\mu s$.

$$d_k = \frac{t_k \cdot c}{2};\quad k = 1,2,3 \dots M;\quad M = 180$$

3) Verify distance measurements according to the scenario dimensions and the minimum distance allowed to any beacon (eliminate false-positive):

$$d_{min} < d_k \leq d_{max}$$

4) Identify the interval of interesting distances according to the effectual angle of the ultrasonic sensor. The result is a vector, $\boldsymbol{dm}$, including few distance measurements:

$$\{Int_o, Int_f\};\ \boldsymbol{dm} = (d_{Int_o} : d_{Int_f})$$

5) Compute the arithmetic mean of the $\boldsymbol{dm}$ vector:

$$dist_i = \overline{dm} = \frac{\sum_{d=1}^{n} d_d}{n}$$

6) Make the robot performs a pre-defined angular and linear displacement:

$$\{\Delta\theta, \Delta x, \Delta y\}$$

7) Estimate new robot position considering previous position and the robot displacement:

$$p_i\{x_{i-1} + \Delta x_i\ , y_{i-1} + \Delta y_i\}$$

8) Before (9), repeat from (1) to (8) until $i = N$; where $N = 3$ is the total number of scans performed in non-optimal positions.

9) Compute b matrix:

$$b = \begin{bmatrix} d_1^2 - \Delta x_1^2 + 2\Delta x_1 x_b - \Delta y_1^2 + 2\Delta y_1 y_b - d_3^2 + \Delta x_3^2 - 2\Delta x_3 x_b + \Delta y_3^2 - 2\Delta y_3 y_b \\ d_2^2 - \Delta x_2^2 + 2\Delta x_2 x_b - \Delta y_2^2 + 2\Delta y_2 y_b - d_3^2 + \Delta x_3^2 - 2\Delta x_3 x_b + \Delta y_3^2 - 2\Delta y_3 y_b \end{bmatrix}$$

10) Compute A matrix:

$$A = -2 \cdot \begin{bmatrix} \Delta x_1 - \Delta x_3 & \Delta y_1 - \Delta y_3 \\ \Delta x_2 - \Delta x_3 & \Delta y_2 - \Delta y_3 \end{bmatrix}$$

11) Multiply matrices to get robot position:

$$p_i = (A^T A)^{-1} A^T \cdot b$$

12) Plan a new robot trajectory according to its estimated position p_i and taking into account the total number of scans ($K = 3$) required in optimal positions.
13) Make the robot performs the pre-defined angular and linear displacement:

$$\{\Delta\theta, \Delta x, \Delta y\}$$

14) Estimate new robot position considering previous position and the robot displacement:

$$p_i\{x_{i-1} + \Delta x_i\ , y_{i-1} + \Delta y_i\}$$

15) Repeat from (1) to (5)
16) Repeat from (9) to (11) taking into account the new range measurement and two old measurements.
17) Repeat from (12) to (16) until the absolute error has been reduced.

7.2.4.3. TST.06 Results

Initially, the robot is placed somewhere within the defined area. Then, the robot performs three scans in three different arbitrary positions. After that, it is estimated the robot initial position, which is $(0.434, 0.359)$ in this case. In Figure 47 it is possible to compare the initial robot estimated path and the kinematic robot model path.

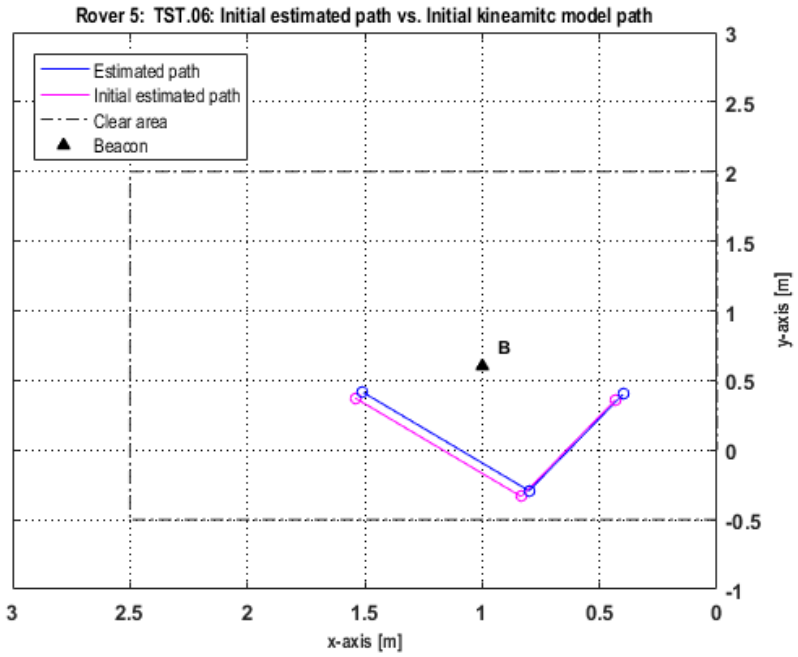

Figure 47. TST.06- Initial estimated path vs. Initial kinematic model path

As shown in Figure 47, the absolute error is 0.034 m on x-axis and 0.041 m on y-axis. In this case, the absolute error is higher than the one obtained at TST.05. Furthermore, it is possible to verify that the absolute error is higher on y-axis than on x-axis. Once again it is possibly due to the location of the arbitrary points regarding the references axes selected. In addition, Table 17 collects the robot angular and linear displacements.

	1→2			*2→3*		
	Δx [m]	Δy [m]	Δθ [°]	Δx [m]	Δy [m]	Δθ [°]
$IT1$	0.400	-0.693	300	0.707	0.707	45
$IT2$	0.707	0.707	45	0.600	0.000	-45
$IT3$	0.600	0.000	-45	-0.875	1.515	120
$IT4$	-0.875	1.515	120	-0.875	-1.515	120
$IT5$	-0.875	-1.515	120	0.400	-0.693	180
$IT6$	0.400	-0.693	180	0.400	0.693	60
$IT7$	0.400	0.693	60	-0.400	0.693	60

Table 17. TST.06- Robot linear and angular displacements

As seen in Table 17, the robot does the same displacements between iterations because on each iteration it is taken into account a couple of previous estimated points. For instance, the IT2 (1→ 2) considers the same displacements than the ones performed in IT1 (2→3).

