Politecnico Di Torino

Department of Control and Computer Engineering

MSc. Mechatronic Engineering

InSystems Automation GmbH

Berlin, Germany

Development of an algorithm to find loads on the floor relatively to the position of the robot

University Tutors:
Prof. Luigi Mazza
Prof. Alessandro Rizzo

Company Tutor:
Eng. Henry Stubert

Student:
Daniele Indelicato S232815

POLITECNICO DI TORINO

Abstract

Mechatronic Engineering

InSystems Automation GmbH

Master of Science

Development of an algorithm to find loads on the floor relatively to the position of the robot

by Daniele INDELICATO

The Autonomous Guided Vehicles are a product that saw huge increase in the demand in the last decade. The reason behind this is due to the fact that they provide optimized solution to speed up production and reduce the costs. AGVs may differs accordingly to the specific task to fulfill or the building characteristics.

InSystems Automation GmbH is a German company in Berlin that, among other fields, is investing considerable financial resources into research and development of innovative solutions in this field. These new research plans and solution often come from the customer desire or demand to acquire an automated vehicle that would be able to fulfill a certain task. The research conducted in this work and thesis aims to develop an algorithm for one of those customized AGV called *The Stacklifter*. The project started two years ago from the idea to have a customized bot that it is able not only to recognize standardized objects from the ground, but to lift and re-locate them within a facility.

The major difference with what is currently available is the development of a customized software that should be able to fulfill the task, performing the recognition in a 2D environment, trying to avoid as much computation as possible as the robot is driven by PLC. It is actually this point that makes a huge difference as the majority of detection algorithm run on PC and work in a 3D environment exploiting image recognition. However, compared to a PC, a PLC has a huge limit in term of computation, especially since this kind of system is designed to run 24/7.

Acknowledgements

I would not have been able to pave this path leading to my professional career and to my graduation if it were not for my family and friends who supported me from the beginning of this journey and for this reason I thank you.

I would like to thank the InSystem Automation family as it has been a major pleasure to work with them as well as a major opportunity to learn and personal growth. In particular, I would like to say mention Henry Stubert And Torsten Gast as they believed in me since day one, motivating me even at difficult times.

I would also like to thank my Professors Luigi Mazza and Alessandro Rizzo from Politecnico di Torino, for introducing me to InSystems and supporting me during my time here in Berlin.

Contents

List of Figures

List of Tables

List of Abbreviations

AGV	**A**utonomous **G**uided **V**ehicle
RF	**R**eference **F**rame
LR	**L**inear **R**egression
Matlab	**M**atrix **L**aboratory
Rot	**R**otation
transl	**T**ranslation

Chapter 1

Introduction

1.1 What are AGVs?

An AGV (Automated Guided Vehicle) is a portable robot which follows a specific pathway and it is mainly used to move a large quantity of material, pallet loads between shipping/receiving docks and storage racks and move workparts between machine tools and stations.

AVGs travel velocity is usually slower than the typical human walking speed and they have several safety features such as obstacle detection, emergency bumper, warning lights and warning sounds.In order to control AGVs industries have got locator panel, CRT color graphics display, and central logging and report.

AGVs can be applied to various kind of industries and often in customized design. Some example of industries where these are being utilized are: pharmaceutical, chemical,manufacturing, paper and print, hospital and warehouses.

1.2 Market and applications

AGVs play a huge role in terms of material handling optimization and are becoming more sophisticated over time as well as their capabilities to accomplish more complex tasks.

(A)

(B)

(C)

FIGURE 1.1: Examples of AGVs [5]

Their demand in the industrial market (i.e. for warehouse application) has been increasing in the last decade as the whole manufacturing and distribution processes needs to be automated. According to statistics the automotive industry has the highest share of the total AGV market (almost 24%) immediately followed by the manufacturing industry[10]. However, they also found their way through the healthcare industry especially in the drug manufacturers supply chain. Besides that, there is very strong demand in countries in APAC such as China or India.

The estimated market size is about 1.12 billion in USD and the forecasts show a positive trend for the next decade[**grandviewreaseach**].

The are many advantages of using AGVs for optimization and one of the major reason is that these kind of systems are independently operated and are usually guided along defined pathways in the facility floor. Although the investment is very high at the beginning, in the long term the maintenance costs have proved to be way lower than hiring long term employees. Not to

mention that they are also able to accomplish tasks for which human-driven vehicles are used.

The ability of an AGV to find and follow the correct path is usually strictly connected to some elements that must be present in the warehouse, such as pathways. For this reason InSystems Automation GmbH has decided to invest in innovative solution to drive AGVs autonomously and to make them more versatile in order to suit the customers needs.

1.3 Why choose AGVs?

AGVs have proven to be particularly effective, not only in terms of productivity but also reliable, safe and flexible, as these robots can be reprogrammed to change their path or their task, without needing new physical equipment to be installed.

In addition, AGVs can operate almost around the clock and their method of delivery is very predictable. These can also operate in conditions that are not suitable for humans and most of the time they have better accuracy than normal workers.

Despite the fact that a big initial investment is immediately required at the beginning, in the long term the overall expenditure would be actually smaller than paying full time employees for the same job. Also, once the control system is set up, it is possible to increase the number of AGVs and thus decreasing the initial investment and the demand grows.

1.4 Company description

The project took place by **InSystems Automation GmbH** in Berlin, from March 2018 to August 2018. InSystems Automation develops special machines for production, material flow and quality tests and it is specialized on the production of robot since 2012.

The company was founded in 1999 by Henry Stubert and Torsten Gast and since then the number of employees keeps growing (around 60 in 2018).

Among the different kind of activities carried out by the company there are robot for productions or AGV. These are specifically designed according to customer demand and they can deal with loads of different weight, usually from 30 to 1000kg. The transport robots are developed under the name proANT.

The company features a certified competence since 2006 and some of its partners are Siemens and Wago.

FIGURE 1.2: InSystems ProAnt AGV line.

1.5 Case of study

One of the most innovative solution that InSystems wants to provide to its customers is the Stacklifter. This is an autonomous robot specifically designed to find and pick up boxes from the floor with the least contact possible with the objects. This AGV has been thought to be used warehouse and its shape and features are strongly related to that purpose.

However the major difference between most AGVs currently available is that it is not using any pathways or magnetic tape to drive. For this reason, the robot features a frontal 2D laser scanner whose purpose is the identification of the object as well as the driving towards the targeted location, while avoiding obstacles.

Thereby the purpose of this project is to develop an algorithm that takes care of all the sub-tasks associated with this major one. In particular the algorithms needs to:

- drive to a specific goal, avoiding obstacles in the path.
- Identify boxes among other objects and/or people.
- Select the box in the specific goal location.
- Plan the trajectory to pick up the box.
- drive to the unloading station

More specifically, the company was interested in the procedure of box recognition and trajectory planning towards the object, which is known as fine positioning, as normal driving and obstacle avoidance have been already taken care of by another algorithm.

The novelty of the approach consists in an innovative solution that exploits data coming just from a 2D laser scanner, whereas usually such systems features either more sensor or 3D ones. However, there were many challenges that needed to be addressed, such as the fact that, since the boxes are standardized, they are represented in 2D with something similar to a rectangle: a geometry that is very common in a 2D environment.

Nevertheless, the main challenge was making sense of this set of points, called point cloud, which is provided by the laser sensor. The point cloud is basically, a description of the environment, but this kind of information alone is not sufficient to fully understand how the different points are related to each other as well as where a particular object is located.

Despite the fact that there already exist a lot of different ways to autonomously drive a robot, the company has decided to explore other possibilities. In particular, because such an algorithm would allow to have less restriction and at the same time more flexibility as it could be used for other purposes as well.

1.6 Thesis Structure

All the aforementioned problems and objectives will be discussed in the following chapters. In particular the thesis is split in six parts, each one addressing a specific topic.

FIGURE 1.3: example of an AGV picking up a box.

Chapter 2 is about the current status with a brief discussion on how usually this kind of problem is being dealt.

Chapter 3 is an introduction to the hardware, like the robot and the sensor, as well as the different softwares used to analize and program .

Chapter 4 discusses how it is possible to gove meaning to data coming from a 2D enviroment and whether it is possible to manipulate this data.

Chapter 5 explorers the different techniques commonly used to drive a robot as well as the solution that revealed to suit the AGV for this particular application.

Chapter 6 brings about all the differences and adaptations that come organically when trasionining from theory to practice.

Chapter 7 briefly summarizes and discusses the obtained result while opening the path for future improvements .

Chapter 2

State of the art

2.1 Identifying the problem

According to what has already been said, this case of study can be decomposed in two sub-problems:

1. Object recognition and detection
2. fine positioning

There are already a lot of methods available to control AGV and fulfill that task in the current literature. This Chapter will mainly explore the current status in term of research for what concerns mainly the detection part, but also the motion planning and fine positioning.

2.2 Navigation

AVGs may differs from one another according to the ability to move in throughout the space[4]. The ability to actually recognize its relative position as well as to identify the path to follow is the major concern of navigation. There are many option when it comes to decide how a robot can be driven autonomously and the most common ones are:

- **Wire.** A wire is used to transmit radio signal to the robot, which features a sensor, close to the ground, to detect its relative position with respect to the radio signal.
- **Guide tape.** Tapes can be magnetic or colored. In this case the robot features appropriate sensors to recognize the pathway. Although this technique may not be expensive, it is subjected to degradation (damages and dirt).
- **Laser.** The AGV features a laser transmitter and receiver which is able to triangulate its current position exploiting the reflection coming from reflective tapes mounted on the objects.
- **Gyroscope.** A computer system assigns tasks to the robot, which through the gyroscope checks whether is following the correct path.
- **Natural Features.** The robot exploits the data coming from several different sensor and through the Monte-Carlo/Markov localization techniques it is able to find dinamically its position and to calculate the shortest path for the assigned task.
- **Vision Guidance.** The AGV operates using cameras to record features on the route, building a 3D-map with 360° images.
- **Geoguidance.** This technique allows the robot to recognize features of the environment or of the warehouse.

2.2.1 Steering control

Mainly, there are three different type of steering control for this kind of robots.

The first one is the differential control, which is the most common among the others and the simplest[4]. It allows the the AGV to move both forwards and backwards and it does not requires additional steering motors or mechanism. Basically the translation and rotation movements are possible due to the fact the wheels can move with a different speed from one another.

The second one is the steered wheel control, which tries to imitate a car's steering ability. Unlike the previous type this control is suitable for every kind of application.

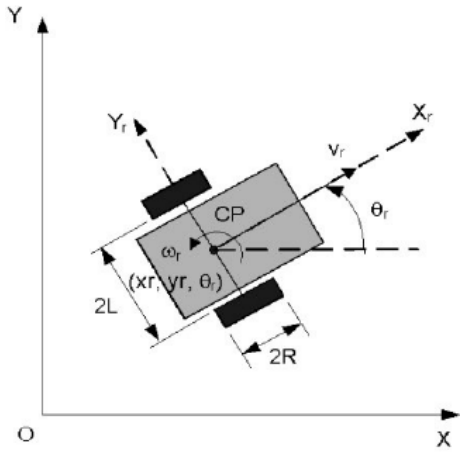


FIGURE 2.1: A differential driven robot [9]

The last type involves a combination of the previous ones and thus allowing the robot to rotate like a car and at the same to drive in differential mode along all directions.

2.2.2 Pathway

While moving AGVs should also take decision regarding the path to be followed. There are three main technologies that help the robot with such a task:

- Embedded guide wires. Wires on the floor that emit electromagnetic signal so that the vehicle follows them.
- Paint Strips. The vehicle features an optical sensor to track white strips on the floor.
- Magnetic Tape. This technique not only provides information about the patch but it can also regulate other parameters such as the speed.

2.2.3 Traffic

When a system features more than a single AVG, traffic control is necessary to avoid collision between robots. Even though, traffic control will not be treated here, it is worth mentioning three major techniques that are used to handle such a problem.

- **Zone control.** It is easiest to install and expand. Robots uses a wireless antenna to communicate each other and to know whether the path is clear.
- **Collision avoidance.** It exploits radar like sensor to avoid collision in the AVG area.
- **Combination control.** It is a combination of the aforementioned techniques. The redundancy of this approach allows to avoid collision in any case[4].

FIGURE 2.2: A forklift with safety laser sensor

2.3 Object recognition

Object recognition is actually a very general term as there are several different ways in which this can be performed, which strongly depend on the field of application. In fact, finding the shape of an object could be more interesting than its color, for example. Usually, it is possible to derive this data from sensors, in particular 3D sensors, mainly 3D LiDARs or 3D cameras, or a combinations of those sensors with other ones, like 2D LiDARs and so on. Combining the sensors is often useful, when it comes to compensate systematic errors coming from each sensor,as redundancy for safety reasons or even

to just be more accurate. The majority of these approaches is related to image processing and in particular in objects detection within an image. The image to be processed can often come from either a camera or a set of points coming from a sensor.

These sensors allows the extraction of set of points in XYZ coordinates which is fed as input to the detection algorithm and then afterwards the algorithm its self extracts all the information needed for the recognition process. However, this kind of algorithms are strongly based upon image processing and often require multiple stages and training, that allows the machine not only to detect the object and classify it, but also to keep track of the object when the this is moving, as well as storing in memory some new information acquired each cycle.

A research conducted in India [3] showed how it is possible to detect and distinguish between objects using a 3D camera. The camera could distinguish between different poses of the same objects. The camera provides the data as a point cloud and then the detection algorithm is able to perform the recognition after a training stage in which it acquires the necessary data. The object extraction is performed using Euclidean distance in the space.

Another reseacrh from 2005 [11] concerned the extraction of object in the terrain, using a 3D laser scanner. The data coming from the sensor is then, again processed, through image processing algorithm and requires a library that the algorithm needs for the classification phase, but ignoring all the data outside a preset region of interest.

Allan Zelener [12] also study the possibility to extend the ability to learn for machines in such a way that the overall knowledge and capability to recognize the different objects in 3D environment can be improved taking advantage of machine learning first.

2.3.1 A different purpose

Considering the status of the literature and the applications in many fields of detection algorithm, there are a few factors that are always common: the use of 3D sensor and image processing techniques. This is due to the fact that in most applications the algorithm used to process the data focus on multiple kind of object.

However, the purpose of this case of study can be seen on a different perspective as it is not necessary to classify every object that has been detected, but it is more important to recognize, detect and track the specific object to move around the facility. Besides, the company expressed the desire to have the detection algorithm running on the PLC, which does not allow the use of complicated image processing to extract data from the sensor. This is the major reason, aside from the costs, that motivated the company in researching something new and different for this kind of application.

