



universidade
de aveiro

Departamento de Eletrónica, Telecomunicações e Informática (DETI)

Ricardo Jorge Amaral Nestler

**Desenvolvimento de um Protótipo para Colheita Automática de Plantas
Aromáticas**

**Automatic Medicinal and Aromatic Plants Harvester Robot Prototype
Development**

Aveiro

2018



**Ricardo Jorge Amaral
Nestler**

**Desenvolvimento de um Protótipo para Colheita
Automática de Plantas Aromáticas**

**Automatic Medicinal and Aromatic Plants Harvester
Robot Prototype Development**

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob supervisão científica do Doutor Paulo Bacelar Reis Pedreiras, Professor Auxiliar da Universidade de Aveiro, e do Doutor Manuel Bernardo Salvador Cunha, Professor Auxiliar da Universidade de Aveiro.

Dissertation submitted to the University of Aveiro in fulfillment of the requirements for the degree of Master in Electronics and Telecommunications Engineering, under the supervision of Doctor Paulo Bacelar Reis Pedreiras, Assistant Professor at the University of Aveiro, and co-supervision of Doctor Manuel Bernardo Salvador Cunha, Assistant Professor at the University of Aveiro.

I dedicate this work to my family and close people around me that supported me during this work. I want to dedicate this work to myself that contributed to my personal happiness and success in life and therefore to the future people connected to me.

júri / Jury

Presidente / President

Prof. Doutor Pedro Nicolau Faria da Fonseca
Assistant Professor, University of Aveiro

Vogais / Examiners Committee

Prof. Doutor Paulo Bacelar Reis Pedreiras
Assistant Professor, University of Aveiro (supervisor)

Luis Miguel Pinho de Almeida
Associate Professor, Faculty of Engineering, University of Oporto
(Opponent)

Agradecimentos

Em primeiro lugar quero agradecer aos meus orientadores, Paulo Pedreiras e Bernardo Cunha, por aceitarem o desafio de me guiar nesse processo. Quero dar meu especial agradecimento ao meu orientador Paulo Pedreiras, que foi muito dedicado e acompanhou todo o desenvolvimento com muito apoio e paciência.

Quero agradecer a minha família, que me apoiou durante todo este processo. Quero também expressar minha gratidão à minha namorada que, só por existir, me motivou bastante.

Através deste trabalho aprendi muito sobre a profissão da Engenharia, sobre a vida e sobre mim mesmo. Aprendi como muitos dos conteúdos que foram ensinados ao longo do curso são realmente importantes na prática da Engenharia. Aprendi também muito sobre o controlo das emoções, prazos, estimativa de carga de trabalho e sobre o valor do trabalho duro.

O trabalho apresentado é uma porta de entrada para o que espero venha a ser o meu futuro sucesso pessoal e profissional.

Acknowledgments

First of all I want to thank for my advisors Paulo Pedreiras and Bernardo Cunha for accepting the challenge of guiding me through this process. I want to give my special gratitude to my advisor Paulo Pedreiras whom was very dedicated to the project and followed the development very closely with much support and patience.

I want to give my thanks to my family that supported me throughout work time. I want to express my gratefulness to my girlfriend that just for existing motivated me a lot.

Through this work I learned a lot about the technical professional concepts, about life and myself. I learned how many of the content that was taught through the course along the years actually works in practice. I learned much about emotions mastery, workload estimation deadlines for plans and about an increased value of hard work.

The presented work is a gateway for great success in future.

Palavras chave:

Colheita de ervas aromáticas, Eletrônica, Reconhecimento de Imagem, PAM (Plantas Aromáticas e Mediciniais), robô agrícola

Keywords

Tea harvesting, Electronics, Image recognition, MAP (Medicinal and Aromatic Plants), agricultural robot.

Resumo

Este trabalho tem como objetivo desenvolver um protótipo para automatizar o processo de colheita de plantas medicinais e aromáticas. Atualmente existem algumas soluções para este propósito, mas estas apresentam muitas limitações. Assim, o objetivo deste projeto é criar uma solução mais inteligente para o problema. A principal capacidade deste robô é reconhecer a posição das plantas e colocar um dispositivo de corte, acoplado a uma ponte de 3 eixos, no centro da planta, para realizar a colheita.

Esta dissertação relata o desenvolvimento do protótipo, desde o projeto da sua estrutura mecânica até os resultados finais, passando pelos circuitos eletrônicos e software.

Abstract

This work aims at developing a prototype for automating the harvesting process of medicinal and aromatic plants. Currently there are some solutions for this purpose, but they have many limitations. The goal of this project is to create a smarter solution for the problem. This robot's key capability is to recognize the plants positions and position a robotic 3-axis-bridge pointer on the center of the plant for harvesting.

This project follows the prototype development from its physical structure up to the final results, going through the electronics circuit and software.

Index

1. Introduction	1
1.1 Mediterranean Aromatics	1
1.2 Motivations.....	2
1.3 Thesis Overview.....	5
2. Related Work.....	7
2.1 Commercial Solutions Review	7
2.2 Related Solutions Review.....	9
2.3 Evaluation.....	13
2.4 Control Systems.....	14
2.4.1 PID controller	15
2.5 DC Motor	17
2.6 H-bridge.....	19
2.7 Control of the DC Motor with PWM.....	20
2.8 Encoders for DC Motors.....	20
2.9 Servo Motors	21
2.10 Classification and interpretation methods in image recognition.....	22
2.10.1 Statistical based analysis - Bayesian Classifier	22
2.11 Edge detection filters	24
2.12 Hough Transform	26
2.13 Precision and accuracy	28
3. System Architecture	30

3.1 Block specification and functionality	32
4. Implementation.....	34
4.1 Physical Structure.....	35
4.2 Electronic Circuit.....	38
4.3 Software.....	41
4.3.1 Communication between the PC and micro-controller through UART.....	42
4.3.2 Timers setup and PWM generation	44
4.3.3 Encoder position reading.....	46
4.3.4 Axis motors controllers	47
4.3.5 Pixel classification and training.....	47
4.3.6 Image recognition – progress development.....	49
4.3.7 Image recognition – implemented	53
4.3.8 High level program (Top view)	57
4.3.9 Alignment recognition.....	62
5. Results and Analysis.....	64
5.1 Buffer explicit example results.....	64
5.2 Direction servo motors	67
5.3 DC motor for the robot’s traction movement	71
5.4 Axis length according to encoder’s pulses	73
5.5 Controllers and axis motors.....	73
5.6 3-axis-bridge specification	79
5.7 Image recognition – shape validation.....	84
5.8 Image recognition – results with loaded images.....	87
5.9 Image recognition – results on robot version 1.....	88
5.10 Image recognition – results on robot version 2.....	91
5.11 Visual odometry – motion tracking	95
5.12 Image recognition – time efficiency	96
5.13 Alignment Recognition.....	97
6. Conclusions	100