Figure 48 shows the trajectory followed by the robot, as well as its estimated trajectory.

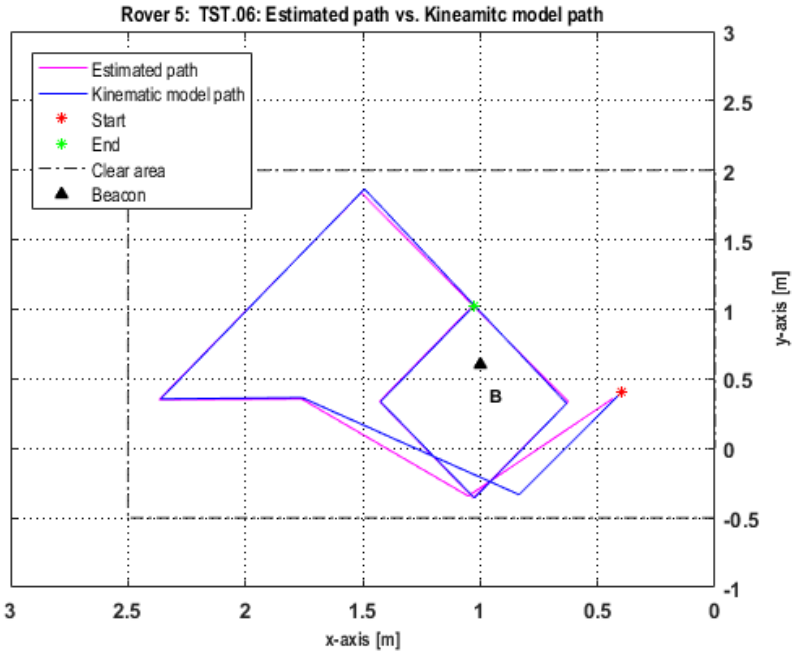

Figure 48. TST.06- Real robot trajectory vs. Estimated trajectory (7 IT)

Lastly, Figure 49 shows the absolute error against the number of iterations on both axes.

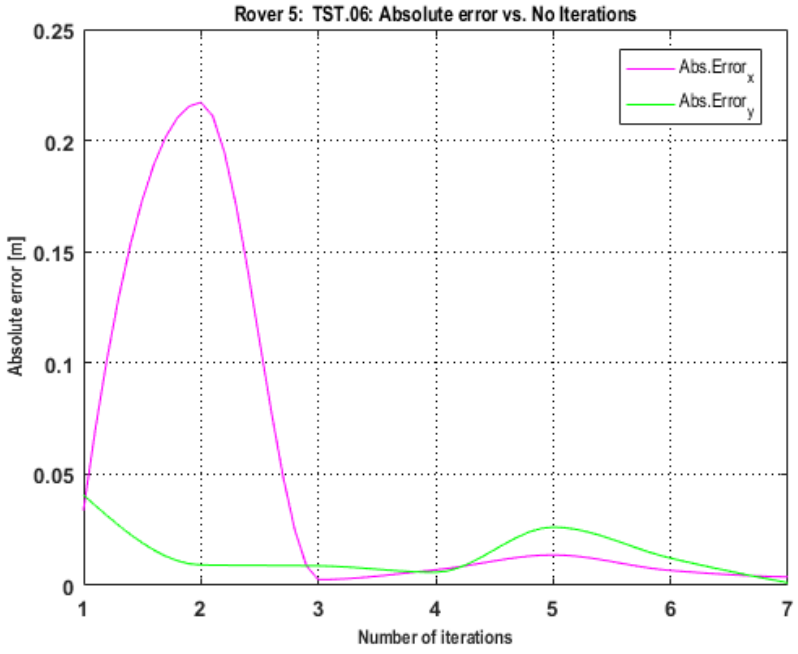

Figure 49. TST.06- XY axes Absolute error

7.2.4.4. TST.06 Conclusions

In conclusion, the absolute error decreases every new iteration (see Figure 49). For example, the absolute error on y-axis is reduced from 0.0406 m to 0.0012 m. The reason for this is that initially the robot position is estimated using range measurements performed in arbitrary positions. As soon as the robot position is calculated with range measurements made in optimal points, the absolute error on both axes starts to decrease until the minimum achievable with the technology used.

7.3. CASE 03: 2D Single Target Positioning Using Triangulation And A Single Beacon

In this case, it is done a couple of experimental tests based on solving the mobile robot position by triangulation. Both tests consider a single beacon and a unique ultrasonic sensor module, which is placed over a rotating platform. The main difference between the tests is the position where the robot scans its surroundings, which are arbitrary or optimal positions depending on the test performed. Table 18 contains a brief description of each test performed in case 03.

Test	*Scanning positions*	*Sensors*			*Rover 5 status when sensor is reading*
		Qty.	*Status*	*Type*	
TST.07	Arbitrary	1	Rotating	SR-HC04	Stationary
TST.08	Optimal	1	Rotating	SR-HC04	Stationary

Table 18. Case 03: Tests summary

7.3.1. Test 07 (TST.07): Bearing Measurements at Arbitrary Positions

7.3.1.1. TST.07 Description

A couple of goals are intended to be reached with TST.07. These are (1) to apply triangulation technique when it is considered a single beacon and one ultrasonic module and (2) to identify the main differences in the results with regard to the trilateration outcomes.

In this case, the Rover 5 sensor configuration is the same than the one used in TST.06 (see Figure 8). Consequently, there is one HC-SR04 ultrasonic module mounted over a rotary platform, which is driven by the SM-S2309S servomotor. This allows a 180° scanning CW.

The scenario dimensions are limited to $1.200\ x\ 1.000$ m^2. The origin of the inertial frame is located at $(0,0)$, which corresponds to the lower-right corner of the scenario previously defined. A stationary beacon is placed arbitrarily in this area, which position is $B(0.600, 0.600)$. For simplicity, the y-axis is reversed due to the servomotor operating mode.