2.3.2 triangle approach

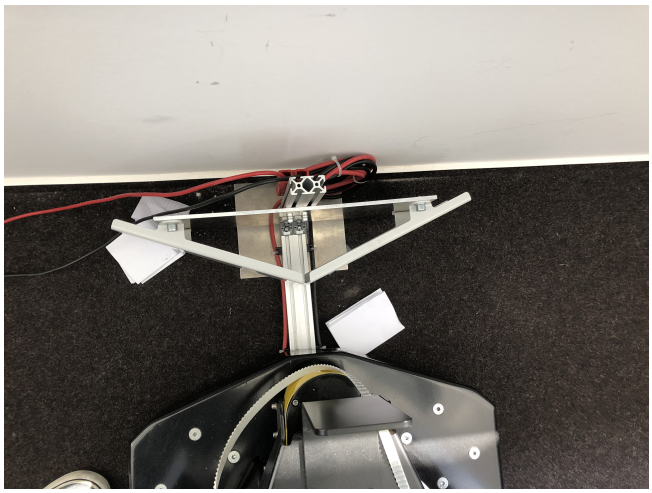


FIGURE 2.3: Triangle station.

The so-called "triangle approach" is the current way in which the company approaches to this problem. It exploits 2D or 3D navigation laser scanner or combination of them with cameras to detect a very specific geometry. In fact the triangle is used to let the AGV know where the targeted location is, avoiding to process each object present in the environment. The robot is specifically looking for the almost unique shape and geometry of the object. However the reason why the same algorithm cannot be used for the same purpose in this application is due the fact the a box features a 90° angle that especially in a 2D environment is a common geometry. Moreover, getting rid of the triangle station is actually one of the major goals for the company, as it allows to reduce the costs for the systems.

This is the starting point of the journey that has led to the development of the algorithm, where several possible paths have been evaluated and choices have been made in order not only to make it work, but to optimize it in

term of workload and efficiency. The next chapter explores the hardware and providing a better picture of the initial point and going even further into detail on the description of the problem.

Chapter 3

Hardware and Software

3.1 Overview

Before going into the details of what the algorithm is about and how the system works, it is important to understand the initial stage of the project. This chapter is an introduction to the hardware as well as the developing environment and the way these things can communicate with one another. Furthermore, it is provided an explanation of what the company is willing to achieve with this project and some of the reasons behind the choices that have been made.

3.2 The Stacklifter

The Stacklifter was born from the idea and need to build a special kind of AGV that is able to detect and lift boxes from the ground, with the least contact surface possible and the least space required to accomplish the task. It was thought to be used in warehouses or facilities and to be able to run continuously, where dimensions hugely affects the overall cost of the system as a whole. Indeed, this is the major reason behind the design choices.

The robot presents itself with a U-shape as well as two different sensors, one mainly used for navigation safety purposes and one, in the back inside the right arm, for the correct positioning and picking stage. It is manufactured to take boxes with a standardized length of 400 mm, whereas width

can vary between 200 and 600 mm and it can lift up to 100 kg. In addition, the arms are joint-less which means that these do not move even in the lifting process. Instead, it features the so called "L-profiles" which can be taken down or up allowing the lifting movements. These profiles are controlled by servo motor that work synchronously and these are placed in the arms. When they are taken down, they get completely out of the way whereas when they are taken up they push the box towards the center of gravity of robot, thus allowing a correction of the box position with respect to the robot.

However, for the robot features a frontal laser scanner used just for navigation purposes. The front scanner allows the robot to drive with the arms on the back and to detect any obstacle during the go-to-goal stage. The reason behind this choice is mainly safety, as driving in the opposite direction causes the arm to cover the line of sight of the sensor and therefore two laser scanner are needed. The second one located in the right arm is just used for fine positioning.

A fine positioning stage is performed every time the robot detects a box and tries to plan a motion in order to pick it up. Indeed, once the robot has reached the desired position it will turn in such a way that the arms are facing the detected box stacks to lift. The back laser scanner on the back is, then, activated allowing the robot to accomplish the task. Once the stack has been lifted the normal driving is restored, by disabling the scanner on the arm, allowing the robot to reach the drop-out position.

Similarly to the laser scanners, the are two different devices that are used to control the robot. Once is a computer, that controls the normal driving by means of the open source software ROS, and the other one is a PLC, which takes control of the robot in the fine positioning stage. The robot features also, a sensor that checks whether the loading procedure has been successful. The are also some LED button which are used to for the loading phase and normal turn on/off as well some emergencies buttons that are used to stop the robot whenever a fault occurs.

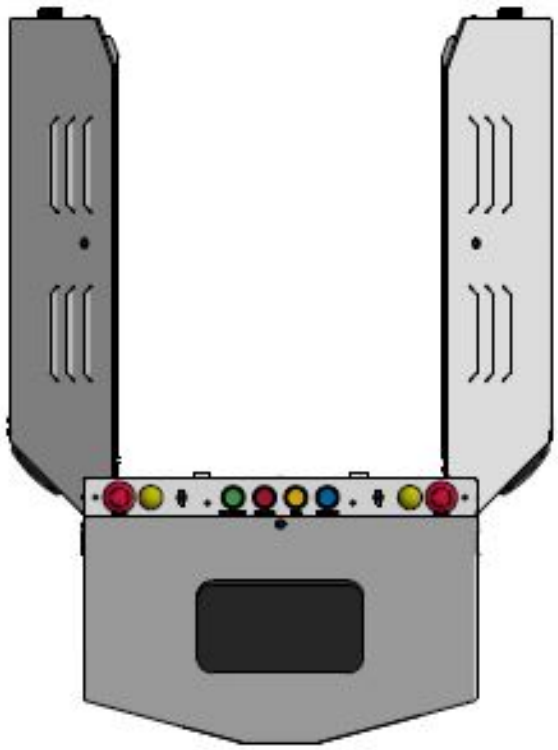


FIGURE 3.1: 2D Drawing of the Stacklifter

3.3 PLC

Programmable logic controllers or PLCs are computers specifically designed for managing and controlling industrial processes. PLC can in general execute a program and elaborate both digital and analog signal coming from sensors and actuators within the plant.

PLCs are modular objects and they are characterized by an extreme robustness. which is the main reason why those are usually used in noisy environments or exposed to extreme conditions. These can also run 24/7 on plants that can never stop working.

In terms of programming, most PLCs execute instruction from top to bottom and the language is subject to the IEC 61131 standard, which regulates the elementary data types ad well some syntax rules. This standards originated several different languages both graphical and textual which are:

- Ladder Diagram
- Function Block Diagram

FIGURE 3.2: A set of PLCs

- Structured text
- Sequential function chart

The first two languages are graphical whereas the remaining are textual.

The language used to carry this project on and test it is actually Structured text according to the standard, for which are available several libraries and options for code generation and testing.

3.3.1 Beckhoff CX9020

The PLC that was mounted in the robot and real brain of the operation is a Beckhoff CX9020. This device can be programmed through TwinCAT 2.11 using structured text and it features Ethernet connectivity, which is used to communicate with the sensor. As it is possible to see in fig.3.3 the device features an Ethernet socket as well a USB one, State LEDs, two micro SD slots and a PROFIBUS CANopen.

This PLC was mounted in the robot and it is responsible for controlling the motor and motion of robots. Generally speaking it can be used to receive data by an auxiliary machine in normal driving situation and then transmit the data to the motors.

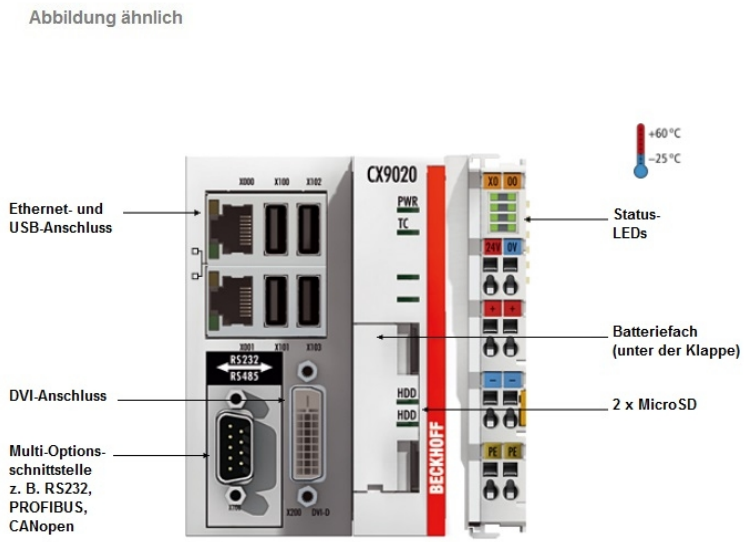


FIGURE 3.3: Beckhoff CX9020

FIGURE 3.4: CX9020 mounted in the Stacklifter

3.4 LiDAR sensor

"Lidar is a surveying method that measures distance to a target by illuminating the target with pulsed laser light and measuring the reflected pulses with a sensor. Differences in laser return times and wavelengths can then be used to make digital 3-D representations of the target. The name lidar is considered an acronym of light detection and ranging."[7]

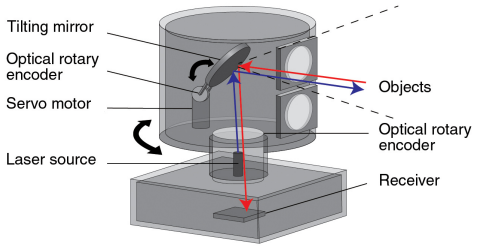


FIGURE 3.5: A typical Lidar Scheme [8]

Together with cameras, LiDARs are the most common sensor installed in robots to derive movement from local detection. High-end sensors can provide a dense point cloud over the field of view and, but those can be particularly sensitive to dust particle which may inhibit it. A Lidar is actually very similar to a radar, that uses radio waves instead of light.

A laser act as a source for a LiDAR system. It sends a beam of light with a fixed frequency, which allows the system to gather information about the environment according to how this light particles interact with the surroundings. The beam of light is coherent and particularly dense which makes it particularly suitable for detecting objects as it can detect materials that produce a weak radio echo.

3.4.1 Working principle

LiDARs and laser scanners in general are able to detect object through the Time of flight (TOF) principle. By this mean, the scanner measures the distances and the bacis shape of an object, exploting the reflected light from the surface. This kind of sensors usually feature a very accurate stop watch that allows to compute the time between when the beam of light is sent and it is reflected back.

The beam of light usually comes from a laser diod through pulses. Since the speed of light is constant and it is equal to c, the time of flight allows to compute the distance between the laser and reflecting surface as

$$d = \frac{tc}{2} \tag{3.1}$$

The accuracy of laser scanner is strongly dependent on how accurate the time can be measured as well. The sensor detect the distance from the first surface on which it shots the beam of light. It has to be clear that it explore its vision field one point at a time, changing then the direction of the bean of light. This is possible by rotating the laser diod or exploiting a complex system of mirrors. Tipically this last method is the most common as mirrors are lighter and can be rotated easier they are and more accurate.

The kind of data that usually is obtain as a consequence of that, is a set of points in the plane, in case of 2D scanners, which is commonly called Point Cloud. Due to some errors coming from the sensor of to the reflectivity of a certain surface the point cloud is affected by noise. However, the point cloud

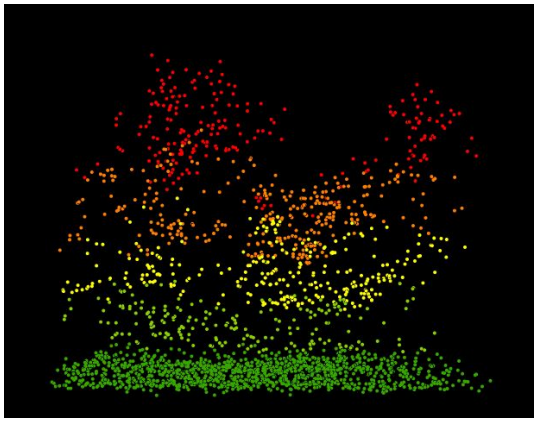


FIGURE 3.6: Lidar Point Cloud

alone is not sufficient to get all the needed pieces of information and it also depends on the kind of application these are used for. Making sense of the point cloud, then, is very important not only to to derive the pose of a object, which is the major goal for this application, but also to recognize objects and shapes.

In this particular application not only the LiDAR is used to estimate the pose of a particular box and therefore the detection phase, but also to dynamically adjust the robot trajectory during the picking-up phase, to help the robot select the closest box and to distinguish between boxes. In order to do that, the system needs an algorithm that is able to process the raw data coming from the sensor and give those a meaning is a such a way that there AGV is given all the necessary information not only to drive safely, but to accomplish the task with a very low rate of failure. The working principle of this algorithm as well as the way the scanner is used to drive the robot in the environment will be discussed in the next chapters.

3.4.2 SICK TIM561 2D laser scanner

FIGURE 3.7: SICK TIM561 2D laser scanner

The sensor used for the fine positioning is the SICK TIM561 2D laser scanner. This scanner can be used to export data using standard protocol and thereby communicating with the PLC. The scanner is positioning on the back in the left arm and the reason why this particular kind of scanner was selected is due mainly for its angular resolution, as the points in the point cloud would be particularly dense in proximity on an object, which would allow to have more accurate computations. Although Appendix A contains the device full data sheet, it possible to summarize the most important characteristics in the following table.

Aperture angle	270°
Scanning frequency	15Hz
Angular resolution	0.33°
Operating rage	0.05÷10m
Integrated interfaces	TCP/IP

TABLE 3.1: Scanner Specifications

FIGURE 3.8: The scanner mounted into the robot

In order to communicate the data via TCP/IP protocol, the sensor use the telegram format which may differ according to the chosen communication language (CoLa). In particular this type of sensor uses two languages which are:

1. **CoLaA**: characterized by variable length ASCII strings
2. **CoLaB**: which is based fixed length Binary telegrams

CoLaB language is the most suitable one when it comes to PLC application due to the fixed length of binary data and this is the reason why it has been selected.

For each scan the sensor is able to provide:

- array containing the distances
- start angle

CoLa A	ASCII	<STX>sRN{SPC}LMDscandata<ETX>
	Hex	02 73 52 4E 20 4C 4D 44 73 63 61 6E 64 61 74 61 03
CoLa B	Binary	02 02 02 02 00 00 00 0F 73 52 4E 20 4C 4D 44 73 63 61 6E 64 61 74 61 05

TABLE 3.2: Telegram example

- number of points
- accuracy
- RSSI1 (optional)

The distance comes in the form of radial distances which means that those must be associated with an angle to make sense. The sensor does not provide such data but it possible to create that array by exploiting the other pieces of information. Considering, also that the scanner is upside-down, the array vector can be calculated as it follows:

```
for n=1 to Npoints
        angle(n) = (180-((((n-1)*accuracy)+Start_angle)));
end_for
```

All the points are then translated into Cartesian coordinates as it is more intuitive working with those. Furthermore, it could happen that during the scanning phase, some of the points may be dazzled or invalid. Thanks to RSSI1, which is an additional array of optional values available directly from the scanner,which take into accoint the energy levels of the signal received, it possible to know when some points do not provide useful information or any information at all.

3.5 The developing environment

As it has been said already,the PLC uses a specific programming language called IEC 61131 Structured language. However, to ensure a certain flexibility to the algorithm as well as the possibility to implement the same idea in different devices, exploiting code generation, the Matlab/Simulink platform has been used.