7. Bibliography..... 102

List of figures

Figure 1: Lime-Thyme on the field	3
Figure 2: A vertical cut view of the shape of the result of a usual machine	4
Figure 3: A vertical cut view of the shape of the result of a harvest by hand work or the presented machine..	4
Figure 4: Two Man plucking Machine, by NCC	8
Figure 5: MAP harvesting machine by McLeça.....	9
Figure 6: Basic design of the robotic arm developed in the MAGALI Project [9].....	10
Figure 7: General system scheme concept of the harvester Agribot.[14].....	12
Figure 8: HSV model: hue (H), saturation (S), Value (V).....	13
Figure 9: Open loop concept model [19].....	14
Figure 10: Close loop concept model [19]	15
Figure 11: PID controller diagram.....	16
Figure 12: Schematics of a DC motor [20].....	18
Figure 13: DC motor concept [21]	18
Figure 14: Electric diagram of an H-bridge, in the rest position (left), in the two possible active positions (others) [23].....	19
Figure 15: Representation of the signal on the two channels in the two different directions [22].....	21
Figure 16: Posterior probabilities functions representation for class W1 and W2 given a dataset x [24]	24
Figure 17: (a) plane xy (b) plane mb.....	27
Figure 18: Image domain representation of $x\cos\theta + y\sin\theta = \rho$ [25]	27
Figure 19: Standard deviation plot [27].....	28
Figure 20: System Architecture Top-View.....	31
Figure 21: Systems Architecture	33
Figure 22: General Central decision making block's data flowchart.....	33
Figure 23: Prototype Model.....	34
Figure 24: Picture of the prototype on a printed line of plants	35
Figure 25: Base structure with the moving motors.....	36
Figure 26: 3 axis bridge structure	37
Figure 27: x axis structure of the 3 axis bridge	37

Figure 28: y axis structure of the 3 axis bridge	38
Figure 29: Electric diagram of the overall circuit.....	39
Figure 30: Full-H driver H-bridge configuration with fast diodes in reverse conduction	40
Figure 31: Encoder output circuit.....	41
Figure 32: TCST 2103 top view scheme	41
Figure 33: Buffer representation with the initial and final pointers.....	43
Figure 34: Activity diagram of the buffer management (UML).....	44
Figure 35: Illustration of the PWM states	46
Figure 36: Pixel classification through Sobel filters.....	48
Figure 37: Comparison of the different classifiers	49
Figure 38: Bayes classifier data flowchart (UML).....	49
Figure 39: Centers detection.....	50
Figure 40: Peaks of overlapping rate	50
Figure 41: Starting image	51
Figure 42: Analysis of a blobs shape in a valid plant case	52
Figure 43: Circles recognition through Hough transform	52
Figure 44: Combination of the Hough transform with the developed shape validation system	53
Figure 45: Blob centralization.....	54
Figure 46: Blob clearing, case 1	54
Figure 47: Blob clearing, case 2.....	54
Figure 48: Absolute error function computing, x axis on the left and y axis on the right	55
Figure 49: Radius likelihood function, x axis on the left and y axis on the right	55
Figure 50: Analysis of a blobs shape.....	56
Figure 51: Circle conflict example, case 1	56
Figure 52: Circle conflict example, case 2	57
Figure 53: Scheme of the first solution for the camera position and mapping idea (top view)	58
Figure 54: Simulation of the mapping algorithm with compiled loaded picture part 1	58
Figure 55: simulation of the mapping algorithm with compiled loaded picture part 2.....	58
Figure 56: Simulation of the algorithm with loaded actual pictures from the robot on the printed plants	59
Figure 57: Scheme for the second solution for the camera position (top view)	59
Figure 58: High level main program activity diagram (UML).....	61
Figure 59: Classified image for alignment recognition	62
Figure 60: The plot of one line values of a classified image	63
Figure 61: Demonstration of the array with the largest peaks.....	63
Figure 62: Original image with the recognized alignment drew with a white line and the real in grey	63
Figure 63: Buffer management example 1	65
Figure 64: Buffer management example 2	66

Figure 65: Buffer management example 3	67
Figure 66: Setup of the measurements of the servo's angle	68
Figure 67: Resulted paper sheet with all the measurements for the left side servo motor.....	69
Figure 68: Graph of the angle with standard deviation error for impulse width for left side servo motor	69
Figure 69: Standard deviation graph for left side servo.....	70
Figure 70: Resulted paper sheet with all the measurements for the right side servo motor.....	70
Figure 71: Graph of the angle with standard deviation error for impulse width for right side servo motor	70
Figure 72: Standard deviation graph for right side servo	71
Figure 73: Graph relating the duty cycle of the signal and motor's speed and voltage for the right motor.....	72
Figure 74: Graph relating the duty cycle of the signal and motor's speed and voltage for the left motor.....	72
Figure 75: Graph comparing the speed of both motors in relation to the average applied voltage	73
Figure 76: Motor's response to for a movement of 4, 10 and 100 counts, respectively, for proportional controller with gain of 3	74
Figure 77: Motor's response to for a movement of 4, 10 and 100 counts, respectively, for proportional controller with gain of 6	75
Figure 78: motor's response for the parameters $K=4$, $t_i=0.02$, $y=10$, $w_d=10$	75
Figure 79: motor's response for the parameters $K=4$, $t_i=0.09$, $y=10$, no w_d	76
Figure 80: Motor's response for the parameters $K=4$, $T_i= 0.09$, $w_d=5$	77
Figure 81: x axis motor motion/time statistics	78
Figure 82: y axis motor motion/time statistics	78
Figure 83: Motor response curve in two different cases.....	79
Figure 84: Setup for the 3-axis-bridge measurements	79
Figure 85: Picture to study the components relations pixel-millimeter and coordinate-millimeter.....	80
Figure 86: Measurements results for pixel-mm relation.....	80
Figure 87: Measured coord-mm relation	82
Figure 88: Standard deviation for axis x measurements.....	82
Figure 89: Standard deviation for y axis measurements.....	83
Figure 90: Millimeter sheet used for the coordinates validation	84
Figure 91: Precision (SD) vs Accuracy for x axis	84
Figure 92: Precision (SD) vs Accuracy for y axis	84
Figure 93: Analysis of a blobs shape in an invalid small plant case.....	85
Figure 94: Analysis of a blobs shape in an invalid plant case	86
Figure 95: Analysis of a blobs shape in a false negative case	86
Figure 96 Image recognition results example 1.....	87
Figure 97: Image recognition results example 2.....	88
Figure 98: Image recognition results example 3.....	88
Figure 99: Image recognition, false positive example.....	88