Initially, the robot is placed somewhere taking into account the above mentioned scenario dimensions. Then, the robot keeps stationary while it performs a scan of its environment. After each scan, the robot performs the pre-defined angular and linear motion, in a way that robot orientation and position varies before next scan. In this case, it is considered that the robot performs a total of three scans ($N = 3$).

7.3.1.2. TST.07 Algorithm

In this section it is detailed the algorithm used to carry out the test 07.

Algorithm TST.07

Given a single stationary beacon with known coordinates, $B(x_b, y_b)$:

1) Make the robot carries out a 180° CW scan at its current position, $p_i(x_i, y_i)$.
2) Compute the range measurement (d_k) using the TOA ranging techniques. It is considered the ultrasound waves travel at the speed of sound, approximately $340\ m/s$ or $0.00034\ m/\mu s$.

$$d_k = \frac{t_k \cdot c}{2};\quad k = 1,2,3 \dots M;\quad M = 180$$

3) Verify distance measurements according to the scenario dimensions and the minimum distance allowed to any beacon (eliminate false-positive):

$$d_{min} < d_k \leq d_{max}$$

4) Identify the interval during which the beacon is recognised by the ultrasonic sensor. The result is a vector, $\boldsymbol{\phi_m}$, including few angles:

$$\{Int_o, Int_f\};\ \boldsymbol{\phi_m} = (\phi_{Int_o} : \phi_{Int_f})$$

where Int_o is the lower limit and Int_f is the upper limit of the above mentioned interval. Accordingly, the initial and final angle measurements of the $\boldsymbol{\phi_m}$ vector are ϕ_{Int_o}, and ϕ_{Int_f}, respectively

5) Compute the arithmetic mean of the $\boldsymbol{\phi_m}$ vector:

$$\overline{\phi_m} = \frac{\sum_{d=1}^{n} \phi_d}{n}$$

where n is the number of bearing measurements between the upper and lower interval limits

6) Make the robot performs a pre-defined angular and linear displacement:

$$\{\Delta\theta, \Delta x, \Delta y\}$$

7) Estimate new robot position considering previous position and robot displacements:

$$p_i\{x_{i-1} + \Delta x_i\ , y_{i-1} + \Delta y_i\}$$

8) Before (9), repeat from (1) to (8) until $i = N$. In this case, $N = 3$, then:

$$p_1(x_1, y_1);\quad p_2(x_2, y_2);\quad p_3(x_3, y_3)$$

$$x_1 = x;\ \ y_1 = y;\ x_2 = x + \Delta x_2;\ \ y_2 = y + \Delta y_2;\ x_3 = x + \Delta x_3;\ \ y_3 = y + \Delta y_3;$$

$$\phi_1; \phi_2; \phi_3$$

9) Compute the modified beacon coordinates (reference coordinate p_2)[1]:

$$x_1' = x_1 - x_2;\ y_1' = y_1 - y_2$$

$$x_3' = x_3 - x_2;\ y_3' = y_3 - y_2$$

10) Compute ϕ_{12}, ϕ_{23} and ϕ_{31}

$$\phi_{12} = \phi_1 - \phi_2; \quad \phi_{23} = \phi_2 - \phi_3; \quad \phi_{31} = \phi_3 - \phi_1$$

11) If $\phi_{31} = 0 \ \vee \ \phi_{31} = \pi$ continue in (18). If not continue in (12).

Only if: $\phi_{31} \neq 0 \ \vee \ \phi_{31} \neq \pi$

12) Compute the three cot(.):

$$T_{12} = \cot(\phi_2 - \phi_1);\ \ T_{23} = \cot(\phi_3 - \phi_2);\ \ T_{31} = \cot(\phi_1 - \phi_3)$$

13) Compute the modified circle centre coordinates:

$$x_{12}' = x_1' + T_{12}y_1'; \quad y_{12}' = y_1' - T_{12}x_1';$$
$$x_{23}' = x_3' - T_{23}y_3'; \quad y_{23}' = y_3' + T_{23}x_3';$$
$$x_{31}' = (x_3' + x_1') + T_{31}(y_3' - y_1'); \quad y_{31}' = (y_3' + y_1') - T_{31}(x_3' - x_1')$$

14) Compute $k_{ij}{}'$:

$$k_{12}' = 0; \quad k_{23}' = 0; \quad k_{31}' = x_1'x_3' + y_1'y_3' + T_{31}(x_1'y_3' - x_3'y_1')$$

15) Compute D (if $D = 0$ return with an error):

$$D = (x_{12}' - x_{23}')(y_{23}' - y_{31}') - (y_{12}' - y_{23}')(x_{23}' - x_{31}')$$

16) Apply Cramer's rule to solve the equations system:

$$x_2 = x_b - \frac{k_{31}'(y_{12}' - y_{23}')}{D}; \quad y_2 = y_b - \frac{k_{31}'(x_{23}' - x_{12}')}{D}$$

17) Compute (x_1, y_1) and (x_3, y_3)

Only if: $\phi_{31} = 0 \ \vee \ \phi_{31} = \pi$

18) Compute the three cot(.):

$$T_{12} = \cot(\phi_2 - \phi_1);\ \ T_{23} = \cot(\phi_3 - \phi_2);\ \ T_{31} = \cot(\phi_1 - \phi_3)$$

[1] *ToTal Algorithm*

19) Compute the modified circle centre coordinates:

$$x'_{12} = x'_1 + T_{12}y'_1; \quad y'_{12} = y'_1 - T_{12}x'_1;$$
$$x'_{23} = x'_3 + T_{12}y'_3; \quad y'_{23} = y'_3 - T_{12}x'_3;$$
$$x'_{31} = (y'_3 - y'_1); \quad y'_{31} = (x'_1 - x'_3)$$

20) Compute k'_{ij}:

$$k'_{31} = x'_1y'_3 - x'_3y'_1$$

21) Compute D (if $D = 0$ return with an error):

$$D = (x'_{23} - x'_{12})(y'_{31}) + (y'_{12} - y'_{23})(x'_{31})$$

22) Apply Cramer's rule to solve the equations system:

$$x_2 = x_b - \frac{k'_{31}(y'_{12} - y'_{23})}{D}; \quad y_2 = y_b - \frac{k'_{31}(x'_{23} - x'_{12})}{D}$$

23) Compute (x_1, y_1) and (x_3, y_3)

7.3.1.3. TST.07 Results

Initially, the robot performs three different scans in three arbitrary positions close to each other. The aim is to verify if close scanning points could affect the robot estimated position, as with trilateration. For that reason, the displacements performed by Rover 5 during this test are ($\Delta x_2 = 0.100$, $\Delta y_2 = 0.050$) and ($\Delta x_3 = 0.050$, $\Delta y_3 = 0.100$). It is done a couple of iterations.