Simulink is mostly suitable for real-time applications and features its own unique block diagram programming language,where blocks are executed from left to right. Among the many features offered by this platform, code generation in on of them. Code generation allows to deploy a Simulink project in different kind of device, translating the code in other languages. In particular the most important available ones are:

- C/C++ code
- IEC 61131-3 Structured Text for PLC

3.5.1 setting up the sensor

In order to fully understand which data the sensor was transmitting and how the telegram can be sent correctly, all the packets between the sensor and the PC have been analyzed through an open source software called WireShark. It can basically sniff packets exchanged using different protocols and transform them in a readable form for humans, helping to understand if something is missing, or if there is any sort of error in the communication. Using the

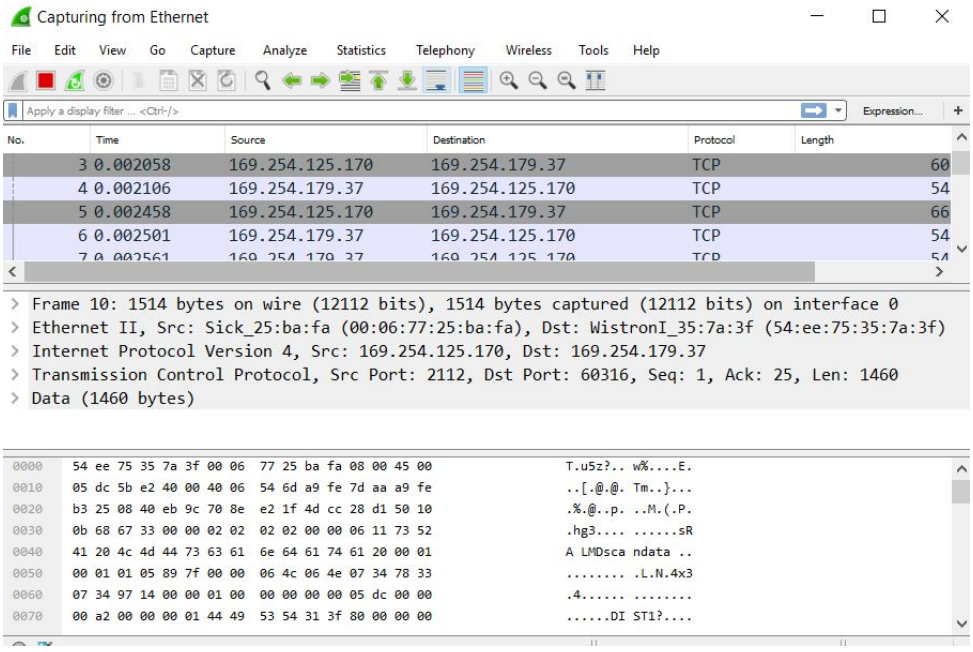


FIGURE 3.9: WireShark interface

results, provided by the packets analysis, it is possible to build a Simulink communication model, to extract and elaborate the data from the sensor. An

example of the initial stage communication model is provided by the following picture. This is the initial configuration setup, which allows the extraction

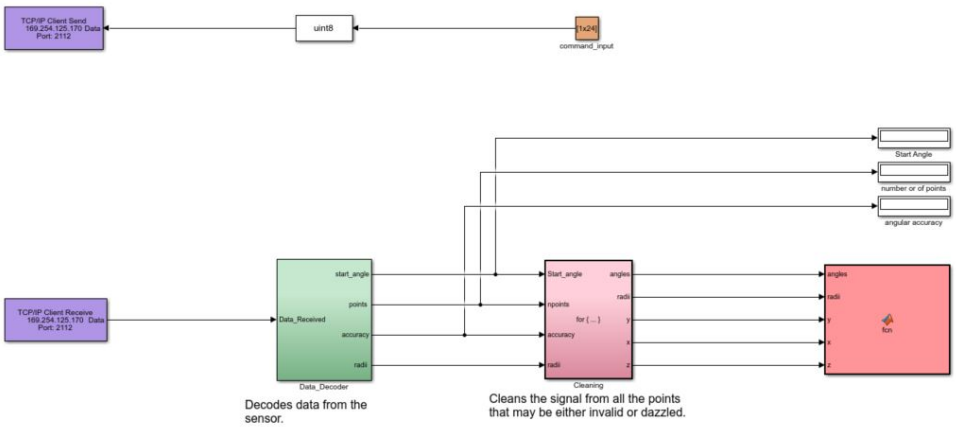


FIGURE 3.10: Simulink Initial Setup

of all the relevant data coming from the sensor and to build up a map of all the points in the sensor mobile reference frame.

The working principle is pretty easy to understand. First of all a command is sent to sensor in order to receive the data. As soon as the sensor replies the telegram is received through the purple block in the form of uint8 data type and then the green block decodes the received signal, giving as an output:

- the total number of points
- the start angle
- the radial distances
- the angular resolution

All of this data is then fed as input to the next block which basically derives the array containing the angles which are later used to derive the Cartesian coordinates, meanwhile removing all the invalid or dazzled points. After that all the data has been processed it is possible to build the point cloud. The total number of points is used to give the correct dimension to the array containing the radial distances from the center of the scanner Reference Frame. However, for the angle , the scanner does not provide the whole array, but just the start angle and the angular resolution, which can be used to derive the desired array.

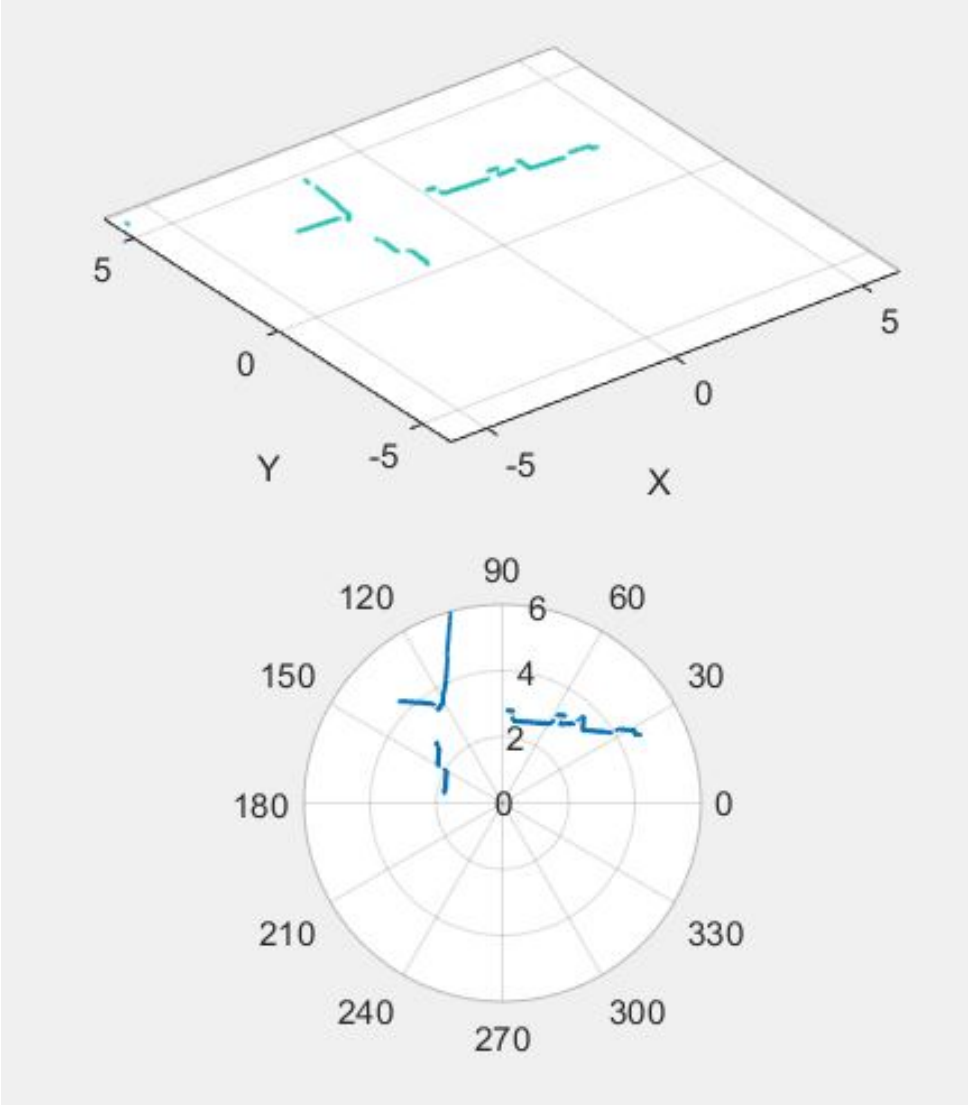


FIGURE 3.11: 2D Point Cloud

In Fig.3.11 it is possible to see the 2D point cloud, coming from the raw data extracted from the sensor. However, aside from being a se of points, this data does not contain any other information. Thereby, it is imperative to give them a meaning extracting all the useful information that would allow the robot to sense the environment,or at the least the most relevant information. The next Chapter discuss how this is possible and what kind of assumption can be mind to actually simply the problem.

Chapter 4

Making sense of the point cloud

4.1 The Goal

Once that communication has been established , allowing the extraction of the data gathered by the sensor, a point cloud is obtained. The point cloud is, basically a set of points in the $x - y$ plane. Each point in the point cloud belongs to a real object which causes the light to be reflected back to the sensor.

However, the point cloud alone does not provide any additional information,aside from coordinates. This means that it is necessary to investigate the nature and the relationships between points,as well the different possible shapes and the accuracy of the data. These problems need to be taken care of by a suitable algorithm, which is subject, however, to some limitations. The PLC, indeed, is not capable to execute complicated 3D image processing algorithms. These requires an high computational power and considering that the PLC must run continuously, stressing the CPU for long periods of time may not be the best solution.

This Chapter is about the idea behind that algorithm and how to take advantaged of the provided set of points as well as justifying the choices that have been made to achieve the algorithm. As mentioned before, the major goal is , of course, to detect the box. The starting point of this analysys begins from the known data, which are:

- a set of points in Cartesian coordinates

- the width of the box
- the shape of the box

Understanding which data is possible to obtain from these two initial pieces of information is actually the first step. Even though this known data alone is not sufficient for the resolution of the problem, it has to be said, that it is possible to extract additional information that not only is useful to distinguish between different objects, thereby identifying the box, but also to estimate other designing parameters, such as how big the area containing the box should be, how big the minimum distance between two consecutive boxes should be and the minimum distance that allows the robot to fully recognize the box and plan the trajectory.

On one hand, since all of the parameters related to the required space are subjected to the designing process, it may be desirable to reduce or keep them to the minimum possible, especially considering that space, at the least in terms of cost, is the major factor. Saving space would allow to keep more piles of boxes in the same facility, which is good for both the production and costs reduction.

On the other hand, however, tolerances and the necessary space for the robot movements must be taken into account. Which,in other words, means that the warehouse should be structured in such a way that the AGV can function at the best if its capabilities.

Finding a middle way between these two points is not always as easy as it seems, as it is basically reducing warehouse costs versus decreasing the computational effort and efficiency, which may, however, increase the overall costs.

Furthermore there are other pieces of information that can be indirectly derived from the initial given data. Indeed, it is important to keep in mind that the robot is not looking for the closest box. In fact Its moving algorithm provides some stop locations within the map, called goals. When the robot stops into a certain goal position it tries to identify a box in a specific area and , if no box is located, it just to moves to next goal position.

Since there a lot of things that need to be considered in the data processing phase, it was decided to split the problem into a subset of smaller problems,

addressing first the ones that do not depends on other sub-problems. This subset is made up by the following smaller tasks:

- Grouping all the points that belong to the same objects (Cluster Analysis)
- Recognizing the box in the set of objects
- Finding the fitting side

The next section of this chapter will specifically address the first point, illustrating what kind of techniques have been chosen or rejected, leading with a final and optimized solution, specifically tailored for this type of application.

4.2 Cluster analysis

In statistics clustering or cluster analysis is a set of techniques used to analyze data, whose purpose it the selection and the grouping of homogeneous elements, called clusters[6]. Clustering algorithms are generally used to serve this purpose. Although this kind of analysis is usually very popular in the statistics field, it has proven to be very effective for this kind of application. Grouping points that belong to the same object is indeed the same task for which these algorithms are used in the first place.

There are many kinds of clustering algorithms and each one of those is based on measurements related to how similar the elements are. There are various parameters that come into play when it comes to define how similar two points are, however, it very common that this measure is often expressed in term of distance.

Clustering algorithms usually differs according to definition of distance they are based upon and the way grouping is performed. It possible two classify clustering algorithms in two major categories:

- bottom-up. This means that at the beginning each element is seen a cluster. The algorithm keeps collecting elements into cluster until a prefixed amount of cluster is reached or the distance is above a certain threshold.

- top-down. At the beginning all the elements are considered as being part of the same cluster and then the algorithm proceeds to split the major cluster into smaller homogeneous clusters. The algorithm usually stops when a certain stopping condition is satisfied.

Clustering algorithm can also be exclusive, if it is not possible to assign one element to multiple subsets, or, non-exclusive if it is.

Another important factor is the way the general space is divided into subsets and in particular it is possible to define a **partitioning clustering**, with a prefixed amount of partitions or **hierarchical clustering**, where a hierarchy is built, characterized by a number of clusters, represented by a tree structure.

For this specific application, however, it is desirable to use an exclusive and partitioning algorithm. For the sake of this case of study, two different algorithms have been analyzed but only the one who revealed to be as a easy as efficient was ultimately implemented.

4.2.1 First attempt: K-means

The K-means algorithm is a partitioning algorithm which allows so separate a set of objects in K groups based on their attributes. The major goal of the algorithm is to minimize the total variance between clusters and each og these can be identified through an average point called "centroid"[1].

The iterative algorithm creates at first K partitions of the initial space, assigning points randomly or using heuristics methods. After that, it computes the centroid of each cluster, originating a new partition where each point is associated to the cluster with the closest centroid. Every time the clusters change, the centroids are computed again until the algorithm converges.

Let $X = X_1, \ldots, X_N$ be a set of N elements and $P = P_1, \ldots, P_K$ a set of partition characterized by the following properties:

- $\bigcup_{i=1}^{K} P_i = X$: the total set of partitions must contain all the initial objects
- $P_i \cap P_j = \oslash, i \neq j$: exclusion property

[1] 2.

- $\oslash \subset P_i \subset X$: no partition can be empty

Obviously $1 < K < N$ as it would not make sense to look for either one cluster or having a number of cluster equal to the number of objects.