Figure 100: Printed images with plants used for recognition	89
Figure 101: Setup for the accuracy measurements	90
Figure 102: Sheet's segment with the recognized results.....	91
Figure 103: Histogram of x and y axis accuracy errors.....	91
Figure 104: Histogram of absolute accuracy error	91
Figure 105: Setup for the final precision and accuracy results.....	92
Figure 106: Program recognition results display example	92
Figure 107: Comparison of the recognition and pixel classification between normal lab brightness and extra brightness conditions	93
Figure 108: Histogram of x and y axis accuracy errors.....	94
Figure 109: Histogram of absolute accuracy error	95
Figure 110: Accumulative motion measurements of visual odometry	96
Figure 111: Accumulative motion measurements of visual odometry measurements.....	96
Figure 112: Line's offset 'b' vs slope.....	97
Figure 113: Angle correction example.	98
Figure 114: Recognized corrected angle (alignment) vs Accuracy error	99
Figure 115: Recognized angle vs Accuracy error.....	99

1. Introduction

Agricultural machinery to improve agricultural efficiency is a theme that has been going on for many centuries, and nowadays the kind of machinery tends to develop towards automatization of the tasks. There are many kinds of machinery involved in agriculture from the simplest to the most advanced and intelligent.

There are different areas where machinery can be applied in the process of farming [1,2]:

- Preparing the soil (ploughing, levelling and applying fertilizers);
- Fertilizing and Pest control;
- Irrigation;
- Planting/seed sowing;
- Weeding – removal of unwanted plants;
- Harvesting;
- Crop Selection – Crop Sorter;
- Storage.

This work focuses on the harvesting automatization process of Medicinal and Aromatic Plants (MAP).

There is a vast amount of work on harvesting automation solution for plants such as: wheat, potatoes, grapes, rice, sugarcane, corn, carrots, olives, strawberries, tomatoes, apple, orange, etc. Some of these cultures require simpler solutions, while other impose more complex ones. The machine studied in this dissertation is designed for automatic aromatics harvesting.

1.1 Mediterranean Aromatics

Aromatics is the term used for plants with an aromatic propriety that can be used in different fields such as medicine, food, tea and hygiene products. The most produced cultures in the European region are the Mediterranean cultures [3]. In Portugal the main Mediterranean cultures are Pepper mint, lemon verbena, lemon balm, thyme, lemon thyme, and savory. There is a big diversity between farmers that grow other kinds

of plants such as: basil, rosemary, oregano, stevia, salvia/saves, mint, Thymus mastichina, tarragon, marjoram, prince herb, pennyroyal, Urtica dioica, Calendula officinalis, Gomphrena globosa, Echinacea purpurea, Agastache foeniculum, Agrimonia eupatoria, Lavanda, Althaea, Ocimum basilicum.

Aromatics and Medicinal plant farmers use machinery, manual human labor or, more often, a combination of both. In Portugal, the class of plants addressed in this study are usually harvested from the mid of spring until mid of autumn, because they need warm and dry weather to grow.

1.2 Motivations

To have notion of the market size, in the European Union (EU), medicinal and aromatic plants are cultivated on an estimated total area of 70000ha. In 2012 the global trade value was of 102 billion Euros, compared with 173 billion of agricultural products in America, Europe, Brazil and Canada. Germany imported 65000 tons for 201 million Euros. Spain imported 18000 tons of MAPs for 51 million Euros and exported 14500 tons. France imported 19000 tons for 78 million Euros [6].

In a typical Portuguese harvesting day, because of the weather conditions, the work must start in the early morning, between 6-8 am depending on the season. Starting so early allows to get the most profitability from worker payday because after a certain time of the day it become unbearable to carry out manual work at the plantations, especially in the summer, where the temperatures frequently reach 40°C at the peak in the South of Portugal and 30°C in the North. In the north the air is a little more humid, giving similar sensation as in the south when it is hot. After harvesting, the workers have to put the harvested branches in boxes or treadmills to let them dry in a greenhouse or a warehouse. There is usually no need for an electric dry system and most of the producer don't have one. Opening the spaces during the day to let the air ventilate the herbs is enough.

Before entering in the economic perspective, first let's take a look at the plant's biological and quality benefits.

Whether a producer has any kind of machines or not, he has to divide the work in two parts during the day, that may be executed in parallel or alternate. One part consists in harvesting the plants, either using machines or manually. The other part of the work consists in organizing the branches in a warehouse to dry. Nowadays there are many kinds of plants that still have to be harvested by handwork, to be harvested properly. Currently available machines are too rough, cutting these delicate plants in a way where there is too much waste of green matter, while making it hard for the plant to restore itself again for the next harvest. As such, the best way is still use handwork for harvesting.

The diversity of plants is a challenge for automation, as each plant has its size and form. Moreover, some of these aromatics are extremely delicate.

Let's take the example of lime-thyme, which is a much-appreciated plant and it is also one of the top cultures in Portugal, for explaining the importance of the way in which plants are harvested. Figure 1 shows a photo of Lime-Thyme plants on the field and Figure 2 illustrates the comparison of a vertical cut view of the plant

before and after being harvested by a typical machine. In an analysis of the state of the plant before the cut, it can be seen that it gets sunlight from every side and it naturally grows uniformly. After the cut we notice that a vital part of the plant was removed, thus it will take much longer to recover. This happens because the remaining branches, that take energy and carry nutrients from the roots to the leaves, are farther away from the center. Right after the cut the plant still has to feed the side branches so they don't dry and at the same time it has to quickly protect the exposed center from the heavy sunlight that dries it.

Figure 3 compares the shape of a vertical cut before and after a harvest by a worker or by the machine described in this work. This is the optimal scenario, where the plant is cut uniformly so it can grow uniformly again right after the cut. It can apply the energy uniformly, so it grows faster. That way the center of the plant is always protected from the sun, so the plant's health is better and consequently maintain the quality of the leaves.

For the case of ground plants, this difference has an even higher impact because, as the name suggests, ground plants don't grow much in high but instead on the ground. There is a great amount of damage and waste if they are cut horizontally, because the branches that are stick to the ground will cover the floor and the plant will stop growing as fast, as they are most likely to dry, to get old and will constantly get unprotected from the sun after each harvest.

In case of the plants that grow more vertically, the impact is not so relevant, therefore this machine has also a module to cut it horizontally when it is needed.