The robot position estimated by triangulation is compared with the position estimated with the robot kinematic model (see Figure 50), verifying that closer scanning points increase the estimation error, as with trilateration technique. Again, the absolute error on x-axis (0.582 m) is higher than on y-axis (0.373 m), which is probably due to the scanning points selected are closer to each other on the x-axis. As seen in Figure 50, the difference between the results gathered on each iteration is trivial.

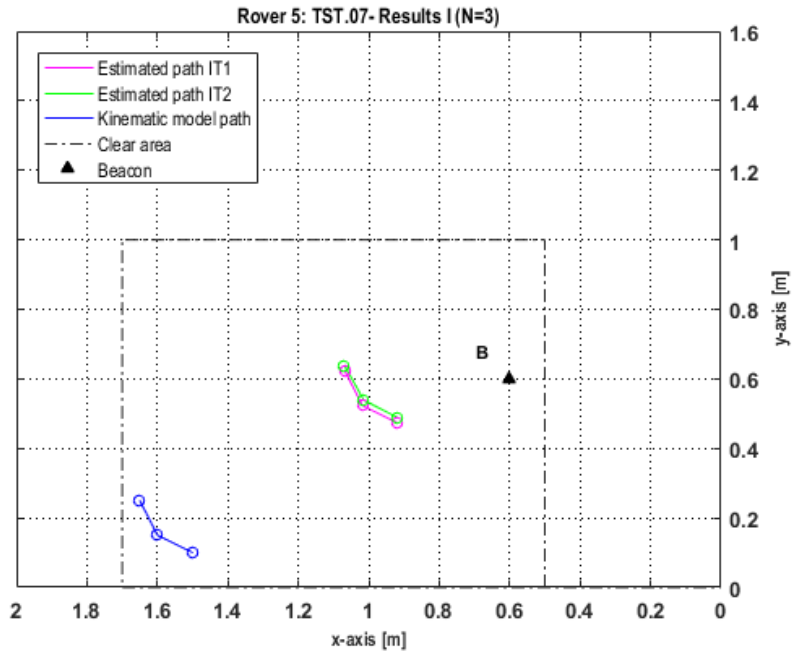

Figure 50. TST.07- Estimated path vs. Real path

Then, it is considered that the robot performs three different scans in three arbitrary non-collinear points with largest displacements (≥ 0.283 m) between scanning points, in order to avoid error increases due to closer scanning points. Consequently, the linear and angular robot motion between the scanning points are $(0.410, 0.287)$ from point 1 to point 2, and $(-0.383, 0.321)$ from point 2 to point 3. Moreover, the stationary beacon is placed in $B(0.600,\ 0.400)$ now. It is made one iteration. The real robot path is compared with the estimated path in Figure 51.

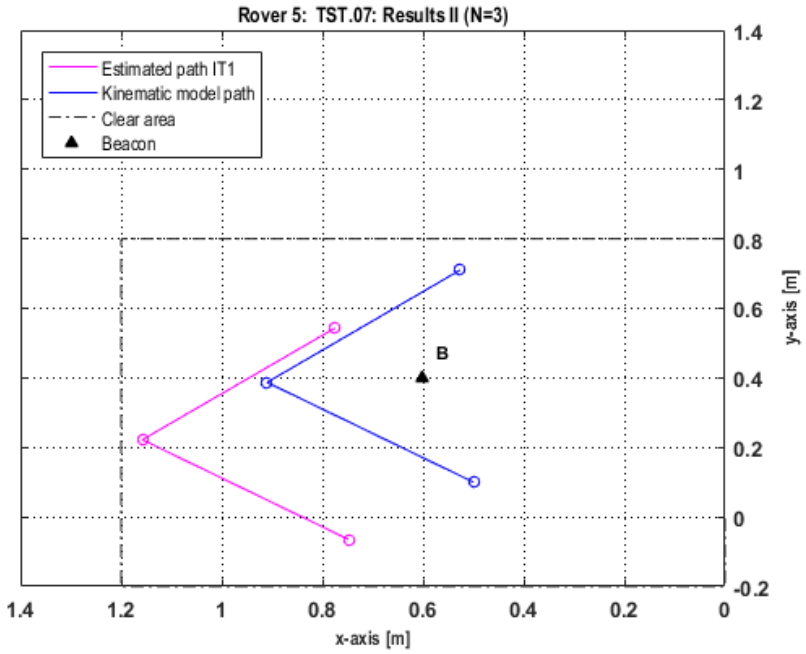

Figure 51. TST.07- Estimated path vs. Real path

From Figure 51 can be deduced that the absolute error is 0.248 meters on x-axis and 0.165 meters on y-axis. These errors are a bit higher compared with the ones obtained after solving same case by trilateration technique, which were 0.184 meters on x-axis and 0.141 meters on y-axis (see section 7.2.1). Furthermore, the error on x-axis is higher than the one on y-axis again, which again is probably due to the arbitrary points selected.

Lastly, it is measured the reliability of the estimated position computing the inverse of $|D|$ parameter (see section 5.2.2.1), after testing different arbitrary scanning points configurations.

Figure 52 shows the three configurations tested, hereinafter, *Case 1*, *Case 2* and *Case 3*. In all cases, the robot starts in $(0.600,0.200)$ and performs same angular motions, $\Delta\theta = 30°$, between scanning points. Consequently, the main difference between these three cases is the robot linear motion (see Table 19).

	Case 1		Case 2		Case 3	
	Δx [m]	Δy [m]	Δx [m]	Δy [m]	Δx [m]	Δy [m]
1→ 2	0.086	0.050	0.433	0.250	0.866	0.500
2→ 3	0.050	0.086	0.250	0.433	0.500	0.860

Table 19. TST.07: Rover 5 linear motion for Case 1, Case 2 & Case 3

Additionally, Figure 52 shows the trajectory followed by the robot on each case.