Each partition is represented by a matrix $U \in \mathbb{N}^{K \times N}$, where the generic element $u_{ij} = \{0, 1\}$ means that the object j belongs to the cluster i. Let $C = C_1, \ldots, C_K$ be the set of K centroid, the objective function can be defined as:

$$V(U, C) = \sum_{i=1}^{k} \sum_{X_j \in P_i} ||X_j - C_i||^2$$

and then

1. Generate randomly U_v e C_v
2. Compute U_n that minimizes $V(U, C_v)$
3. compute C_n that minimizes $V(U_v, C)$
4. If it converges $\Rightarrow$ stop, else $U_v = U_n$, $C_v = C_n$ and go to step 2

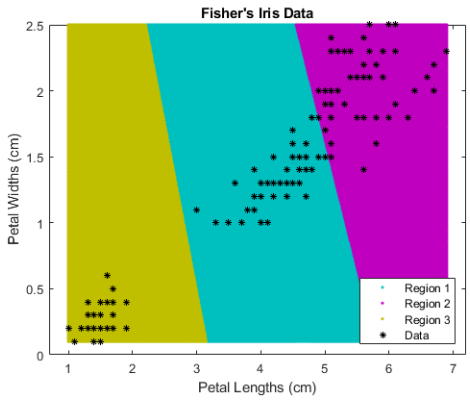


FIGURE 4.1: Example of K-means

The reason why this algorithm was selected at the beginning is because it is a very notorious one and can converge pretty quickly. However, as D.Arthur and S.Vassilvitskii have proved the algorithm could take a super-polinomial time of $2^{O(n)}$ to converge for a certain subsets of elements in the plan and although it is relatively fast there are several reason why it was not suitable for this application:

- It does not make sense for this kind of application to define at the beginning the number of partitions

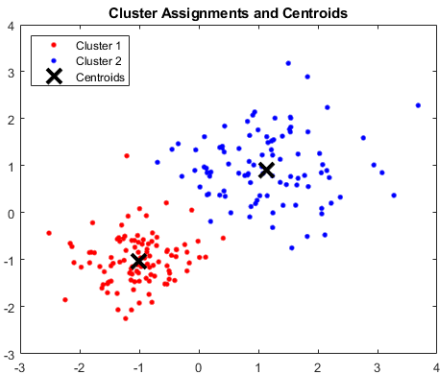


FIGURE 4.2: Object separation

- Most of the computations are useless, as it is not important to be that accurate
- It is not important to have all the clusters, but just separate the points in order to be processed
- It works better with spherical clusters
- Does not work with code generation for structured text

Although this algorithm may have proven to be very effective in statistics, it was not worth it to implement it, simply because the points are less aleatory and therefore, points belonging to the same object must be close to one another, in the Euclidean sense.

4.2.2 Second attempt: exploiting the known data

For the issues discussed before, although the K-means is effective in separating points in the plane, it is not suitable for this type of application. However, it possible to find another way, which is more computational efficient, accurate enough and that can overcome the limitations of the code generator.The main idea, then, was to keep it simple and it was possible to do that by making several assumptions in the attempt to simplify the problem.

First and foremost, the LiDAR sensor has an angular resolution of 0.33°. This is really important because it means that when an object is being detected , then the point cloud must be very dense on the portion of the plane where it is located.

Secondly, the vector containing the radial distances is already ordered, in the sense that if $P_i(\rho_i, \vartheta_i$ in the ith point, then next one is $P_{i+1}(\rho_{i+1}, \vartheta_i + 0.33^\circ)$. Although this does seem to have relevance whatsoever, it can be exploited, in the sense that , if two consecutive points belong to the same object, their distance should be very small. For the same principle, if a sudden jump in te measuremnts is observed, this most likely means that, if do not take noise into account, these do not belong to the same object.

Therefore, instead of looking for an objective function to minimize, as for the K-means , it possible to scan the whole array of elements in order to find a sudden jump. Basically, selecting a threshold toestablish a sort of limit on how close to points can be in order to be classified as belonging to the same object will allow to create split the points into subsets.

To better understand this, a short description of the algorithm is provided:

1. The component of the array is the starting point
2. Until the Euclidean distance is below a certain threshold , continue to add point in a cluster
3. If a sudden jump occurs (i.e. the distance is above the threshold), analyze the cluster
4. If no box is found go to the next cluster
5. If there is no next cluster, go to step 1.

FIGURE 4.3: Example of how dense the point cloud is

On one hand, with these approach the clustering algorithm is strongly linked to the recognition one, since as soon as one cluster is found, it is then immediately passed to the next part of the algorithm.

On the other hand, the computation is much faster (at most $O(n)$)and the idea behind is simple. Of course, this may cause some issues in case of multiple boxes have to be detected. However, that is not the case since, the problem specification requires the robot to scan for just one box immediately in front of it.

4.3 Recognizing the box

After being able to separate the different objects on the plane, the next step is clearly to identify if there is a box among this set of elements. However, what makes a box different compared to a wall or any other possible object, especially in 2D enviroment?

To answer this question it was necessary to analyze how the point cloud would change according to the pose of the Box in the plane. In particular, it was noticed that the it was possible to detect at most two sides of the box at the same time and that, depending on the orientation, sometimes just one.

Furthermore each side, considering also the noise affecting the point cloud, is actually pretty close to a straight line, which is a useful information in terms of geometrical description of the problem.

There are many techniques used to make a signal smoother and reducing the noise affecting it. Most of them, however, are not particularly adapt to signals with discontinuities. In fact the point is affected by discontinuities, which are actually used to distinguish between objects.

The challenge, therefore, was about how to filter the signal, make it more stable in a such a way that an almost perfect segment can be obtained, without impacting on those discontinuities. Filter used for continuous signal are highly inefficient in this case because they may alter the entire point cloud.

However, by looking at one of the most popular and used techniques in statistics, it was possible again to find a solution to this issue. This technique is known as linear regression.

4.3.1 Linear Regression

Regression, in general, allows to find a relation between the measured variables, based on data samples, extracted by an hypothetical infinite population. In statistics regression analysis is also associated with the resolution of the linear model. This technique is extremely versatile and it finds application in chemistry, geology , physics, economics and engineering.

Formally, linear regression is a method used to make an estimation of the expected value for a dependent variable $\mathbf{Y}$, given the values of other independent variables $X_1, ..., X_k$: $\mathbb{E}[Y|X_1, ..., X_k]$. The way linear regression is

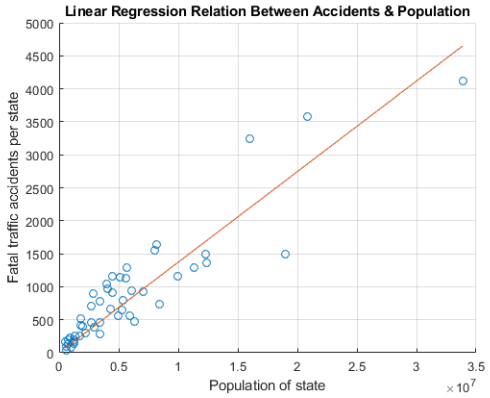


FIGURE 4.4: Example of linear regression

performed is strongly based on the model assumptions, which may influence the way one variable depends on the other. Clearly, in a 2D scatter, the box is well represented by one or two segments and these segments can be represented by a set of n points subjected to a certain law as $Y = f(X)$. Thereby, for this case of study it is safe to assume that there exists a relationship between the dependent variable and the independent one.

There are many different version of regression and each of those depends almost entirely on the kind of relation between the two or more variables. However, for this is specific case, since the entire box is described by segments, the relation between the two indipendent variable must be linear:

$$Y_i = mX_i + q \tag{4.1}$$

Where $i = 1...n$ and n is the number of points composing the segment.

It appear obvious that the desired relation between the two statistical variables is the equation of a straight line, which is linear. Thereby among of all the type of regression that can be performed, simple linear regression has been selected[**oss2014introduzione**].

4.3.2 Simple linear regression

Simple linear regression is based on a model where there is a single explanatory variable, called also regressor, which in this case would be X , while Y is considered as a variable dependent on X. Before going into details it is useful to review some statistical concepts, which are the expected value, the variance and the covariance[**oss2014introduzione**].

Let X be a random variable such that $X : \Omega \to \mathbb{R}$ and $\mathbb{P}$ its measurement of probability, the expected value of X, formally $\mathbb{E}(X)$ is defined as:

$$\mu = \mathbb{E}(X) = \int_{\Omega} X(\omega) d\mathbb{P}(\omega) \tag{4.2}$$

however in the descrete case, if p_i is the probability function, the expected value $\mathbb{E}(X)$ can be expressed as:

$$\mathbb{E}(X) = \sum_{i=1}^{\infty} x_i p_i \tag{4.3}$$

but since, in this particular case, $p_i = p_j = \frac{1}{N} \forall i, j$ and only a finite number of samples is available, it is possible to express the mean as:

$$\mathbb{E}(X) = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{4.4}$$

Another important statistical quantity is the variance which is defined as the expected value of the squared deviation from the mean:

$$\sigma^2(X) = \mathbb{E}[(X - \mu)^2] \tag{4.5}$$

which, by adding to this definition the same hypothesis that are valid for the mean value, can be expressed as:

$$\sigma^2(X) = \frac{1}{N-1} \sum_{i=1}^{N} [(x_i - \mu)^2] \tag{4.6}$$

Finally, another important parameter while dealing with more than a single statistical variable is the covariance, which is measurement that quantifies the variation of a variable with respect to the other one, and therefore their dependency from each other:

$$\sigma(X, Y) = \mathbb{E}\{[X - \mathbb{E}(X)][Y - \mathbb{E}(Y)]\} \tag{4.7}$$

which can be expressed as:

$$\sigma(X, Y) = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \mu_i)(y_i - \mu_i) = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y}) \tag{4.8}$$

Definition of the linear model

In order to perform simple linear regression it is necessary to find a linear model model for the problem. Considering all that has discussed until now, it should be clear that the linear model can defined as it follows:

$$Y_i = \beta_0 + \beta_1 X_i + u_i \tag{4.9}$$

Where:

- $i = 1, ..., n$
- Y_i is the dependent variable
- X_i is the independent variable
- $\beta_0 + \beta_1 \mathbf{X}$ is the regression function
- u_i is the statistical error

It is important that Y_i and X_i are measurements coming directly from the sensor and thereby can be treated as random variables. By expressing the linear model in this form is possible to study the relationship between the two variables.

However it is worth mentioning that this model is accurate while dealing with segments, which means that it is not suitable for other kind of object whose shape cannot be decomposed nor approximated into a subset of segments.

Formulation of the problem

Once the linear model has been defined, it is possible to proceed with the formulation for a two variable problem, which is:

$$y_i = a + bh(x_i) + \varepsilon_i \tag{4.10}$$

$h(x_i)$ is a generic function of x. In general it is assumed that $h(x) = x$ which leads to:

$$y_i = a + bx_i + \varepsilon_i \tag{4.11}$$

So the variable y can be extracted from an independent variable x and a random quantity ε_i.
There whole regression problem can, therefore, be reduced to the determination of a and b which define the relationship between y and x.
Some assumptions are necessary for the linear regression model:

- x is a deterministic variable
- $\mathbb{E}(\varepsilon_i) = 0$
- $\sigma^2(\varepsilon_i) = const\ \forall i$
- $\sigma(\varepsilon_i, \varepsilon_j) = 0\ \forall i \neq j$

Taking advantage of these assumptions it is possible to calculate the coefficients a and b according to the Ordinary Least Squares (OLS) method:

$$S = S(a,b) = \sum_{i=1}^{N} \varepsilon_i^2 = \sum_{i=1}^{N} (y_i - a - bx_i)^2 \tag{4.12}$$

Where N is the number of observations. It is possible to obtain the estimation of those values by solving:

$$\{a,b\} = arg\min_{a,b} S(a,b) \tag{4.13}$$

The solution to that equation can be derived from the partial derivatives of S with respect a and b when these are equal to zero:

$$\frac{\partial S}{\partial a} = -2\sum_{i=1}^{N} (y_i - a - bx_i) = 0 \tag{4.14}$$

$$\frac{\partial S}{\partial b} = -2\sum_{i=1}^{N} (y_i - a - bx_i)x_i = 0 \tag{4.15}$$

Then:

$$aN + b\sum_{i=1}^{N} x_i = \sum_{i=1}^{N} y_i \tag{4.16}$$

$$a\sum_{i=1}^{N} x_i + b\sum_{i=1}^{N} x_i^2 = \sum_{i=1}^{N} x_i y_i \tag{4.17}$$

Solving these equations it is possible to obtain:

$$b = \frac{N\sum_i x_i y_i - \sum_i x_i \sum_i y_i}{N\sum_i x_i^2 - (\sum_i x_i)^2} = \frac{S_{xy}}{S_{xx}} = \frac{\sigma(x,y)}{\sigma^2(x)} \tag{4.18}$$

$$a = \frac{\sum_i y_i \sum_i x_i - \sum_i x_i \sum_i x_i y_i}{N\sum_i x_i^2 - (\sum_i x_i)^2} = \bar{y} - b\bar{x} \tag{4.19}$$

Recalling that the variance is:

$$\sigma^2(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2 \tag{4.20}$$

and that the covariance is:

$$\sigma(x,y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y}) \tag{4.21}$$

where $\bar{x}$ and $\bar{y}$ are the observed mean values, it is possible then to derive:

$$\Rightarrow b = \frac{\sigma(x,y)}{\sigma^2(x)} \tag{4.22}$$

$$\Rightarrow a = \bar{y} - b\bar{x} \tag{4.23}$$

The OLS method, described until now, is basically a straight line that interpolates a scatter of points, minimizing the squared sum of the distances ε_i of the points composing the straight line. The reason why the squared quantities are considered is mainly to avoid compensation between positive and negative quantities during the calculations.

FIGURE 4.5: Linear regression: working principle

This may seem to have nothing to do with the major goal described at the beginning of this chapter. However simple regression not only does allow to denoise the point cloud, but also it stabilizes the signal, with enough accuracy. These are all qualities that are necessary for the estimation phase, as the overall uncertainty has been reduced.

There is one last piece that is missing to finish the puzzle. Nothing has been said about how it is possible to make the sensor know when an object can be represented as a set of segments and which are the conditions that legitimize this approach. Luckily the world of statistics offers enough tools and parameters to take care of those issues.

4.3.3 Pearson's Correlation coefficient

In statistics, there is a particular parameter that is used to estimate how strong the link between two variables can be. This is usually called Pearson's correlation coefficient or simply normalized covariance.

By definition, given two statistical variable X and Y is defined as ratio between the covariance and product of their standard deviation

$$\rho_{XY} = \frac{\sigma(X,Y)}{\sigma(X)\sigma(Y)} \tag{4.24}$$

As a main property:

$$-1 \leq \rho_{XY} \leq 1 \tag{4.25}$$

In particular the is some interesting information attached to $|\rho_{XY}|$. This parameter is basically able to tell whether there exist a correlation between the two variables involved.For this reason it s possible to make, at first glace, a classification, distinguishing between three major cases:

- $\rho_{XY} > 0 \Rightarrow X$ and Y are correlated positively
- $\rho_{XY} = 0 \Rightarrow X$ and Y are not correlated
- $\rho_{XY} < 0 \Rightarrow X$ and Y are correlated negatively

However, there is more. Nor only the normalized covariance is a mean to acknowledge the correlation, but it also quatifies the streght of it. In particular, the closer the value of $|\rho_{XY}|$ is to 1, the stronger the correlation is. Thereby, according to different values of the Pearson's coefficient, it can be established how strong the link between the two variables is:

- $0 < |\rho_{XY}| < 0.3 \Rightarrow$ the correlation is weak
- $0.3 < |\rho_{XY}| < 0.7 \Rightarrow$ the correlation is moderate

- $|\rho_{XY}| > 0.7 \Rightarrow$ the correlation is strong

Looking just at number does not make it clear why this is indeed the last piece to complete the puzzle. However, by representing the different degrees of correlation, one interesting more fact comes into play. In fact, when the correlation is strong, and in particular $|\rho_{XY}| = 1$, the link is of the linear kind.