Figure 1: Lime-Thyme on the field

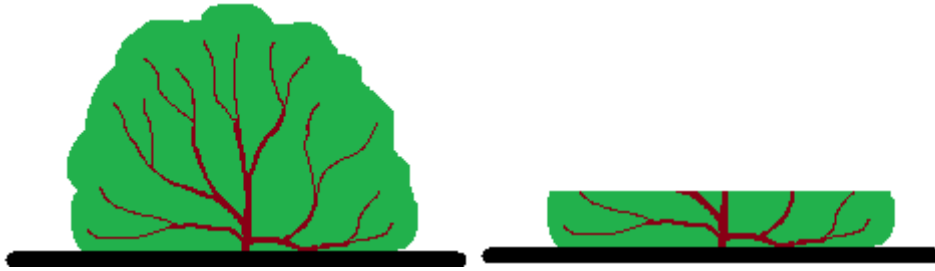


Figure 2: A vertical cut view of the shape of the result of a usual machine



Figure 3: A vertical cut view of the shape of the result of a harvest by hand work or the presented machine

Finally, let's get into the economic evaluation.

For this thesis there was a previous study about the costs and gains around the aromatics business for the plants Verbena and Lime Thyme. The source of the information comes from some Portuguese aromatics reports through a questionnaire that was sent to all the aromatics farmers registered in the EPAM Portuguese website, which got a reply rate of 15%. The content of some of the replies was conclusive enough to perceive that using machines in the harvesting process makes the whole process more efficient and economically viable. Assuming that the costs of each worker is six euros per hour, which is a common value that includes the social insurance too in Portugal, then the average harvesting cost using machines is 0.8 €/Kg and without machines is 1 €/Kg. Similar results appear for the study of the Thyme Lyme culture. For the study of this culture the results of the questionnaire where inconclusive and therefore the data was gathered from one specific plantation, with much detail. The direct conclusions about the economic perspective where that with manual labor the harvesting costs are 1.1 €/Kg without machines and 0.5 €/Kg with machines. This data was taken from a part of the field that is always harvested with a machine, corresponding to the other data from a part of the field that is always harvested with manual labor. Both parts of the field are directly connected with the same conditions of water, lightening and wind. It is obvious that the use of the machines improves the efficiency of the harvesting process significantly, so we can wonder why such a big difference on harvesting cost boils down to a so modest difference in what regards the overall gain. The answer lies in the plant's growing explanation previously described. The plants harvested by hand grow much healthier, with better quality and faster. That means that one area harvested with hand labor results in a larger outcome of good green quality material. A specific study to verify this fact was also made and reported. The study was

conducted in an aromatics field in south Portugal. It consisted in the analysis of the data of some typical harvests of Thyme Lime. The data documented was the number of plants harvested, weight of the plants and the harvest method for each harvest. The total number of plants harvested were 798 by hand and 5994 with machine, in this study. The conclusion was that in the same conditions the plants harvested with a machine had in average 12g per plant available to harvest, while the others had around 16g per plant. A 4g of difference does not seem too much, but in proportion to the collected material, these 4g represent 33% of the 12g plant. There is another interesting fact in this particular regard. Since the manually harvested plants grow healthier and faster, the farmer can even have the possibility to make an extra entire harvest during one year and gain much more profitability from the field. One field can have between two to 5 harvests per year, depending on many factors.

Considering the above discussion, the aim set for the final version of the robot designed in the scope of this dissertation is to combine the benefits and advantages of manual labor harvesting with the ones that a machine can provide, creating an innovative solution for the harvesting process of aromatic and medicinal plants, especially the most delicate ones, such as Thyme Lyme. This robot is an intelligent machine with the ability to simulate the manual cut in the harvesting process.

1.3 Thesis Overview

This project is about the development and study of a prototype of a real machine, realized in a reduced scale. The purpose of this small scale prototype is to develop the basic individual components and make a proof of concept of the machine and its different components.

The structure of this thesis starts with the motivation and market study economic perspective, presented in section “Motivations”. This section studies the reasons for developing this projects in order to justify it.

Before developing something in any area is important to know what alternative solutions exist in the market. The Related Work section presents a review of the various existing solutions on the current market place, in the “Commercial Solutions Review” subsection. The “Related solution review” subsection presents a review of some of the most significant solutions for other kinds of cultures, mainly for fruits. The projects chosen to be covered here are according to the degree of innovation that they brought and the level of the systems intelligence. Thus, this section does not cover pure mechanical solutions.

Section “Basic concepts” explains with detail how the various concepts work. The concepts that were chosen to be covered are the ones that are key components in the development of the prototype. It is important to understand how the various components work conceptually, because they are the technical principles for the development.

Section “Systems Architecture” describes the top view systems architecture of the prototype. By separating the different main components it is possible to get an overview of the system parts. It is important to understand how it works as a whole to be able to understand the development process in the next section.

Section “Implementation” details the development process, starting from the initial basic physical structure, to the electronic circuitry and finally to the software. It covers also the intermediate development steps and failures. This is the most technical practical part of the project, as here it covers the actual technical problems and solutions.

It follows the “Results and Analysis” section, that in some sense extends the implementation section. While on the implementation section only covers the development progress and main results, this section covers all the additional results that helped to take the corresponding development decisions during the implementation. It is here where the importance of this section lies on. When a problem appears, there is an idea to solve the problem, then the solution is designed implemented and tested. During the testing phase there are results that are analyzed in order to validate the solution.

Finally comes the “Conclusions” section, where there is a review of what the robot capable of, its limitations and possible further developments. This section covers the review of the final perspective about the robot’s working state. Two types of future developments are covered, one related to the current prototype limitations, while the second addresses late development possibilities and expansions.




Right at the end there is a list of bibliographic references, which are the main source of knowledge to write the thesis.

2. Related Work

2.1 Commercial Solutions Review

Currently there are diverse companies worldwide that provide machinery for automated aromatic and medicinal plant harvesting. They are well known in this sector and the products they provide are effective, resulting in economic benefits to the farmers. Some of these companies develop and manufacture everything by themselves, from the machine mechanical parts to the software. In general these companies build other kinds of machines, such as planting machines, tree pruning machines (resizing in pyramid, cone and round shape), soil treatment, fertilization, etc. About the tea harvesting machines some are hand-held and others are self-propelling ones.