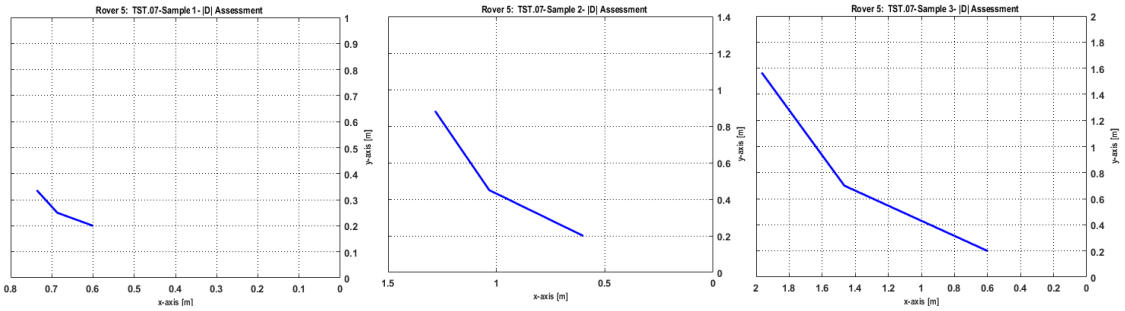

Figure 52. TST.07- |D| Evaluation cases (from Left-Right: Case 1, Case 2 & Case 3)

In Figure 53, it is possible to compare |D| parameter and its inverse, per each case represented in Figure 52.

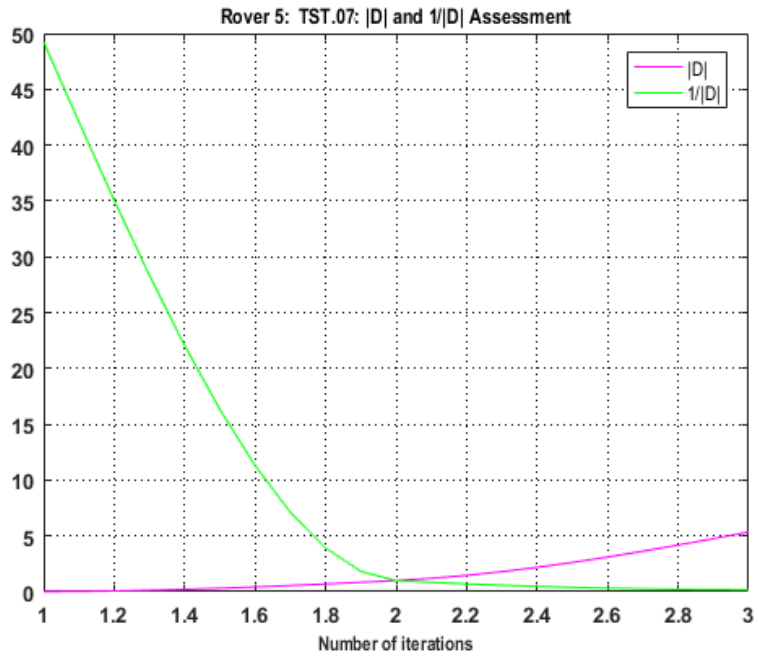

Figure 53. TST.07- Analysis of the inverse of |D| parameter

7.3.1.4. TST.07 Conclusions

It is concluded that if the scanning points are close to each other, the estimated robot position contains large errors, as with the trilateration technique.

From the second test (see Figure 51) where the robot performs larger angular and linear displacements, it is deduced similar position estimation error in both axis than the one obtained with trilateration technique. Furthermore, the absolute error on y-axis is lower than on x-axis, which is probably due to the scanning points regarding the reference system established.

Finally, the inverse of $|D|$ parameter decreases to 0 when the robot approaches to the critic centre, i.e., parallel power lines, which is consistent with the theoretical findings. According to Figure 53, the estimated position error, $1/|D|$, increases when the scanning points are close to each other or the power lines trends to be parallel, which can be verified with the test results obtained previously and the theoretical outcomes.

7.3.2. Test 08 (TST.08): Bearing Measurements at Optimal Positions

7.3.2.1. TST.08 Description

The TST.08 implementation pursues two basic goals: (1) to assess the robot estimation position using triangulation technique with angle measurements performed in optimal points, and (2) to analyse how the results vary depending on where the beacon is located with regard to the optimal scanning points.

Again, the Rover 5 requires same sensor configuration that in TST.07.

The scenario dimensions are limited to $1.200\,x\,1.000$ m^2, in order to avoid false-positive detected by the ultrasonic sensors. The origin of the inertial frame is located at $(0,0)$, which corresponds to the lower-right corner of the scenario previously defined. A stationary beacon is placed arbitrarily in this area. For simplicity, the y-axis is reversed due to the servomotor operating mode.

When the test starts, the robot is placed somewhere within the area defined. Then, it moves through pre-defined optimal points, and it takes angles measurements. The robot keeps stationary while it performs a scan of its environment. In this case, it is considered three scans ($N = 3$) performed in three different optimal points.

7.3.2.2. TST.08 Algorithm

The algorithm applied in TST.08 is the same as the one used in TST.07 (see section 7.3.1.2).

7.3.2.3. TST.08 Results

On the one hand, the Rover 5 makes angle measurements from three optimal positions. To that end, the robots performs following displacements ($\Delta x_2 = -0.250$, $\Delta y_2 = 0.433$, $\Delta\theta_2 = 120°$) and ($\Delta x_3 = -0.250$, $\Delta y_3 = -0.433$, $\Delta\theta_3 = 120°$). Furthermore, the stationary beacon is placed in $B(0.550, 0.344)$, so there is same distance between the beacon and all of the optimal points where the robot takes the angle measurements ($d_{r1} = d_{r2} = d_{r3} = 0.289$ meters). The robot estimated path is combined with the robot real path in Figure 54.