Finally not only this parameter is able to tell if a correlation is present, but it is also able to check whether all the assumption, that have been made for simple linear regression, still suffice. Putting all together allows to indentify the box among that ocean of points.

4.3.4 Putting all together

Up to this point, with cluster analysys it is possible to dismember the point cloud into subsets, which represents real life objects on a 2D plane. Furthermore through statistics and simple linear regression it is possible to look for objects in the plane that can be represented by one to two segments.

Actually, the fact that the box can seen as two segments may generates problem as the whole shape of the object is not a segment and cannot be expressed by the linear law and model that has already been discussed.

However, experiments have proven that when two sides are being detected, the normalized covariance measures either a weak correlation or no correlation at all. Exploiting this information, the object can be split in two different subsets. If the two subsets show a strong correlation, then, most likely, the object is a box.

According to what has been already discussed, it is not so easy to grasp the link between the statistical property of the correlated variables and how the box could be detected. To better understand this, it is necessary to talk a little bit more about the geometry of the problem and how statistics come into play.

Except that, the shape of the box in a 2D environment in very common. This means that there could be potential object having a big $|\rho_{XY}|$, because their shape can also be represented by set segment. To be more accurate it is

necessary to add some more condition to be satisfied, in order to eliminate ambiguity.

This issue can be solved by considering other factors that are already known. In fact the shape of the box as well as the dimentions of the box are standardized.

Putting all together, it is possible to conclude that all of these factors matters and need to be part of the algorithm as prerequiste for the targeted set of points. In this way it is not only possible to reduce the number of false positives but also to create an algorithm that is efficent and that avoids to compute more data than necessary.

Chapter 5

Motion Planning

5.1 The Unicycle Model

Not only it is important to implement a robust and efficient detection algorithm, but also to plan the motion of the AGV. This is basically a control problem, which needs a model as a fist thing. Not to mention,that there are other kinds of issues that need to be taken care into consideration such as the ability to drive to a goal location avoiding collision as well as the fine positioning.In this chapter each problematic is discussed in detail as well as the provided solution to overcome them. Also, some additional key points are analyzed as it is necessary.

From a driving perspective, the Stacklifter is essentially a differential drive robot. This mean that it can be driven by acting on the wheels velocities. In particular, when the two wheel are rotating in the same direction and with the same rates,the robot perform a translation. If these are rotating in the opposite direction to one another, then a rotation is performed. Therefore, the lack of a steering wheel does not represent a problem for the motion, as playing with the wheel velocity allows the robot to curve and follow a certain part.

However, building a model under the assumption that it is sufficient to control the wheel velocities is not very intuitive. In fact, it is easier to reason in terms of rotations and translations and, thereby, of angular and linear velocities v and ω.

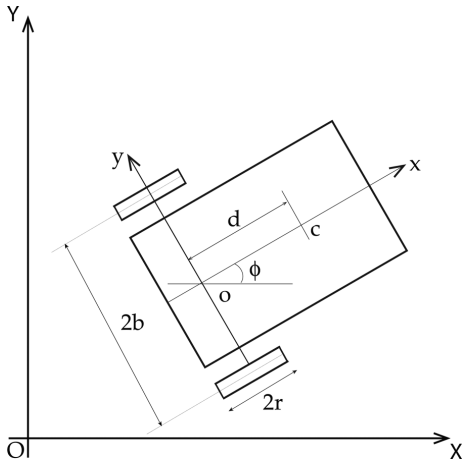


FIGURE 5.1: Differential drive robot model. [9]

Luckily it is possible to find a relation between these quantities and to generate a model that is controllable in terms of v and ω and this is the so called Unicycle model. This allows to get rid entirely of the differential driving problem and to treat the robot as if it has just wheel with steering ability. This model allows to simply the problem because in this way the only rele-

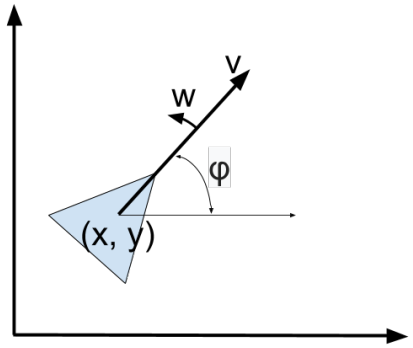


FIGURE 5.2: The Unicycle. [1]

vant information are the position of the robot in the space and the direction where it is driving. So the model needs some sort of relation that links the wheel speed velocities v_L and v_R to these quantities. In order to find that, it is possible to consider as a state the vector

$$[p] = \begin{bmatrix} x \\ y \\ \phi \end{bmatrix} \tag{5.1}$$

Where:

- x and y are the 2D coordinates of the center of gravity
- ϕ is the driving direction

Exploiting the kinematics laws, knowing the radius R and the wheel axis length L it is then possible to obtain the following:

$$\begin{cases} \dot{x} = \frac{R}{2}(v_R + v_L)cos\phi \\ \dot{y} = \frac{R}{2}(v_R + v_L)sin\phi \\ \dot{\phi} = \frac{R}{L}(v_R - v_L) \end{cases} \tag{5.2}$$

With the same consideration for the Unicycle it is possible to obtain the following system of equation.

$$\begin{cases} \dot{x} = vcos\phi \\ \dot{y} = vsin\phi \\ \dot{\phi} = \omega \end{cases} \tag{5.3}$$

These two systems of equations represent two different models. The next step, then, is to implement the unicycle model on the differential drive one as $\dot{y}$, $\dot{x}$ and $\dot{\phi}$ should be the same for both. This leads to:

$$\begin{cases} v = \dfrac{R}{2}(v_R + v_L) \Rightarrow \dfrac{2v}{R} = v_R + v_L \\ \\ \omega = \dfrac{R}{L}(v_R - v_L) \Rightarrow \dfrac{\omega L}{R} = v_R - v_L \end{cases} \tag{5.4}$$

by solving the equation it is possible to obtain:

$$\begin{cases} v_R = \dfrac{2v + \omega L}{2R} \\ \\ v_L = \dfrac{2v - \omega L}{2R} \end{cases} \tag{5.5}$$

Where v and ω are design parameters and R and L are known measured parameters of the robot. So, at this point, a clear relationship between the two model has been established and from now on, the model that will be considered is just the unicycle one.

Differential drive robots are the most common kind of robots and unicycles model them. For this reason there are many ways in which is possible to transform the dynamics of the robot is a such a way that it is even easier to actually track the robot itself. Up until now, the model is characterized by a non linear differential equation. However, while controlling the robot motion the focus, in this case, is not on the particular path that the robot is following, but more on the initial and final position. Indeed it is possible to consider another point on the robot that differs from the center of the unicycle considered so far. Assuming that the new point has a distance l from the previous point, it is possible to derive the following equation

$$\begin{cases} x' = x + lcos\phi \\ y' = y + lsin\phi \end{cases} \tag{5.6}$$

It is interesting to see the dynamic from this new point. Computing the derivatives of the previous set of equations, the following result is obtained:

$$\begin{cases} \dot{x'} = \dot{x} - l\dot{\phi}sin\phi \\ \dot{y'} = \dot{y} + l\dot{\phi}cos\phi \end{cases} \tag{5.7}$$

Considering that $\dot{x} = vcos\phi$ and that $\dot{\phi} = \omega$ it possible to rewrite the equation in this form:

$$\begin{cases} \dot{x'} = vcos\phi - l\omega sin\phi \\ \dot{y'} = vsin\phi + l\omega cos\phi \end{cases} \tag{5.8}$$

Assuming that it is possible to control directly this new point in such a wax that

$$\begin{bmatrix} \dot{x'} \\ \dot{y'} \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5.9}$$

it possible to solve the equations with respect v and ω

$$\begin{bmatrix} cos\phi & -sin\phi \\ sin\phi & cos\phi \end{bmatrix} \begin{bmatrix} v \\ l\omega \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5.10}$$

The first matrix however is indeed a rotation matrix, which rotates the vector of angle ϕ.

$$R(\phi) = \begin{bmatrix} cos\phi & -sin\phi \\ sin\phi & cos\phi \end{bmatrix} \tag{5.11}$$

So the system of equation can be rewritten as it follows:

$$R(\phi) \begin{bmatrix} 1 & 0 \\ 0 & l \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5.12}$$

In order to solve the equations with respect v and ω, it is necessary to invert the matrices. $R(\phi)$ does not generate any problem as rotation matrices as their inverse is always equal to the transposed matrix. Same goes for the next matrix as it is a diagonal matrix and the inverse is always equal to the inverse of the diagonal.

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{l} \end{bmatrix} R^T(\phi) \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5.13}$$

This is a way to actually make the point $P = (x', y')$ behave perfectly and by getting l small P is almost equal to the original point and it is possible to treat the system as

$$\dot{x} = u \tag{5.14}$$

However, this might be sufficient for a go-to-goal situation, whereas is this case also the orientation in the final position is very important in order pick these up. This is why although this simplified model is exploited for normal driving situations, it is preferred to use a regular tracker such as a PD for the fine positioning stage.

5.2 The regulator

The main qualities to be worried about while designing a controller are stability, tracking and parameters robustness. PIDs regulators (PI, PD etc) are controller implemented virtually almost everywhere. Each action of the PID as a specific mean but at the same time the combination of those may lead to achieve those the desired qualities mentioned at the beginning. Generally speaking a PID can be expressed in the Laplace domain as:

$$U(s) = K_P\left(1 + \frac{1}{T_I s} + T_D s\right)E(s) \tag{5.15}$$

By calling:

- $K_I = \frac{K_P}{T_I}$ as the integral constant
- $K_D = K_P T_D$ as the derivative constant

Tuning K_P,$K_I$andK_D is very important to the effectiveness of the controller. In particular each parameter as a specific function. The proportional action is a contributor to stability, in the sense it does not necessarily guarantee stability but it helps the system to become more stable as well as making it more responsive, although not very fast. The integral action, instead, is actually very useful for tracking and, generally speaking if the system is stable, this action alone suffice to track the system. Also the integral action is often used for disturbance rejection and it has a slow-rate responsiveness, due to the fact that it is necessary to accumulate the error over time before getting a response and this may cause oscillations. Finally, there is the last component which is the derivative action. This part of the PID is not responding to a certain value, but to changes in the value is usually characterized by a faster responsiveness. However this is usually very sensitive to noise which means that the derivative constant should not be too large.

By far PID are the most used low level controller and it is possible to find them in chemical processes , DC motors and so and so forth. However, this kind of controller does not guarantee stability and it is not a one size suits all. Usually, in complicated robotic systems the PID is not enough and

the controlling part is designed taking into account additional factors in the modeling process. Nevertheless, the PID is still a feedback controller and it has a remarkable ability to fight uncertainty in model parameters.

5.2.1 Descrete PID

Taking advantage of Euler's left rectangular integration and the backwards derivative it is possibe to express the input of the system as:

$$u_n = K_P \left[e_n + \frac{T}{T_I} \sum_{k=0}^{n} e_k + \frac{T_D}{T} (e_n - e_{n-1} \right] + M_R \tag{5.16}$$

where M_R is a forward compensation term. With trapezoidal integration instead it is possible to get:

$$u_n = K_P \left[e_n + \frac{T}{T_I} \sum_{k=0}^{n} \frac{e_k - e_{k-1}}{2} + \frac{T_D}{T} (e_n - e_{n-1}) \right] + M_R \tag{5.17}$$

However, the derivative term has to made in a different way, limiting its effect.

$$\frac{T_D s}{1 + \frac{T_D s}{N}} \tag{5.18}$$

Where N is usually between 3 and 10.

5.3 Odomoetry

The State of the Robot in the 2D plane is defined by its pose

$$[p] = \begin{bmatrix} x \\ y \\ \phi \end{bmatrix} \tag{5.19}$$

The pose represents the state of the robot at a given time and the way it is possible to obtain this information is called odometry.

Odometry exploits different kinds of sensors, whether these are external (i.e. measures something in the environment, like laser scanners, cameras, GPS etc.) or internal (like a compass, accelerometers and gyroscopes). Usually, the external sensors are not enough alone to grant accurate data on high level applications. This is why these are usually coupled with internal sensors, among which the most common ones are encoders. These devices generally tick count the revolution of the wheels and based on this number it is possible to know approximately where the robot is.

To explain how this is possible, it has to be assumed that a differential drive robot is moving at a constant velocity. Imagining the the path is a curve, the wheels would describe on the ground an arc, which is not going to be the same for both wheels. In fact the opposite wheel compared to the turning direction will mark a different arc compared to the other. These arcs allow to calculate the distance the center has turn, as it is the variable of interest. Calling S_R and S_L the distance each wheel has turn, the distance of the center of gravity can be calculated as:

$$S_C = \frac{S_L + S_R}{2} \tag{5.20}$$

S_C allows to find the pose of the robot:

$$\begin{cases} x' = x + S_C cos\phi \\ \\ y' = y + S_C sin\phi \\ \\ \phi' = \phi + \dfrac{S_R - S_L}{L} \end{cases} \tag{5.21}$$

Most wheel encoders give the total tick count since the beginning. For both of wheels it is possible to consider, then, the difference between the previous tick count and the actual one.

$$\Delta tick = tick_{old} - tick_{now} \tag{5.22}$$

Based on that it possible to compute the turning distance of the wheel as:

$$S = 2\pi R \frac{\Delta tick}{N} \tag{5.23}$$

Considering that N, represents the number of ticks per revolution, it is possible to map ticks onto distance of travel.

However, it is not possible to rely just exclusively upon encoders as this technique does not take into account the drifts. In fact a system like this can drift and especially in situations, like this case of study, it is important to be particularly accurate. For this reason, odometry is also combined with other external sensors, typically laser scanners. For example, in the stack lifter, there are actually two laser scanners, one on the front for normal driving and on the back for the fine positioning.

Chapter 6

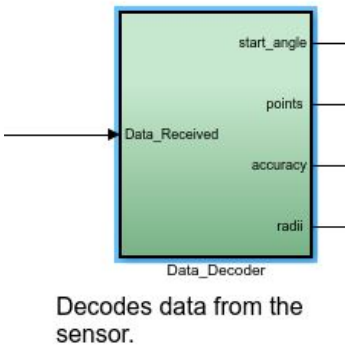
Modeling and testing

6.1 From theory to the designing process

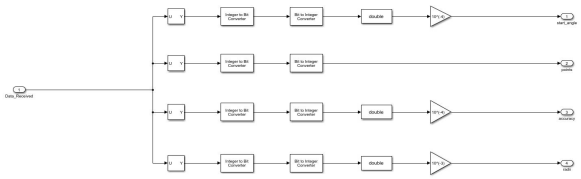
This chapter is about the whole modeling process and all the choices that have been made in order to make the algorithm a little bit more robust and stable. The first step is of course asking the sensor for data. The sensor itself can provide data automatically with 15 Hz frequency or manually sending the command each time. Since it is necessary to make some computation after acquiring multiple data from the scanner, it was decided to use the manual scan. Therefore, the software just asks for new data whenever it is ready to acquire new data and compute again.