Table 1 shows some of the more important companies that are present in this market, as well as the type of machinery they manufacture. The prices of the machines vary between 5000€ and 10000€

Company Name	Company Logo	Location	Special Products (PAM)
New Century Corporation		Japan	<ul style="list-style-type: none"> ▪ Hand-Held Tea Plucking Machine ▪ Self-Propelled Tea Harvester ▪ Tea Sniffing and Pruning Machine
MCLeça		Portugal	<ul style="list-style-type: none"> ▪ Aromatic and Medicinal Plants ▪ ZJKawasaki Cut Machinery
Euro Prima		Serbia	<ul style="list-style-type: none"> ▪ Harvester-mowers ▪ Chamomile harvester

Ortomec		Italy	<ul style="list-style-type: none"> ▪ Self-propelled vegetable harvesters on tracks or wheels ▪ Shaker table ▪ Improver Cleaner
Ochiai		Japanese	<ul style="list-style-type: none"> ▪ Hand and self-propelled Tea Harvesters
Harvester Concepts		New Zealand	<ul style="list-style-type: none"> ▪ Tea harvesters: HT – KumaP, HT – Calla, HT – Cress, HT – Nursery

Table 1: Commercial solutions review table

Currently the best solution for small aromatics are the two types of machines shown in Figure 4 and Figure 5. The Two Man Plucking machine type, which is the most common solution of all amongst the PAM farmers, is based on a blade carried by two men and has a ventilation system which blows air with the harvested branches into a bag in the back. The MAP harvesting machine can be carried by one single person but more often is carried and guided by two persons. It has a blade at the bottom that cuts the branches and a mill that lays the branches on a rotating treadmill which, in turn, carries them up until they fall in a bag hanged in the back of the machine.



Figure 4: Two Man plucking Machine, by NCC



Figure 5: MAP harvesting machine by McLeça

2.2 Related Solutions Review

In the automation of the agriculture area there have been significant developments and interesting solutions for the harvesting process. To create solutions for the harvesting problem it involves a wide range of different disciplines, such as horticulture, horticultural engineering, machine vision, sensing, robotics, control, intelligent systems, software architecture, system integration, mechanics, and crop management.

This section presents some of the more advanced robotic solutions and research results for harvesting crops. This review focuses only on advanced automatic solutions.

The Sweet Pepper Harvesting Robot[7], is an advanced solution for harvesting Sweet-Pepper in a greenhouse. It is able to scan a line of crops, detect, recognize the ones ready for harvest and pick them with a robotic arm. It took 4 345 912 Euros to be developed. It runs on a PC running ROS to move the manipulator, process and control the 3D camera and for the LED flash modules. It uses Arduinos to control the moving platform and the knife. The database acquired to train the recognition image processing software was obtained through about a total of 4000 images, in three different greenhouses databases, one artificial database, two imaging sensors, and 14-15 viewpoints for each crop. A deep learning convolutional neural network was applied for sweet-pepper plant part segmentation with the objective to detect the plant main stem and so calculate an obstacle free approach direction for the robot. The success rate is around 60% and takes about 24 seconds to pick each crop.

Strawberry picker from Octinion is an advance solution for harvesting strawberries in greenhouses[8]. It has a robotic arm, with a soft touch gripper designed to pick without cutting the stem and put the strawberry into the final package. It has a success rate is of 70%. It picks one strawberry every three seconds, in average.

MAGALI Project is a project going for years. The basic idea was developed at Montpellier France which originated the first working laboratory prototype in 1984 to pick apples[9]. The concept is based on a system where there is an arm and a camera fixed on a pathway, with a picking tool at the hand. The prototype has a three degrees of freedom, but the arm itself has a straight line movement. Figure 6 illustrates a representation of the model. The camera was positioned in a way where the coordinates of the vision system coincide with the position of the arm in a straight line alignment. The detection workflow would start with taking images, analyze of the data and classify the images in order to recognize the fruits with the central camera. After that the prototype would align itself to the fruit. The picking system would use an effector with a suction cup to detach the fruit from the tree. This prototype had a success detection rate of more than 50% and took about four seconds to harvest each fruit in average. This prototype was very innovative, although it had a main strong weakness called “line of sight” caused by the hand being in the way.

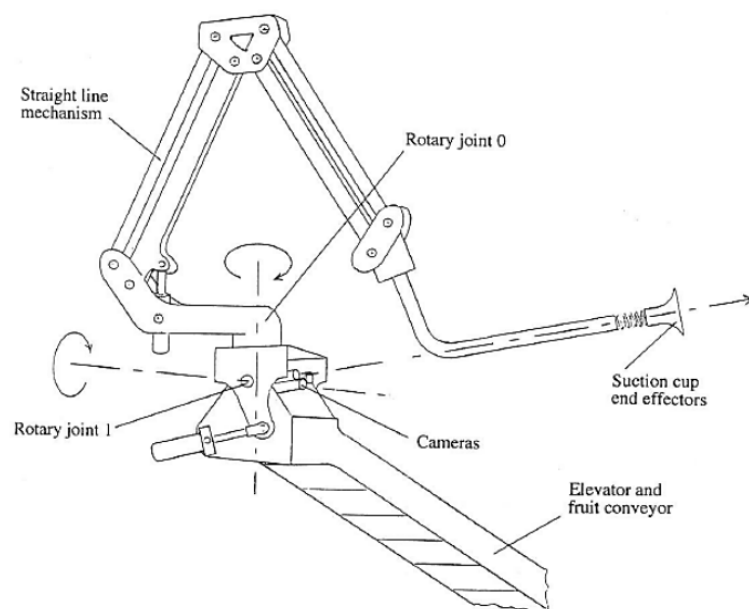


Figure 6: Basic design of the robotic arm developed in the MAGALI Project [9]

Florida Citrus Picking Robot[10] was the following project after MAGALI by another team that addressed the issue of “line of sight” by using a color camera attached at the end of the effector with an ultrasonic transducer to compensate for the distance of the fruit calling this system “eye-in-hand”. The camera would detect and classify the image using the contrast colors between the fruits and the background. The main function of machine vision in this system is to detect the location of the fruit target in the image. The step after is to obtain information of the depth between the effector and the fruit. In reality this step is a challenging problem to solve because of all the external dynamic conditions, and also because it is harder to discriminate the fruit from the background. For this issue this project explored and used a solution based on a

pattern recognition system by visual perception[11]. It uses two-dimensional projections to produce feature abstractions from geometrical properties in order to extract useful information from the scene[12]. This prototype was able to produce results of harvesting each fruit from three to seven seconds with a detection success rate of 75% in average.

Another innovative project is EUREKA[13], whose purpose is to harvest citrus fruits, being composed by an arm with a spherical manipulator, a gripper and a camera at the center of the manipulator. Its machine vision system uses an efficient threshold algorithm applied to the data acquired from a camera with a grey level scheme with a filter of 630nm of wavelength. The obtained result was a detection success rate of 80%. Later they experimented with two other cameras, one with a 630nm and another with 560nm wavelength filters on both cameras. The image classification algorithm uses the Bayesian classifier as discrimination function to detect mature fruits, and the success rate improved to over 90%. The main issues and causes for failure in this solution were leaves that could be in between and occlude the fruits, and the measurement of the distance when approaching the fruit when it was close.