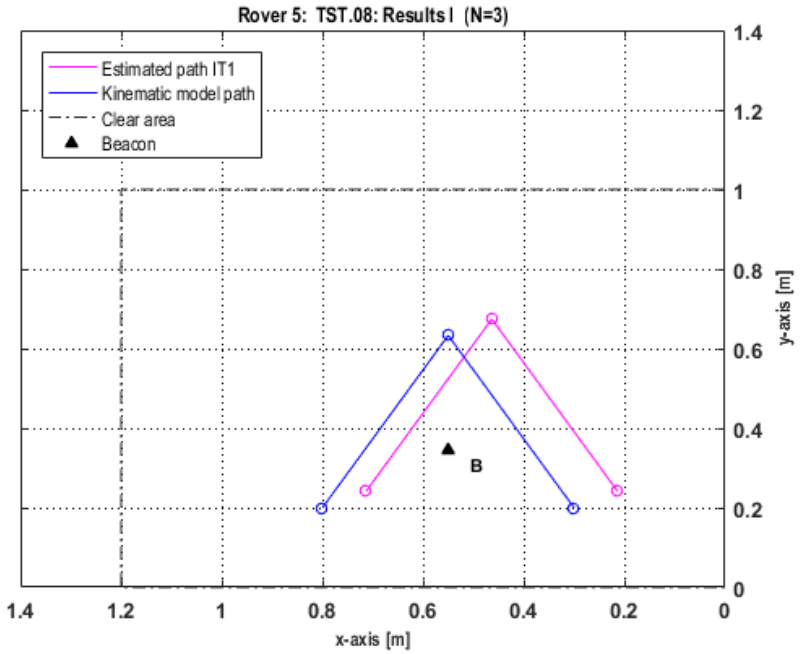

Figure 54. TST.08- Estimated path vs. Real path

After assessing Figure 54, it is concluded that the absolute error on both axes has decreased because of the scanning points are performed in optimal points. In this case, the absolute error on x-axis is 0.086 m, whereas it is 0.041 on y-axis.

On the other hand, it is assessed how the distance between beacon and the optimal scanning points can affect the results. Starting from previous robot and beacon configuration –assumed ideal case-, the beacon is moved along the y-axis and x-axis, so $d_{r1} \neq d_{r2} \neq d_{r3}$. The position of the beacon used on each test, as well as the distance to each optimal points are collected in Table 20.

	Beacon coordinates		*Distances to scanning points*		
	x *[m]*	y *[m]*	d_{r1} *[m]*	d_{r2} *[m]*	d_{r3} *[m]*
(Ideal case)	0.550	0.344	0.289	0.289	0.289
$IT1$	0.550	0.244	0.254	0.389	0.254
$IT2$	0.550	0.078	0.278	0.555	0.278
$IT3$	0.550	0.008	0.315	0.625	0.315
$IT4$	0.550	0.000	0.320	0.633	0.320
$IT5$	0.550	0.600	0.472	0.033	0.472
$IT6$	0.300	0.300	0.300	0.510	0.100

Table 20. TST.08- Beacon positions and distance to optimal scanning points

According to Table 20, it is computed the absolute error on each case taking into account the different beacon positions (see Figure 55).

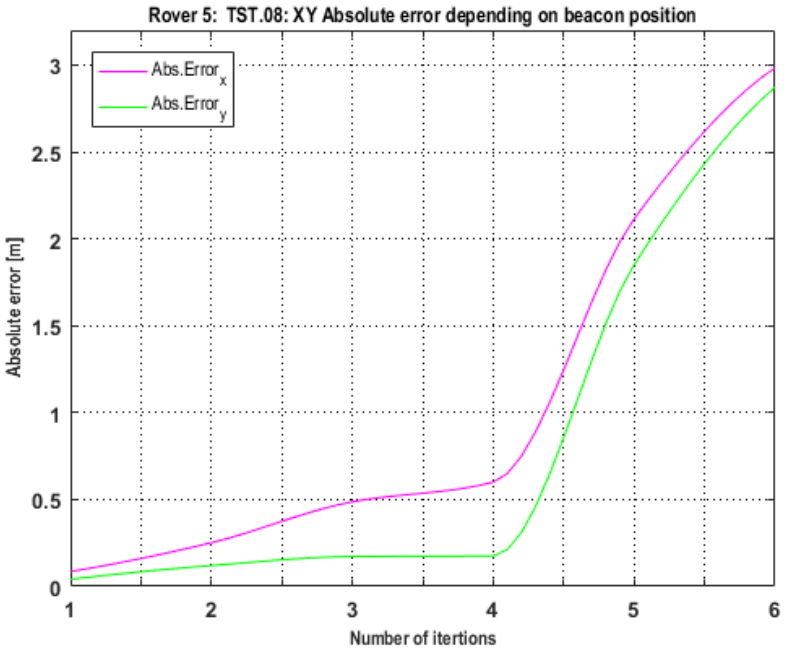

Figure 55. TST.08- XY Absolute Error

As can be seen in Figure 55, the beacon position affects the estimated position absolute error even when the robot scans in optimal points. From IT1 to IT4 the absolute error keeps below 0.5 m. This is possibly due to the beacon position varies only on y-axis

and there is no a large difference with the ideal case. However, it is deduced that the error increases exponentially when the beacon position is away of the ideal case, see for example IT5 and IT6. Additionally, it is noted that the absolute error is always higher on x-axis, independently of the beacon position.

Finally, it is also seen that the distance at which measurements are performed affects the accuracy of the results; therefore, the optimal points in triangulation are dependent of the angle of measurement but also the distance in this case.

7.3.2.4. TST.08 Conclusions

Comparing the estimated position absolute error obtained by triangulation (0.086 m on x-axis and 0.041 m on y-axis) with the one got by trilateration technique for the same number of scans (0.003 m on x-axis and 0.001 m on y-axis), it can be noted that the estimation error is higher with the triangulation algorithm than with the trilateration algorithm. The reason can be the sensors and the algorithm properties.

Finally, it is also concluded that even when the scanning points are performed in optimal points, the estimated position absolute error on both axes can vary depending on the distance between the beacon and these optimal points. The ideal case is when all distances are equal. Thus, the optimal points in triangulation depend on angle and distance, which does not happen when trilateration technique is applied.

THIS PAGE INTENTIONALLY LEFT BLANK

8. Robot Development Problems

In spite of the suitable results obtained with Arduino technologies, many different weaknesses have been detected.