6.2 Decoding and filtering

The received data from the sensor needs, of course, to be decoded. The sensor generates, upon request, a response telegram containing all the data, which are sent as bytes. Now the response telegram is made up of different parts, each one containing different information regarding the sensor parameters or flags and so on. Despite the fact that all the data comes in the form bytes, it needs to be decoded as each information uses different types(like uint8 or int32), thereby requiring different conversion methods to the decimal system.



(A) Data Decoder block



(B) Inside the Data Decoder Block

The received telegram consists of 2556 bytes from which it is possible to extract:

- the start angle
- the number of points
- the angular resolution
- the radial distance of each point

However, this data needs to be processed a little bit more so that it is possible to remove all the invalids point and generate the array containing the angle as well as the Cartesian coordinates. So the next block is meant exclusively to fulfill that task.

6.3 Box finder block

The box finder block is basically the core of the detection algorithm and, therefore, the most interesting part of the whole detection model.

There were several issues that needed to be addressed during the development phase. The most problematic factor was Simulink coder generator as

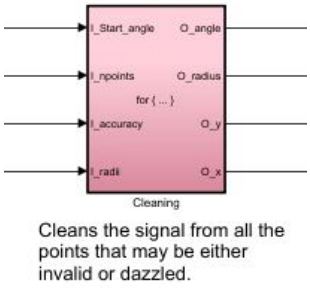


FIGURE 6.2: Cleaning block

it does not support variable size arrays and there also other kind of restrictions on the functions that can be converted to be used in the PLC.

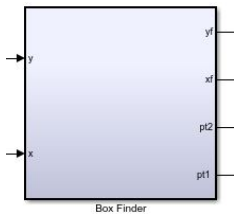


FIGURE 6.3: Box finder block

6.3.1 objects separation

As mentioned before, the technique used to distinguish between objects is solely based on the euclidean distance:

$$d_{n-1}(\underline{x},\underline{y}) = \sqrt{(x_{n-1}-x_n)^2-(y_{n-1}-y_n)^2} \quad \forall n \in \mathbb{N} : n = 1,\dots,N$$

where

- $\underline{x} = [x_1,\dots,x_n]$ is the array containing all the x coordinates
- $\underline{y} = [y_1,\dots,y_n]$ is the array containing all the y coordinates
- N is the total number of points
- $\underline{d} = [d_1,\dots,d_{n-1}]$ is the array containing the euclidean distances between two consecutive points

After that, it is necessary to set an appropriate threshold to justify whether the points belong to the same object. In order to do that several experiment

have been made and the optimal value was set 0.05 m. Considering that the angular resolution is 0.3333°, that value is justified by the high density of points when these are describing an object.

Basically, the algorithm extracts the object from the X and Y arrays. Each object is delimited by a starting and an ending point. As soon as an object is being detected, it is automatically processed in the attempt to find the box.

With respect to the K-means algorithm, in these way it is possible to reduce the overall number of computations necessary as the goal of the algorithm is to find just a single box and to stop when that happens. As a result, it is highly unnecessary to build an array for each object nor to set a predefined number of objects and so on, as these information are irrelevant.

When an object is located, the next step is to understand which kind of object the system is dealing with. The next section will describe the possible scenarios as well as the implementation of linear regression as a mean to identify the box.

6.3.2 Detection

The detection phase must take into account of all the possible situations that may occur, especially since the box is not necessarily positioned right in front of the sensor, but it could be slightly inclined on the plane.

In a 2D environment that implies that at most two sides of the box can be detected at once. Thereby among all of the scenarios there are two major cases:

- A horizontal line is detected which means that only one side is being detected
- L-shape made up by two different segments is detected, which means that two sides are being detected.

Horizontal line

The horizontal line is the easiest and most probable scenario and it is simply necessary to check whether there correlation between X and Y is strong

enough and then, to measure the length of the segment. If the length is acceptable, then it is assumed that the detected object is the box.

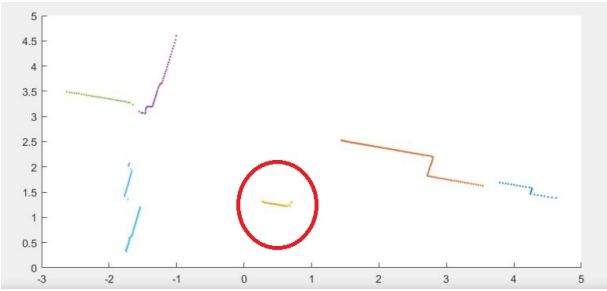


FIGURE 6.4: Example of a detected horizontal line

The reason why this case is the most probable one is due to the fact that the robots will always try to position itself along the normal through the middle points.

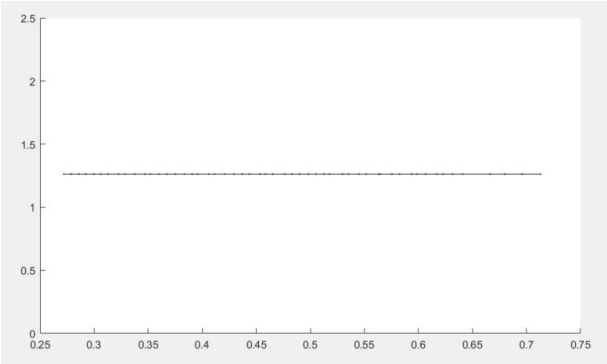


FIGURE 6.5: Example of simple linear regression

L-shape

Depending on the orientation of the box or from the position with respect to the sensor, it is possible that the laser detects two sides of the box. When that occurs, things get more complicated, because until now it has been said that the algorithm can recognize a straight line and not this particular shape.

It would be possible to adopt another type of linear regression in order to get a best approximation of the L-shape, but still it does not solve issue due to the fact that two sides are seen as a single object. Moreover approximating perfectly the curve is not relevant. For these reasons, among other technique, it was chosen to adapt this scenario, to the algorithm so that its simplicity could still be preserved.

Therefore, the L-shape scenario exploits the same principles already described but instead this time it is necessary to perform simple linear regression twice. The main idea consists in splitting the whole figure into two subsets of objects in order to treat each one of these as a segment and check whether the object could be an inclined box. However, in this case it is necessary that the Pearson's correlation coefficient is acceptable is both cases in order to proceed with further analysis. Once the box has been recognized, it also detect the side which side the robot should pick up the box from.

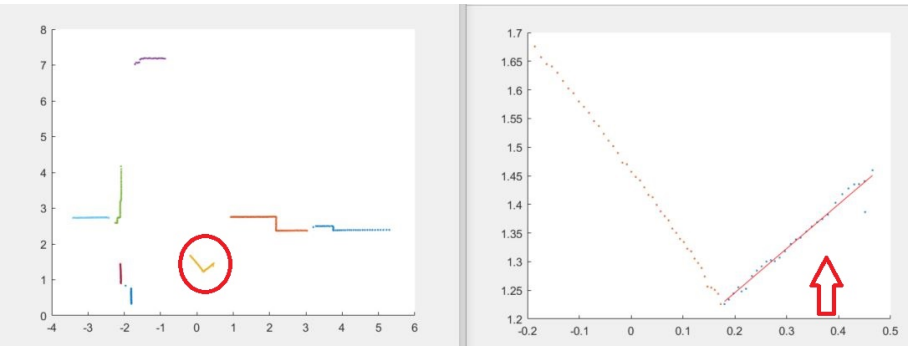


FIGURE 6.6: L-shaped box

The algorithm

The overall steps that the algorithm performs can be summarized in few bullets point that convey the general idea behind it:

1. From a starting point, create a subset (which is the object) of points and fill it up until there is a sudden jump in the distance between two consecutive points
2. Compute all the statistical values associated with the object
3. Evaluate ρ_{XY}. If the value is acceptable, measure the length of the segment and check it is the box, else go to the following step
4. Find the intersection between the two segments
5. Compute all the statistical values for both segments
6. Evaluate ρ^1_{XY} and ρ^2_{XY} for both segments. If both parameters are above the threshold,measure and check. Else, go to step 1.

6.3.3 Exceptions

For each of the possible scenarios listed in the previous section, there is a sub-case in which the statistical approach does not seem to be very effective in detecting the box. When one side of the box is parallel to the $y = 0$ line the correlation between Y and X is not strong as $\rho_{XY} \longrightarrow 0$.

This happens because in the mobile Reference Frame of the sensor the equation of the line would not appear in the canonic form:

$$y = mx + q$$

Where:

- m is the angular coefficient
- q is a translation constant

but instead in equation appears more in the form of

$$y = k \quad \forall x$$

where k represent a constant number. This mean that, in this reference frame, the Y is no longer a variable dependent from X , but instead it behaves like an independent variable as it is no longer a function of X.

This mainly occurs when a horizontal line is detected, but sometimes depending on the pose of the box, also for the L-shape curve. Theoretically speaking it would be possible to treat this case in a different manner, but the number of cases will increase. Plus it is not so easy setting the threshold to determine whether this is the case. For all of these reasons, it was decided to use rotation matrix to overcome this issue:

$$[R] = \begin{bmatrix} cos(\vartheta) & -sin(\vartheta) \\ sin(\vartheta) & cos(\vartheta) \end{bmatrix}$$

where $\vartheta = \pm 20^\circ$ depending on which kind of rotation is needed. In this way it is possible to maintain the number of cases to two, as once the rotation has been performed Y is again dependent on X.

6.3.4 Stability and robustness

the aforementioned algorithm is pretty effective in the determination of the box pose, however some extra steps were necessary to stabilize the overall system in order to ensure fewer parameters fluctuations as well as to reduce the number of unnecessary iterations and computations. Briefly, the several issues have been addressed:

- Removing unused or unnecessary points
- Noise affecting the point cloud
- Reducing the number of iterations
- Dealing with more than a single box

Removing unused or unnecessary points

To reduce the number of points and thereby of iterations that the algorithm has to make, it is important to keep in mind that:

- The AGV's attempts to detect the box, begin when it reaches a goal position, which is typically placed 1 meter away from the box
- It is not important to identify object, but just to find the box
- the sensor has 270° angle aperture and work between 0.05m and 10m
- the longest side of the box is at most 600mm

At first glance it is possible to restrict the number of points by excluding for example some angular section as it is not important to detect something in that regions. Same goes for the radial distance, which can be reduced from 0.05m to 3m. This adjustment itself reduce the overall workload by half, without having a negative impact in the whole system, which became more efficient and faster.

Noise affecting the point cloud

The way noise itself manifest within the point cloud is through point fluctuations and misalignment with respect the ideal straight line that the box should draw. Clearly, this implies that even the statistical quantities themselves are subjected to sudden changes for each cycle and thereby it is possible that the Pearson's correlation coefficient is not almost one, but smaller. Reducing the overall threshold for an admissible level of ρ_{XY} is of course a solution, but subjected to limits as decreasing this parameter too much may result in false positives. The picture above is a clear example of how the noise

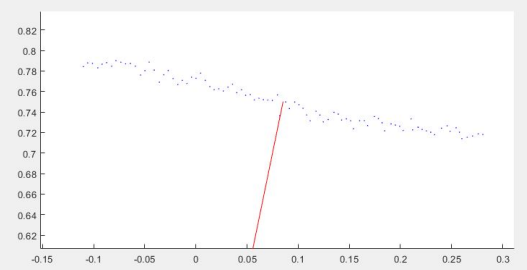
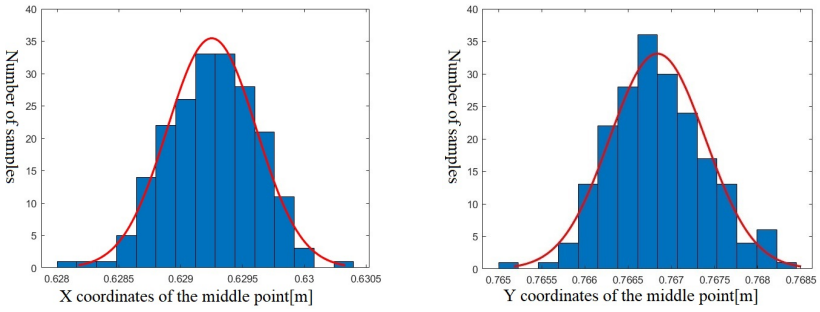


FIGURE 6.7: Noise affecting the point cloud

if affecting the point cloud and it is also example of how the width of the box is represented in the 2D scatter. The red line represent the perpendicular to the middle point of the side and it is used to positioning phase. Performing linear regression not only allows to perform better measurements during the detection phase but also, makes that perpendicular line more stable.



(A) (B)

FIGURE 6.8: x and y coordinates fluctuations of the average values

Another way in which is possible to see how much noise interferes with the algorithm is when two sides are being detected simultaneously. As a matter of fact, in the L-shape case it is possible to observe huge fluctuations in

the intersecting point, which may cause to misread the length of the segment or data misinterpreting.

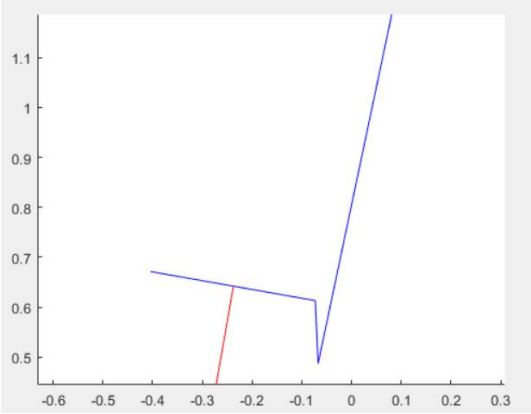


FIGURE 6.9: Example of perturbation

One way two make the algorithm a more stable is to compute the ideal intersecting point. Basically, the algorithm receives an object with a L-shape and it has to find the initial intersecting point between the two segments, which is the point with minimum coordinates. In this way however, this point is not stable due to noise/. Computing the ideal intersecting point, after performing linear regression, strongly reduces the fluctuations for this point, making the whole identification process more stable.

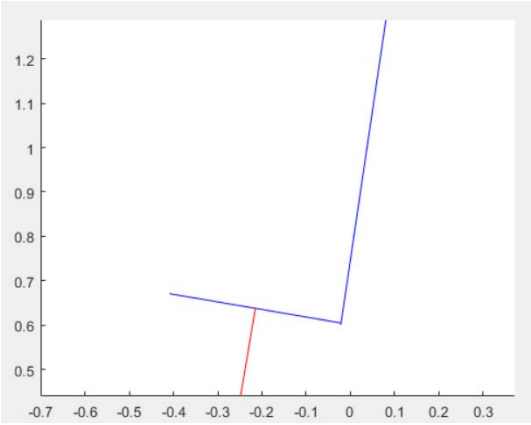


FIGURE 6.10: Stabilized intersection

Reducing the number of iterations

The number of iterations for each cycle can be reduced by avoiding to analyze object that obviously cannot be the box. It is in fact possible to consider the dimension of each cluster and to set both a lower and an upper bound. If

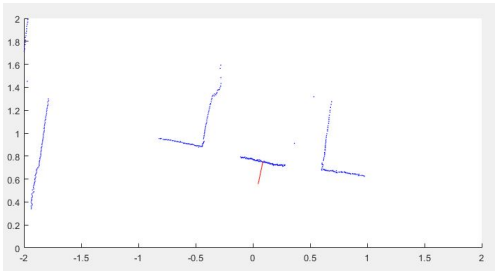
fact after several attempt the number or points needed to represent the box is between 50 ÷ 140 depending on many sides are being detected at once. In this way, for example, if a wall is detected it is most likely that its cluster dimension would be bigger than the set upper bound and the entire cluster is not be analyzed. Same goes for isolated points and small objects, with the lower bound.