The Agribot Project is a solution designed to combine the human function working with the aid of the machine, to create an aided harvesting strategy[14]. Figure 7 illustrates this system which is composed by vehicle with two picking arms mounted and a human guiding and commanding the vehicle. A human operator sitting in the cabin does the detection. His job is to drive the vehicle and put the picking arms in close range with targeted fruits. Once placed in a good position he is in charge to point a laser telemeter to the targeted fruit, by controlling a joystick that sets a pan/tilt with two-degrees of freedom. This laser measures the distance to the fruit, as well as the horizontal and the vertical angle, determining its coordinates relatively to the vehicle. While the operator marks the fruits, the control computer is in charge to determine the best picking sequence of the arms while they are going for the coordinates and picking the fruits. The main factors that limit the effectiveness, productivity and efficiency are related to the way of dealing with: detection, localization, the structure of the manipulator, the environment conditions and the operation mode. The solution to these problems was focused around developing a specific solution and design for the mechanical structure. They achieved a harvesting time of two seconds per fruit in average.

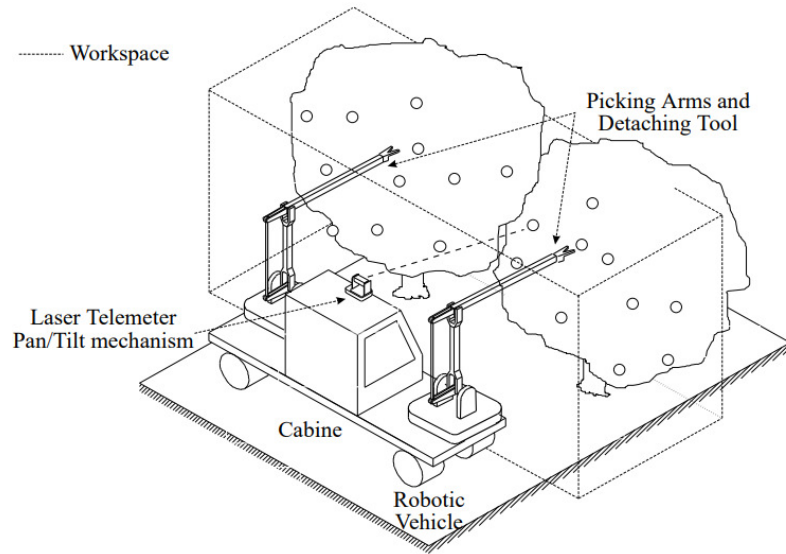


Figure 7: General system scheme concept of the harvester Agribot.[14]

Researchers and collaborators from the University came up with the solution CRAM Citrus Picking Robot[15]. This robot was based on a Cartesian manipulator mounted on a caterpillar to maximize the fruit picking reachability. The machine vision system uses Red and Green colors from the RGB (Red Green Blue) model to detect and distinguish the fruits from the background. The fruit position in the images was determined using the diameter of the classified fruits. The fruits in a cluster are selected in the approaching stage as the manipulator gets closer to the fruit. The distance was estimated based on the movement of the manipulator. With the algorithm Kalman filter it estimates the fruit dimensions. It is an algorithm designed to make exact inferences on a dynamic linear system, and all the space-state variables are continuous and have a normal distribution, both the observed and the non-observed ones. This prototype achieved a picking time of 5.93 seconds in average.

AGROBOT was a solution created in Italy designed to harvest tomatoes. It was composed by a six degrees of freedom picking arm, a hand with a grip and two micro cameras. Regarding the image processing this prototype used the components hue and saturation from the model HSV (Hue Saturation Value) color space to perform the threshold to make the segmentation of the image. Figure 8 illustrates the HSV model.

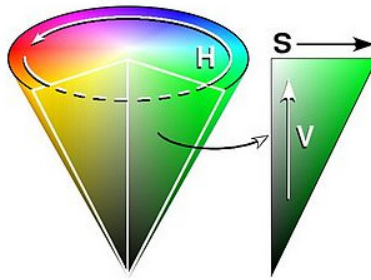


Figure 8: HSV model: hue (H), saturation (S), Value (V)

2.3 Evaluation

This section is destined to make a review evaluation related to the solutions presented in “Commercial Solutions Review” and “Related Solutions Review” section.

The solutions presented in section “Commercial Solutions Review” have all mainly similar limitations. These machines still require handwork in order to operate the machines. The type of solution illustrated in Figure 4 requires two or three workers to operate properly. One supporting each side of the plucking machine and position correctly, while another optional worker can support the bag behind to help, in case the bag gets too heavy or gets stuck to the plants. The type of solution illustrated in Figure 5 uses one or two workers, to guide and pull the machine. The second main limitation is related to the kind of harvest cut. Both have a horizontal cut. Section “Motivations” explains the influence on the type of cut.

In section “Related Solutions Review” the Sweet Pepper Harvesting Robot main limitation is the time it takes to pick each strawberry. In a greenhouse with 20m length and 425 plants, produces around 2550 strawberries per harvest and 5525-11475 in one year (data collected from a greenhouse in non-intensive cultivation in Mira, Portugal). If it would take in average 24 seconds to pick each crop, it takes 17h to make one harvest in this greenhouse. This harvest time would be an accurate estimation if the success rate was 100%. The second main limitation is the success rate of 60% which is considerably low. In the example of this greenhouse it means that on each harvest 1020 crops were left for a second manual harvest.

The solution for strawberry picking provided by Octinion limitation is the success rate of 70%, which may be considerably low.

MAGALI Project, designed to pick apples, has two considered main limitations. One is the success rate of 50%. The other is the “line of sight”, which is the fact that the robotic arm is in the way of the camera used for recognition. These two limitations may be related to one another.

Florida Citrus Picking Robot solved the “line of sight” limitation. The success rate is of 75% which is significantly higher, although there is always room for improvement and can still be considered a low success rate.

EUREKA project was also designed to pick citrus. It has an improved success rate of 90%. The main issues and causes for failure in this solution were leaves that could be in between and occlude the fruits, and the measurement of the distance when approaching the fruit when it was close.

Agribot project, designed to pick citrus crops, combines the human and machine function. Its main limitations are related to the extreme sensibility to external factors. Due to the fact that the recognition and picking phase are separated and may take some time between one and the other. The picking phase works in open-loop. The precision of the mechanical structure may be also a big limitation.

CRAM Citrus Picking Robot limitation is related to the fact that the distance between the manipulator and the fruit is calculated through the diameter of the fruit. Irregularities of the fruits dimensions may affect the success rate.