On the one hand, the ultrasonic sensors modules provide easily false-positive lectures due to the sensors wide operating range. Consequently, the indoor scenario has to be well limited, but nonetheless the sensors lectures need to be filtered most of the time to mitigate future positioning errors. Furthermore, these sensors require specific beacons orientation, i.e. the beacon has to be perpendicular to the plane of propagation of the ultrasonic waves. The ultrasonic sensors also require some beacon properties, for instance, the bigger it is, the stronger the reflected signal will be. Consequently, the accuracy of the results is also affected by the beacon characteristics.

In addition, the ultrasonic sensor modules are powered through Arduino UNO board; this sometimes causes false sensor lectures due to power fluctuations, especially when the main board is supplied with any external source such as batteries.

On the other hand, Arduino has some limitations regarding the amount of data shown at the serial monitor. It is verified that the buffer can be overflown, especially in these cases where the sampling time is high. As a result, Arduino UNO board stops working. It is easily solved cleaning the buffer regularly and reducing the sampling time.

Finally, the batteries wearing affect sensor measurements and the accuracy of the mobile robot motion.

THIS PAGE INTENTIONALLY LEFT BLANK

9. CONCLUSIONS

9.1. General Conclusions

This project aims to control and determine indoor mobile robot pose, position and orientation, using multiple technologies and different relative and absolute positioning methods. To this end, a mobile robot was mainly built using open source Arduino hardware, and its software was developed with Arduino Integrated Development Environment.

The robot empirical kinematic model was estimated by performing several experimental measurements with multiple operating voltages. By doing some real-world tests, it was verified the accuracy of the estimated model, concluding that the kinematic model outputs can be assumed as a real robot positioning reference.

Then, as initial positioning method, a relative positioning was assessed based on inertial navigation sensors, mainly low-cost gyroscopes and accelerometers sensors compatible with Arduino hardware. These sensors provide a self- contained navigation without any external reference, just being required processing sensors data with known initial position, velocity and course. The results showed large position errors which even grew over time. So, it was concluded that this type of position estimation becomes inadequate for long-term applications and useful enough for short-time positioning.

At the core of this project, absolute positioning based on active or passive beacons were studied. A couple of mathematical techniques were implemented in several experimental tests. First positioning technique was based on range measurements, which is also known as trilateration. The mobile robot position was estimated considering the distance between different beacons and itself. In this case, the ranging technique was based on TOA. Second positioning technique implemented was based on triangulation, which consists of finding robot position using angle measurements made from a reference point. The ranging technique used by triangulation was AOA.

9.1.1. 2D Single Target Positioning Using Trilateration

Multiple tests were carried out to estimate the vehicle position using range measurements with respect to a known beacon. These tests differed between them in the number of sensors available and the quantity of beacons used to determine the robot position.

It has been observed that the same accurate results can be achieved even when the number of sensors and beacons are reduced. Moreover, it has been demonstrated that if range measurements are taken at theoretical optimal points, the results are extremely close to the maximum achievable, according to the theoretical references obtained from FIM or Cramer-Rao Lower Bound.

Finally, it was shown that there is no need to increase the range measurements from a certain number of measurements due to the results accuracy is not going to be improved. It was also proved that when it comes to real systems, the sensor lectures are not the same on each iteration, which differs from theoretical results.

9.1.2. 2D Single Target Positioning Using Triangulation

From the triangulation tests, it was shown that if the scanning points are arbitrary, the error in the position estimation is similar to the one obtained with trilateration technique. However, if the bearing measurements are performed in optimal points, the position error is higher than the one obtained by trilateration.

Moreover, it was also demonstrated that even when the scanning points are performed in optimal points, the estimated position absolute error in both axes can vary depending on the distance between the beacon and these optimal points, being ideal case when all distances between the scanning positions and the beacon are equal. This problem does not exist when it is applied trilateration technique.

Then, it has been demonstrated how both techniques, trilateration and triangulation, can be effectively implemented to localize accurately the mobile robot using low cost systems for measuring ranges and angles, respectively. Moreover, it has been shown that if the measurements are taken in the theoretical optimal points (in the sense of Fisher information), the accuracy obtained for the estimated position is very close to the theoretical maximum given by the Cramer-Rao Bound.

9.2. Future work

As future work, one important improvement that could significantly enhance the project value is the use of the Extended Kalman Filter to fuse trilateration and triangulation mobile robot position estimations. This improvement could make possible to obtain high accurate geo-positioning, even though low cost technologies and simple algorithms are used.

Likewise, the use of more sophisticated measurement sensors such RF transmitters/ receivers could mitigate some of the limitations found with the ultrasonic sensors.

So far, the experimental data is only shown on Arduino IDE through serial communication. One interesting improvement could be to save the data into a microSD card or sent it by WiFi to other Arduino board.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] J.Borenstein, H.R. Everett, L. Feng. *Where am I? Sensors and methods for mobile robot positioning*. University of Michigan, 2nd Edition. Michigan, United States. April 1996

[2] J. Borenstein, H.R. Everett, L. Feng, D. Wehe. *Mobile Robot Positioning: Sensors and Techniques*. Journal of Robotic Systems, Vol.14, issue 4, pp 229-340. April 1997. ISSN: 1556-4967

[3] E. Olson. *A primer on odometry and Motor Control*. Massachusetts Institute of Technology. Massachusetts, United States. December 2004

[4] D. Titterton, J.L. Weston. *Strapdown Inertial Navigation Technology*. The Institution of Engineering and Technology (IET), 2nd Edition. March 2005. ISBN: 978-0863413582

[5] M.W. Spong, S. Hutchinson, M. Vidyasagar. *Robot Modeling and Control*. John Wiley & Sons, Inc., 1st Edition. December 2005. ISBN: 978-0471649908

[6] H. Fan. *Theory of errors and Least Square Adjustment*. Royal Institute of Technology (KTH), Stockholm, Sweden. August 2010

[7] M. Adalja Disha. *A comparative analysis on indoor positioning techniques and systems*. International Journal of Engineering Research and Applications (IJERA), Vol.3, Issue 2. March-April 2013. ISSN: 224-9622