Dealing with more than a single box

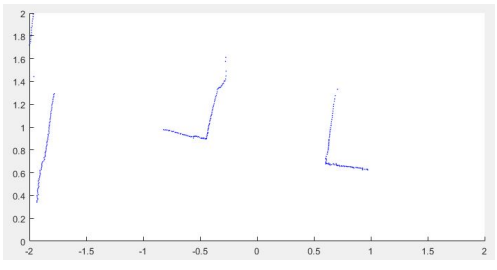
When dealing with multiple boxes the algorithm might be highly unstable, due to the fact that it is constantly switching between the detected boxes.

In addition, this represent a problem for the final application, as the only detecting region of interest in located in front of the robot. This means that if no box is detected, the robot should move to next goal position, instead of looking for a nearest box in an another location.

One way to solve this issue is to adopt further restriction in the region of interests. Doing this not only improves the overall performances as the data to be processed is being reduced, but also avoids to detect multiple boxes, and box outside the region of interest. As it is possible to see from the pictures



(A)



(B)

FIGURE 6.11: Multiple boxes environment

above, the algorithm only detect a box in a specified goal location. If no boxes are present in that location, no other box is being detected.

6.3.5 Warehouse specifications

Another aspect that is important to keep in mind is the applicability of the model in a facility. In other words, it means providing an answer to the following questions:

- How far can two consecutive boxes be from each other?
- How far can the goal location for the AGV be for the detection phase?
- How much space does the robot need to steer and finding the correct path?

To answer this question it is necessary to take into account the following factors:

- The robot needs to perform a rotation, before entering in the detection phase
- The robot arm is about 15 cm
- The robot needs to be able to drive to the box location

For all these factors at the beginning it was decided to be very conservative in terms of facility specifications, which is possible to find in the following figure. At the beginning it was thought to be very conservative with the specifications and to change them later, if possible, depending on test result and new techniques.

6.3.6 Tracking

The main goal of the algorithm is to detect a box in the ground. However, nothing has been said about cases where there is more than a single box lazing on the ground. This is a problem because the robot needs to prioritize one with respect to the others and it also needs to be able to stick with it during the motion.

To better understand the issue it is important to clarify that, due to the limitation imposed by the programming language and, the absence of tools

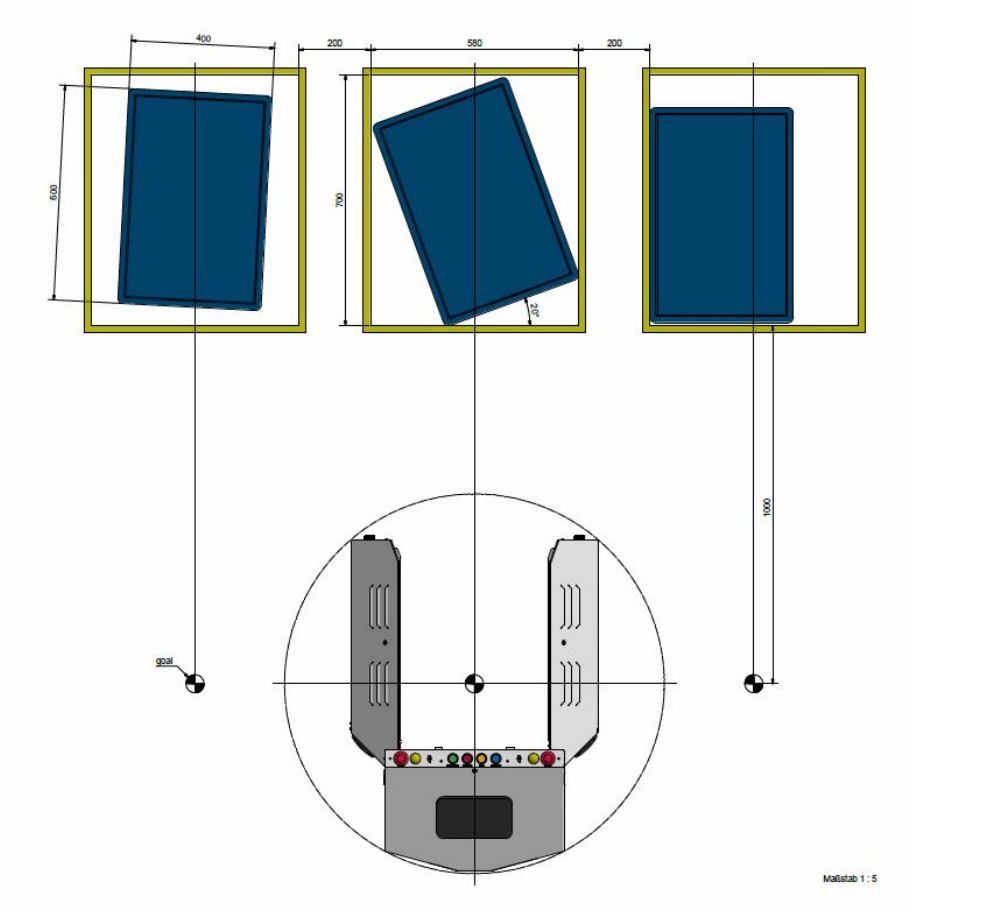


FIGURE 6.12: Warehouse specifications.

like variable size arrays for example, the algorithm does not know anything about the number of objects it is detecting and about the number of boxes being in the environment. In fact it is impossible to predict in advance the number of objects and most importantly how big the array containing the object to analyze should be. Furthermore, the data provided by the laser scanner is of course in relative coordinates and thereby any sort of constraint based on the distance from the box does not suffice, as while moving the perception of the distance may be altered,due to rotations mostly.

This is where the tracking algorithm comes into play. Since the task of the robot is to look for a box, in a very specific goal location, the algorithm impose some further restrictions based on the area where the box should be. However, if a box has been identifies, the tracking algorithm, allows the robot to actually stick with it without changing its target during the fine positioning stage. This allows to have multiple objects with the same characteristics laying on the floor next to each other, without the need of a huge distance between them. Therefore, it allows to save even more space in the warehouse,

as the distance between two piles of boxes is just the minimum required for the robot to to get through in the worse possible misalignment scenario.

6.4 Code generation and adaption

One important aspect, during the testing and implementation phase was to transfer the whole detection algorithm into the PLC. This involves actually changing from Matlab programming language to the structured text one. Although, as it has been mentioned before, Matlab has the capability to transform and generate the code in a readable way for the PLC, there were still some issues, which strongly affected the overall performances of the system.

The most influencing factor was the amount of useless conversion in the generated code, as well as the type mismatching and the different indexing of arrays. Basically, the whole purpose of code generation was just to use Matlab libraries and simulation capabilities to come with the general idea and make quick tests in the PLC to see whether it could work ad be implemented correctly. However, it has been decided, to almost entirely rewrite the code, importing some Matlab libraries and functions, in order to have a system with better efficiency.

6.5 Driving blind

The sensor had a major role in the fine positioning of the robot. However, it is positioned on the Robot right arm and despite the fact the its detection angle can be increased up to 270°, there is a huge limit represented by the other arm. In particular this limit is reached in situation where an harder turning from the vehicle is required in order to reach the right position.

The harder the robot turns the more the left arm could get along the way to the box, cause the robot to not see the box at all and thereby to either stop moving at all or to target another box. This only happens however when the angle formed by the box with respect to x axis of the robot is bigger than 15°.

In order to address the issue it has been decided to decompose the motion in a series of rotations and translations in order to allow the robot to reduce

the misalignment before the fine positioning, exploiting odometry through the encoders in the wheels. During this stage the scan is disabled immediately after detecting the strong misalignment. This is due to prevent the detection algorithm to detect or switch box during this stage. Immediately after reaching the minimum distance point between the center of the robot and the box, the communication with the scanner is enabled again and the fine positioning begins.

6.6 The controller

The controller plays an important role when it comes to fine positioning. In particular the controller receives inputs from the algorithm, in term of coordinates, of the center of the side of the box, which are then used in real time to compute the necessary adjustments in the trajectory. Although the motion of the robot can be expressed in term of v and ω the controller mainly acts on the angular velocity. Indeed, v is regulated by the following law:

$$v = \begin{cases} K & \text{if } v > h \\ Kd(t) & \text{if } v \leqslant h \end{cases} \tag{6.1}$$

where:

- K is a constant value (60 mm/s).
- h is a threshold value that is calculated from the distance.
- $d(t)$ is the distance from the center of the robot to the center of the virtual line representing the box.

V is, therefore, regulated by a simple proportional control that aims to reduce the velocity, when getting closer to the box in order to be more accurate.

For the angular velocity two main strategies have been tested: a PID and two switching PDs. Before going into detail it is also important to mention an important factor when it comes to control ω. Using any of these mentioned

techniques implies that the angular velocity is depending on a constant multiplied an angle. This might be a problem as multiplying an angle would case it go out of $[0, 2\pi]$ and this may cause the robot to spin more than what it is needed as the every angle outside $[0, 2\pi]$ can be actually mapped back. Furthermore, angle outside that set of angles may cause issue in terms of safety as the robot behavior becomes unpredictable.

To correctly address the issue, almost every programming language as a function to map back the angles and it is often called atan2. In this particular case, however, structured text does not have this particular function. In fact, it was necessary to import it from the Matlab libraries.

The atan2 function accepts as arguments the sin and cos of the angle and it is able to map back the angle in a 2π interval that is $[-\pi, \pi]$. This function is defined as it follows

$$atan2(y, x) = \begin{cases} arctan\left(\frac{y}{x}\right) & \text{if } x > 0, \\ arctan\left(\frac{y}{x}\right) + \pi & \text{if } x < 0 \text{ and } y \geqslant 0, \\ arctan\left(\frac{y}{x}\right) - \pi & \text{if } x < 0 \text{ and } y < 0, \\ +\frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0 \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0 \\ \text{undefined} & \text{if } x = 0 \text{ and } y = 0 \end{cases} \tag{6.2}$$

As it has been already said, the controlled variable for the system are v and ω in such a way that

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{6.3}$$

Basically, u_1 and u_2 are the input of the system. However, i order to make things easier it has been decided to split the problem. On one hand, the linear velocity is either kept constant or controlled exclusively by a proportional action, as the closer the vehicle gets to the box the more it is desirable to have smaller v. On the other hand the ω is controlled by a regulator function of u_2.

The next interesting step is how to choose u_1 and u_2. The first one is easier and u_1 is written is explicit form in eq. 6.1. In order to decide the other input u_2 things are a little bit more complicated. First of all, according to the data coming from the algorithm it is possible to get the pose of the box in relative coordinates with respect the RF having its origin in the robot center of gravity of the robot. Then, the problem can be decomposed again in terms of position and orientation. For the position, considering $\bar{x}$ as the middle point of the box side, it is possible to write that :

$$\omega_1 = -K_1 \bar{x} \tag{6.4}$$

On the other hand, if ϕ is the angle between the box and the $x - axis$ of the robot RF, then:

$$\omega_2 = K_2 \phi \tag{6.5}$$

At this point it would be possible to sum the two contribution obtaining:

$$\omega_{tot} = \omega_1 + \omega_2 = -K_1 \bar{x} + K_2 \phi \tag{6.6}$$

However, this approach may not be the best as it is highly desirable that the contribution given by ω_1 is bigger at the beginning of the fine positioning, whereas ω_2 is negligible and, then, to have the opposite situation when the robot gets closer.

For this reason , it was decided to adopt a switching strategy where basically there are two different controllers. This allows to consider the different contribution to the angular velocity according to what it is needed.

6.6.1 PD

The final regulator features a PD controller. The reason why it was chosen to add a derivative action, aside from the simple proportional action, is to speed up the settling time of the system. Indeed, without the derivative action the system has several small oscillation around the regime value, which cause the system to be unnecessarily slower. The derivative action allows the system to have a faster response and although it is true that it makes the system more exposed to disturbances, the trade-off is totally worth it as these uncertainties are negligible.

The final regulator is basically a switching PD that is subject to the distance from the box. When the robot is close enough such that only the orientation is needed to be adjusted then the contribution coming from ω_2 is the dominant one, whereas usually ω_1 dominates, which leads to the following control law:

$$\omega = \begin{cases} K_p^1(u_2^1 + K_D^1 u_{D_2}^1) & d > h \\ \\ K_p^2(u_2^2 + K_D^2 u_{D_2}^2) & d \leq h \end{cases} \tag{6.7}$$

where:

- $u_2^1 = -\bar{x}$
- $u_2^2 = \phi$

- $u^1_{D_2}$ is the discrete derivative computed according eq. 5.18 of u^1_2
- $u^2_{D_2}$ is the discrete derivative computed according eq. 5.18 of u^2_2
- $K^{1,2}_p$ are the proportional constants of the regulators.
- $K^{1,2}_D$ are the derivative constants of the regulators.

In Table(6.1) it is possible to find all the parameters that have been chosen to control the robot in the attempt to solve the task. This parameters have been derived from the testing phase in the attempt to make the fine positioning also shot and stable. Indeed, considering also the driving blind stage which occurs in the attempt to help the robot in accomplish its task, the completion time for the fine positioning lays between 40/65 seconds, including the loading time. Once the controller and the control laws have been estab-

K_v	velocity constant	60 mm/s
h	switching threshold	0.6 m
K^1_p	proportional constant for u^1_2	1.7
K^1_D	derivative constant for u^1_2	3
K^2_p	proportional constant for u^2_2	1.5
K^2_D	derivative constant for u^2_2	0.4

TABLE 6.1: Controller Parameters

lished and implemented in a such a way that the robot reaches the desired position with a negligible error, the next natural step, is of course the loading and unloading stage.

6.7 Loading and unloading

The loading and unloading stage works with the exact same mechanism used when the robot is driving blind, which means exploiting the wheels encoders. During the loading stage, the robot moves at a certain speed that allows it not to introduce enough errors to actually fail. Furthermore, in the bottom left part next to the vertical body a sensor is installed which signal when the box is fully loaded. Besides that, if a fault may happen and the box is not loaded correctly or missed, the robot is programmed to stop as soon as it realize that

something went wrong. Finally, the unloading stage is actually much easier as there is no need for a particular sensor and as soon as the command is sent to the servo motor it proceeds with the unloading procedure, which can also be performed much faster compared to the loading one as there is no need of being that accurate.