2.4 Control Systems

The purpose of a control system is to take a certain input signal, or set of signals, and create a desired behavioral on a target system, by means of actuator(s). There are open loop control systems and closed loop systems. An open loop control system is a system that has a controller that takes a certain input signal, and computes an output signal, according to a given model, that is then applied to the actuator. Figure 9 presents a representation of an open loop controller.



Figure 9: Open loop concept model [19]

An open loop controller is simple to implement and to test, but it has several limitations that arise from the fact that it does not take the results of its actions into account to make correction in the control signal. This causes a lot of unwanted consequences such as the frequent presence of an error between the desired output and the obtained one. In case of external or internal noise that changes the output of the system there is no feedback loop to take this into account and correct the control signal. The same happens for natural changes of the environment.

These problems are addressed by closed loop control systems, which have a feedback loop. The controller and actuator concepts are the same as before, but there is a measurement unit which measures the system's output, treats and inserts it into the system again. Figure 10 represents the model of this system. The negative feedback loop allows, for many systems, design a stable system which aims to produce a controlled behavioral. An example of a behavioral window that illustrates how the system might work follows. The objective is to control the angle of a motor. It takes an angle as input and outputs a voltage to the motor as control signal. The motor is at an angle of 50° and the input is at 40°, thus the error between the desired angle

and the current one is -10° , which means that the input of the controller is -10° . Depending on the type of model this controller can have, it takes this -10° error and reacts with a certain voltage at its output that will actuate on the motor with the aim to make the angle equal to the input value and minimize the error.

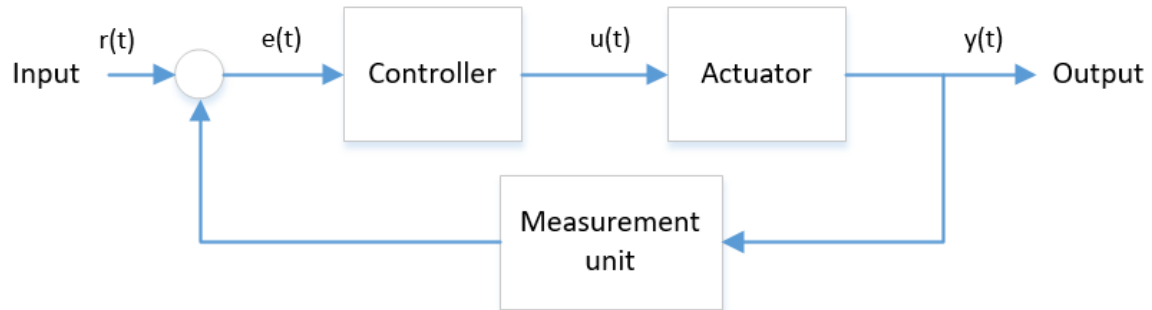


Figure 10: Close loop concept model [19]

2.4.1 PID controller

One of the most used control systems used are the PID controllers, which stands for Proportional-Integral-Derivative [19].

Each one of the parts (Proportional-Integral-Derivative) has a different role, so first let's look into the proportional controller. Figure 10 represents the model of a PID control system with its components. The proportional controller is a controller with a proportional gain applied to the error signal $e(t)$. This means that the expression for the proportional controller can be described as in Equation 1, being $u(t)$ the control signal.

$$u(t) = u_0 + K_p * e(t)$$

Equation 1: Control signal expression of a proportional controller

The error is zero in the stationary state if the gain is infinite or if the signal $u_0 = u(t)$. The first condition corresponds to controller with a similar behavioral as an ON-OFF controller. The second condition implies that the signal u_0 is variable in a way that the error in the stationary state is always zero for any value of the input.

The automatic acquisition of a u_0 variable capable of producing an error zero in the stationary position for any input value introduces to the Proportional-Integral (PI) controller. In system based on a PI controller, there is the proportional and the integral part acting at the same time and the block diagram is shown in Figure 11 without the derivative term. The control signal is obtained by the Equation 2.

$$u(t) = K_p * e(t) + K_i * \int e(t) dt$$

Equation 2: Control signal expression of a PI controller

The control signal depends on the gain of the proportional controller, on the instant value of the error and on the integral value of this error affected by a factor of K_i . The factor K_i can be expressed as K_p/T_i , being T_i called integration time. The expression that describes the error is $e(t) = \frac{u(t)-u_0}{K_p}$, so error in the stationary state is zero if K_p is infinite or $u_0 = u(t)$. The first condition is similar to an ON-OFF controller, the second implies a variable u_0 signal. The automatic acquirement of a variable u_0 is made through the integral influence in Equation 3.

$$u_0 = \frac{K_p}{T_i} \int e(t) dt$$

Equation 3: Expression describing the component u_0 of a PI controller

If the error is positive, the value of the u_0 component is constantly increasing, while if it is negative then u_0 decreases. If the error is zero u_0 maintains a constant value. This behavior eliminates the error in the stationary state.

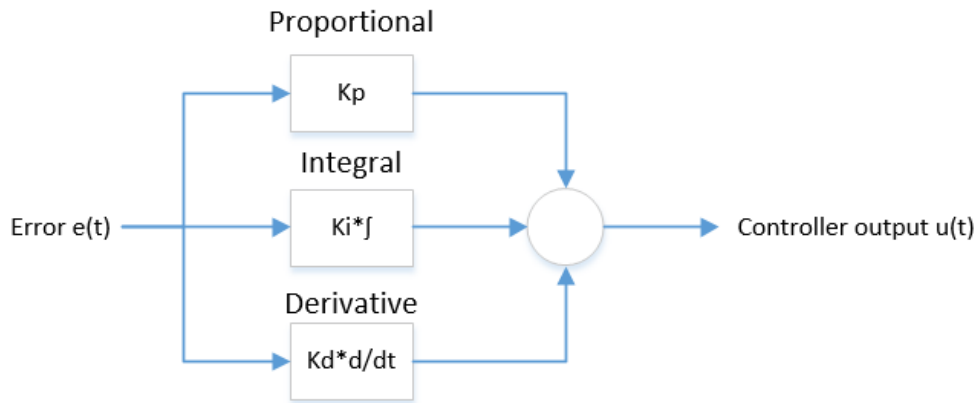


Figure 11: PID controller diagram

In a controller that includes the integral action combined with an actuator that may suffer saturation (real-world actuators have this property), unwanted dynamics behavioral in closed loop may result. For example, a valve saturates when completely open or completely closed. Another example when the control signal is a PWM(Pulse-Width Modulation) signal, then it saturates when the duty cycle is at 0% or at 100%. When the actuator saturates the system changes from a closed loop to an open loop and if the integrator term is continuously accumulating the value of the error signal its output increases in an uncontrollable way (windup effect) until the error signal of its value changes. This behavior can lead to an increased overshoot of the output signal. It has to be noticed too that the integral term has a pole in the origin and therefore is unstable in open loop.