[8] A. Noureldin, T. Karamat and J. B. Georgy. *Fundamentals of inertial navigation, satellite-based positioning and their integration*. Springer, 2013 Edition. ISBN: 978-3642604651

[9] Y. Liu, Z. Yang, X. Wang, L. Jian. *Location, localization and localizability*. Journal of Computer Science and Technology, Vol.25, issue 2, pp 274-297. March 2010

[10] A. Alcocer. *Positioning and navigation system for robotic underwater vehicles*. Technical University of Lisbon- Higher Technical Institute (IST). Lisbon, Portugal. November 2009

[11] G. Papadopoulos. *Underwater Vehicle Localization using range measurements*. Massachusetts Institute of Technology, Massachusetts, United States. September 2010

[12] H. Tan. *Informatics in Control, Automation and Robotics*. Springer, 2011 Edition. ISBN: 978-3642258985

[13] U. Genschel, W.Q. Meeker. *A comparison of Maximum Likelihood and Median Rank Regression for Weibull Estimation*. Iowa State University of Science and Technology, Iowa, United States. June 2010

[14] W. Gander, M.J. Gander, F. Kwok. *Scientific computing- An introduction using Maple and MATLAB*. Springer, 2014 Edition. ISBN: 978-3319043241

[15] Å. Björck. Least Squares Methods. Handbook of Numerical Analysis, Vol. 1, pp 465-652. 1990. ISBN: 978-0444703668

[16] Å. Björck. *Numerical Methods for Least Squares Problems*. SIAM, 1996. ISBN: 0-898713609

[17] D. Moreno, A. M. Pascoal, J. Aranda. *Optimal Sensor placement for Multiple Target Positioning with Range-Only Measurements in Two-Dimensional Scenarios.* Sensors- Open Access Journal, Vol.13, issue 8. August 2013. ISSN: 1424-8220

[18] D. Moreno. *Adaptive Sensor Networks for Mobile Target Localization and Tracking*. National Distance Education University (UNED), Madrid, Spain. June 2013.

[19] K. Derpanis. *Cramer-Raou Bound*. York University, Toronto, Canada. September 2006

[20] A.N. Bishop, B. Fidan, B. Anderson, P. Pathirana, K. Dogançay. *Optimality Analysis of Sensor-Target Geometries in Passive Localization: Part 2- Time of arrival Based Localization.* 2007, 3rd International Conference on Intelligent Sensors, Sensor Networks and Information.

[21] A.N. Bishop, B. Fidan, B. Anderson, K. Dohançay, P.N. Pathirana. *Optimality Analysis of Sensor-Target Geometries in Passive Localization: Part 1- Bearing- Only Localization*. 2007, 3rd International Conference on Intelligent Sensors, Sensor Networks and Information.

[22] C.J. Cohen, F.V. Koss. *A comprehensive study of three object triangulation.* University of Michigan, Michigan, United States. August 1998.

[23] V. Pierlot, M.V Droogenbroeck. *A new three object triangulation algorithm for mobile robot positioning*. IEEE Transactions on Robotics, Vol. 30, issue 3, pp 566-577. 2014. ISSN: 1552-3098

[24] C.D. Guillem, T.S Rappaport. *Infra-red location system for navigation of Autonomous vehicles.* Robotics and Automation, 1988. Proceedings. 1988 IEEE International Conference on Robotics and Automation, pp. 1236-1238. April 1988

[25] J. Font, J.A. Batlle. *Mobile robot localization. Revisiting the triangulation methods*. IFAC Proceedings Volumes. Vol. 39, Issue 15. pp 340-345. 2006. ISSN: 2405-8963

[26] E.Z. Casanova, S.D. Quijada, J.G. Garcia-Bermejo, J.P. Gonzalez. *A new beacon-based system for the localization of moving objects*. IEEE International Conference on Mechatronics ad Machine Vision in Practice, Chiang Mai, Thailand. January 2002

[27] T. Tsukiyama. *Mobile robot localization from landmark bearings*. XIX IMEKO World Congress Fundamental and Applied Metrology, Lisbon, Portugal. September 2009

[28] M. Ligas. *Simple solution to the Three Point Resection Problem*. Journal of Surveying Engineering, Vol. 139, issue 3, pp 120-125. August 2013. ISSN (online): 1943-5428

[29] M. Pedro, D. Moreno, N. Crasta, A.Pascoal. *Underwater Single-beacon localization: Optimal trajectory planning and minimum-energy estimation*. IFAC-PapersOnLine, Vol.48, issue 2, pp 155-160. 2015. ISSN: 2405-8963

[30] S. Adarsh, S. M. Kaleemuddin, D. Bose, K. I. Ramachandran. *Performance comparison of Infrared and Ultrasonic sensors for obstacles of different materials in vehicle/robot navigation applications.* IOP Conference Series: Materials Science and Engineering, Vol. 149, issue 1, 2016. ISSN: 1757-8981

[31] J. M. Hughes. *Arduino: A technical Reference: A handbook for Technicians, Engineers and Makers*. O'reilly, 1st Edition. June 2016. ISBN: 978-1491921760

[32] A. Howard, M.J Matarié, G.Sukhatme. *Localization for Mobile Robot Teams: A Maximum Likelihood Approach*. University of Southern California, Los Angeles, 2001

[34] C. D. Ghilani, P. R. Wolf. *Adjustment computations. Spatial Data Analysis.* John Wiley & Sons, Inc., 5th Edition. 2011. ISBN: 978-0470464915

[35] J. Font, J.A. Batlle. *Consistent triangulation for mobile robot localization using discontinuous angular measurements*. Robotics and Autonomous Systems, vol. 57, issue 9, pp. 931–942. September 2009. ISSN: 0921-8890

[36] R. Siegwart, I. R. Nourbakhsh. *Introduction to Autonomous Mobile Robots.* The MIT Press, 2004 Edition. ISBN: 978-0262195027

[37] M. Schwarz, T. Rodelhutskors, D. Droeschel, et al. *NimbRo Rescue: Solving Disaster-Response Tasks through Mobile Manipulation Robot Momaro*. Journal of Field Robotics, Vol. 34, issue 2, pp 400-425. March 2017. ISSN: 1556-4967