Chapter 7

Conclusions and future work

7.1 Results and achievements

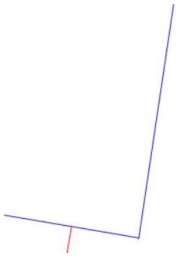
The main purpose of this work was to explore the possibility to develop an algorithm that can detect a specific object in the ground, in order to guide the Stacklifter for the fine positioning phase, trying to overcome the limitations coming from a 2D workspace and the PLC.

Although current literature offers a variety algorithm for object recognition, these are base on a 3D workspace, featuring 3D sensors cameras or a combination of those, which is, indeed, more expansive. Furthermore, these algorithms also require an high computational effort that is much higher than what a PLC can usually handle. For all of these reasons it has been decided to explore and test new solutions that could reduce the computational effort as well as costs, which led to development of the final algorithm.

The novelty of the approach is based upon theoretical concepts, such as linear regression, that are normally used in statistics and economics to organize data and show trends. These concepts have been instead applied to the data provided by the LiDAR sensor for data processing. In particular, linear regression, which is mainly used to studies the relationship between two continuous variables, has proven to be effective to fulfill the task of decoding and filtering the data, allowing not only to drive and to recognize boxes but also to decide which pile of boxes should be lifted.

The reason behind the choice of this technique came organically after analyzing how the sensor perceives a box. Usually, there are two major shapes that describe the box in a 2D environment and it could be either a straight line or two lines with a 90° angle between them (that has been name "L-shape"). Of course, these particular geometries do not suffice alone to determine whether a set point is describing the box, despite the fact that the geometry might be the same (as walls and other similar objects have similar shapes). This led to decision to exploit other additional data available such as the dimensions of the box and use them as a way to distinguish between set of points. This is where linear regression performs its magic as it allows to filter and de-noise the point cloud is such a way that measurements are more accurate and reliable and that allows the estimation of the middle point of the box width, later used to drive the robot towards it. However, linear regression is not the only statistical tool that has been used, as in order to preserve its purpose in this algorithm it needs to know in advance whether a set of points is a line or a set of lines. Pearson's correlation coefficient is a statistical index, that normally measures how strong the correlation between two variable is, but that it this case it acts as a decision factor, that can lead to further analysis or to discard that set. The effect of such a process can be seen in Fig.7.1b, where the algorithm successfully filter the point cloud.

(A) Raw data describing a box



(B) Processed data.

Another important part to successfully perform the fine-positioning is controlling the motion of the robot towards the goal. In order to direct it towards the box, then, two major techniques have been used. The first one is a PD regulator that controls the angular velocity, whereas the linear velocity is controlled by a simple proportional action, that allows the robot to drive

towards the middle point of the box's width and to pick it up. However, there are certain situations in which, according to the box orientation with respect the robot, which may be forced to turn more than usual, causing the other arm to cover the visibility field of the scanner, which may cause the box to be "lost". This is when the second technique is used, which is based on internal sensor such as the encoders. These allows to control the servo motors and by means of the tracking algorithm, to keep driving when the box is lost, for a brief moment , until the box is detected again.

The algorithm was then tested simulating a possible scenarios in facilities with multiple stacks, tolerating a maximum orientation of $\pm 20^{\circ}$ with respect the ideal position. Also, the specification for the dropping out location is a rectangle of 700x560 mm and the distance between each rectangle has been set to 200 mm, which is needed to provide enough space to the robot to pick the box in the worst case in terms of orientation. However, the fine positioning algorithm still succeeds when the angle is bigger, as the robot exploits encoders to drive to a new location that allows to accomplish the task. Most importantly, The distance between the box and robot's goal location is fixed around 1m and task's completion time is around 40 and 70 seconds, as activating the encoders and re-positioning requires extra time. Finally, the robot aims to reach the center of the box width, which is estimated with an error that is less than 1 cm, whereas the actual distance from that point is around 2 cm at most[1]. This does not represent a problem as the maximum tolerance considering how big the robot's arm are is actually 5 cm.

7.2 Next steps

The success of this case of study can be used a base for future improvements for the system itself as well or to introduce the same idea for different purposes. Indeed, The research will continue focusing, in particular, on improvements in terms of the space the AGV needs as well as the speed to accomplish the task as better performance could increase its value on the market as well as the productivity of the whole system. Not to mention that the reduction of the space needed for the fine positioning is actually very important as it

[1]Fig.7.2 Shows the distributions of the values coming from the measurements of the box width, taking into account one hundred samples. At it is possible to see the values assess around 0.37m , due to the fact that in the box is actually slimmer in the middle and therefore, the length is shorter.

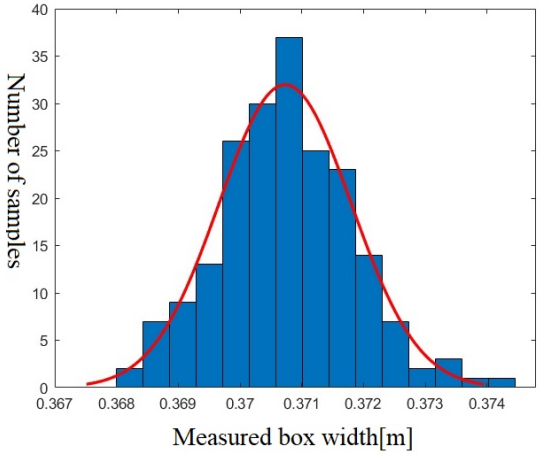


FIGURE 7.2: Measured values over 100 samples.

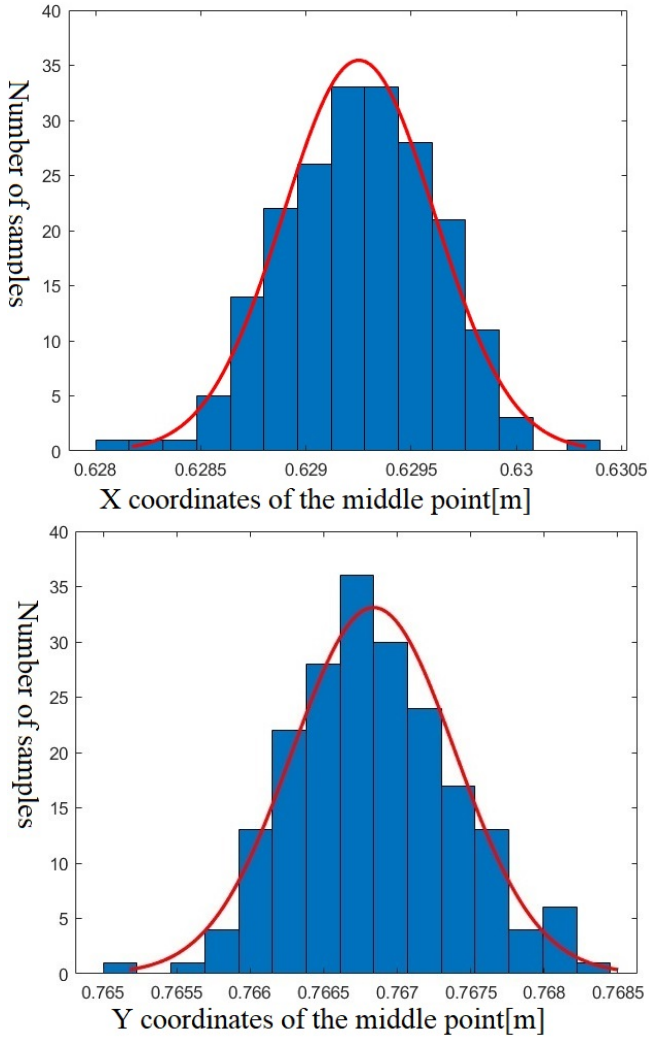


FIGURE 7.3: Middle point coordinates fluctuation over 100 samples.

means cutting off the facility costs remarkably allowing to store more stacks, as the expenditure due to its dimensions has huge impact in the overall costs

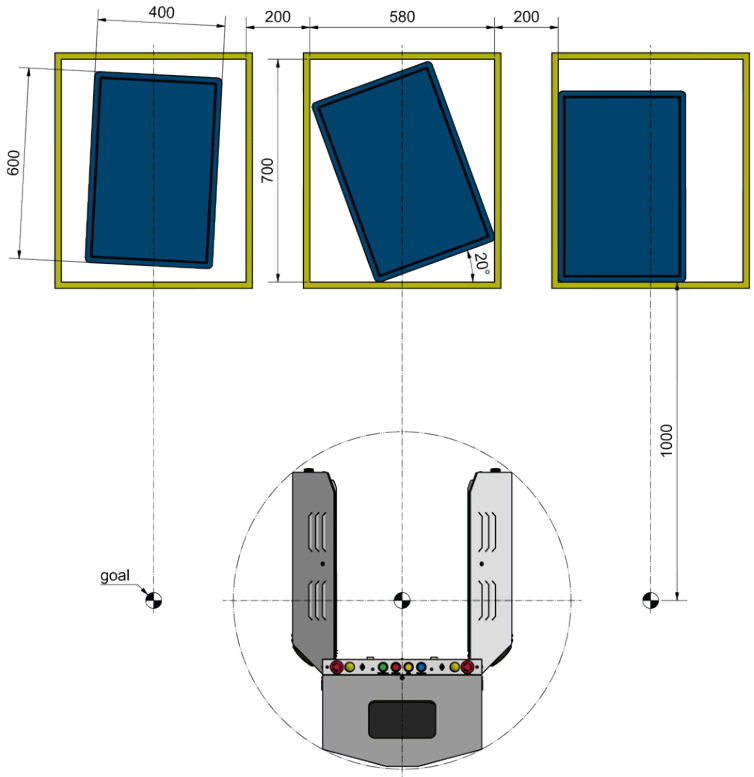


FIGURE 7.4: Specifications.

for these kind of systems.

The company is also considering to implement this algorithm or at the least use the same philosophy in order to develop other similar solutions, involving different kind of objects and shapes. This would open new paths to explore for other customized solution according to customer demand. This is why there is still some work going on with the project. In particular, the algorithm is currently being optimized in terms of robustness and speed and at the same time there are also still researches going on to explore the possibility of extending the same working principle to other objects for similar tasks(currently testing on trolleys).

Appendix A

Sick TIM561 - Technical details

TABLE A.1: Long table caption.

Technical details	
Lots of lines	like this
Application	Outdoor
Light source	Infrared (850 nm)
Laser class	1 (IEC 60825-1:2014, EN 60825-1:2014)
Aperture angle	Horizontal 270°
Scanning frequency	15 Hz
Angular resolution	0.33°
Working range	0.05 m ... 10 m
Scanning range	8 m
Response time	Typ. 67 ms
Detectable object shape	Almost any
Systematic error	± 60 mm 1)
Statistical error	20 mm
Integrated application	Output of measurement data

Technical details	
Ethernet	TCP/IP
USB	Micro USB f
Switching inputs	0
Switching outputs	1 (PNP, "SYNC"/"device ready")
Optical indicators	2 LEDs (ON, "device ready")
Operating voltage	9 V DC ... 28 V DC
Power consumption	Typ. 4 W
Housing color	Gray (RAL 7032)
Enclosure rating	IP67 (IEC 60529:1989)
Protection class	III (IEC 61140:2016-1)
Weight	250 g, without connecting cables
Dimensions (L x W x H)	60 mm x 60 mm x 86 mm
Object remission	4 % ... > 1,000 % (reflectors)
Electromagnetic compatibility (EMC)	IEC 61000-6-3:2006+AMD1:2010
Vibration resistance	IEC 60068-2-6:2007
Shock resistance	IEC 60068-2-27:2008
Ambient operating temperature	–25 °C ... +50 °C
Storage temperature	–40 °C ... +75 °C
Ambient light immunity	80,000 lx
Technical details	

Appendix B

Beckhoff CX9020

Technical data	CX9020
Processor	ARM Cortex™-A8, 1 GHz
Number of cores	1
Flash memory	512 MB microSD (optionally expandable), 2 x microSD card slot
Internal main memory	1 GB DDR3 RAM
Persistent memory	128 KB NOVRAM integrated
Interfaces	2 x RJ45 (Ethernet, internal switch), 10/100 Mbit/s, DVI-D, 4 x USB 2.0, 1 x optional interface
Diagnostics LED	1 x power, 1 x TC status, 2 x flash access, 2 x bus status
Clock	internal battery-backed clock for time and date (battery exchangeable)
Operating system	Microsoft Windows Embedded Compact 7, English
Control software	TwinCAT 3, TwinCAT 2 PLC runtime, NC PTP runtime
I/O connection	E-bus or K-bus, automatic recognition
Power supply	24 V DC (-15 %/+20 %)
Current supply E-bus/K-bus	2 A
Max. power loss	5 W (including the system interfaces)
Dimensions (W x H x D)	84 mm x 99 mm x 91 mm
Weight	approx. 590 g
Operating/storage temperature	-25…+60 °C/-40…+85 °C
Relative humidity	95 %, no condensation
Vibration/shock resistance	conforms to EN 60068-2-6/EN 60068-2-27
EMC immunity/emission	conforms to EN 61000-6-2/EN 61000-6-4
Protection class	IP 20
Approvals	CE, UL, Ex, GL, IECEx
TC3 performance class	economy plus (30); please see here for an overview of all the TwinCAT 3 performance classes

TABLE B.1: PLC datasheet

Bibliography

[1] Dewan Mohammed Abdul Ahad, Amzad Ali Sarkar, and Mohammad Abdul Mannan. "Mathematical Modeling and to carry out a prototype helpmate differential drive robot for hospital purpose". In: *American Academic & Scholarly Research Journal* 6.6 (2014), p. 1.

[2] David Arthur and Sergei Vassilvitskii. "How slow is the k-means method?" In: *Proceedings of the twenty-second annual symposium on Computational geometry*. ACM. 2006, pp. 144–153.

[3] Pankaj Bongale, Anurag Ranjan, and Sahil Anand. "Implementation of 3D object recognition and tracking". In: *Recent Advances in Computing and Software Systems (RACSS), 2012 International Conference on*. IEEE. 2012, pp. 77–79.

[4] Mikell P Groover. *Fundamentals of modern manufacturing: materials processes, and systems*. John Wiley & Sons, 2007.

[5] Insystems Automation GmbH. *Products decription*. 2018. URL: `https://www.proant.de`.

[6] James MacQueen et al. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[7] NOAA1. "LIDAR—Light Detection and Ranging—is a remote sensing method used to examine the surface of the Earth". In: (June 4, 2013).

[8] Renishaw. *Optical encoders and LiDAR scanning*. 2018. URL: `http://www.renishaw.si/sl/optical-encoders-and-lidar-scanning--39244`.

[9] Bruno Siciliano et al. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

[10] Statistics according toh. *AGVs*. 2018. URL: `businesswire.com`.

[11] Alexandru N Vasile and Richard M Marino. "Pose-independent automatic target detection and recognition using 3D laser radar imagery". In: *Lincoln laboratory journal* 15.1 (2005), pp. 61–78.

[12] Allan Zelener. "Object Localization, Segmentation, Classification, and Pose Estimation in 3D Images using Deep Learning". PhD thesis. The City University of New York, 2016.