The solution to this problem consists in limiting the integral in order to assure that the system never turns in an open loop system and this involves using an extra feedback loop for the integral term. There are some ways to implement it. In practice a quick implementation is to compare the error with a defined threshold in

which the integral is active. Another is to compare the value of the saturation level and introduce this value divided by a factor and subtract with the gain of the integral term decreasing it.

In a PID control system there is also the derivative component influencing the controller behavioral present in the Figure 11. This component counteracts some of the integral limitations related to the fact that the integral term can't act on the derivative value of the error creating some possible oscillations when the error is close to zero. It basically calculates the derivative value of the error and predicts the error value (useful when approaches zero) and creates a fast error behavior when the error value changes fast. The behavior of a PID controller can be described by the Equation 4 where T_d is designated as differentiation time.

$$u(t) = K_p * e(t) + K_i * \int e(t)dt + K_d \frac{de(t)}{dt} \Leftrightarrow$$

$$u(t) = K_p * \left(e(t) + \frac{1}{T_i} * \int e(t)dt + T_d \frac{de(t)}{dt} \right)$$

Equation 4: Control signal expression of a PID controller

2.5 DC Motor

A DC motor is nothing more than a motor powered up by direct current. The energy can be applied through brushes or it can be brushless. A DC motor can be controlled just by changing the (average) voltage, differently from an AC motor which its speed is controlled by changing the voltage frequency. A DC motor consists in an axis coupled to the rotor, which is the rotating part of the motor. In Figure 12, the stator consists in two magnets, and the commutator has the function to transfer the energy from the power supply to the rotor.

In Figure 13 the stator consists in the magnets (N and S). The rotor is represented by the coil which is powered by the commutator where there is a certain current flowing. By powering the commutator with direct voltage, there is a direct current generated which is transferred to the coil through the contact with the brushes of the commutator with this coil. The commutator function is to make the link between the power supply and the rotor of the motor. It consists in brushes that make the contact with the rotating motor axis. The magnetic field is generated between the poles north and south of the magnets and have a direction starting from the north and going to the south. The Torque is going to boost the coil which boosts the rotor.

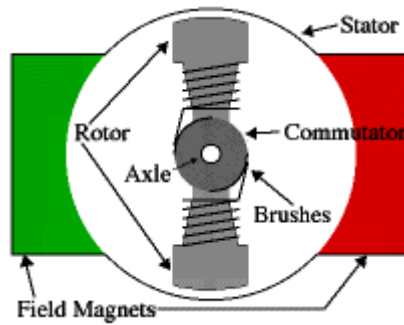


Figure 12: Schematics of a DC motor [20]

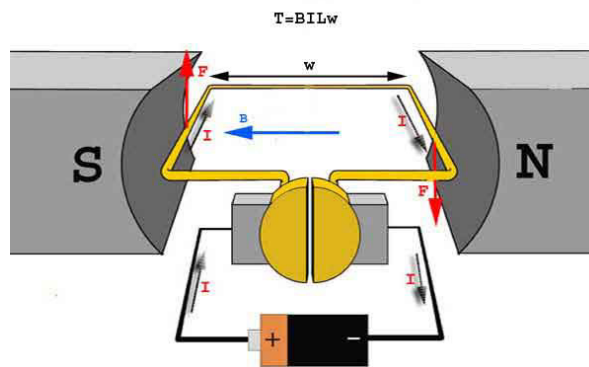


Figure 13: DC motor concept [21]

Basic and derived specifications of a DC motor[28].

-Speed of the axis.

There is a DC voltage (V) applied to the motor in order to make it rotate into a certain direction into a proportional rotating speed (ω). The specifications of the speed of the axis refer to the speed without load in general, which is the maximum speed that the motor can achieve without torque applied.

-Output torque.

The rotation of the axis generates a rotation force called torque. The torque is given in units of force-distance (ex. N-m) it can be of two types: initial/static torque and continuous/dynamic torque. The static torque corresponds to the one to where the motor is stationary and dynamic torque to the maximum value in normal functioning conditions. The torque value is proportional to the induction current.

-Available voltage.

DC motors can be designed to operate in a specific voltage, however most of the times it represents the maximum voltage in which the motor can operate without being damaged.

-Output power.

A common and important specification is the nominal output power (P) which is represented by the product of the torque with the speed. $P = \tau \cdot \omega$. The max output power occurs when the motor is at 50% of its max speed without load and 50% of the stopped torque.

-Dissipated power.

The current absorbed by DC motors are subject to losses and generate heat. The value of these losses are related to the total resistance of the system and include also the losses through friction on the stator and rotor.

2.6 H-bridge

An H-bridge is an electronic power circuit of the type chopper of the class E [23,29,30]. A chopper of the class E converts a constant direct current supply from a power supply, into a variable direct current supply opening and closing gates. That way it is possible to control the direction of the current flow, the polarity of the voltage and the voltage on a certain system or component.

The H-bridge circuit is used to define the direction of the current flow and the value of the voltage in the control of a DC motor. Figure 14 shows the general architecture representation and way of working of an H-bridge. Activating both switches S1 and S4, a positive voltage from the right to the left side of the motor is applied, allowing the current flow from the right to the left. The motor rotates in a certain direction with this configuration. When the switches S3 and S2 are activated, the voltage applied to the motor is the reverse and the rotating direction and current flow too.

When the goal is to break or slow down a motor in movement, the switches S1 and S3 or S2 and S4 are activated and there is a short circuit in one side of the motor. This effect occurs because in this configuration the motor starts behaving like a generator when its axis is in rotating. The necessary torque to maintain or make the motor in rotation raises because the necessary current demanded by the motor stops the movement.

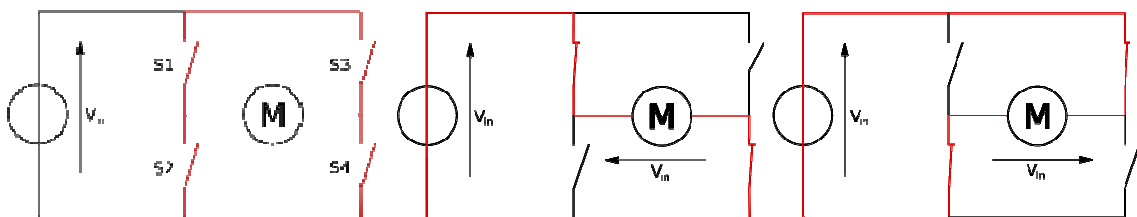


Figure 14: Electric diagram of an H-bridge, in the rest position (left), in the two possible active positions (others) [23]

The switches S1 and S2 or S3 and S4 can't be activated at the same time because it causes a short circuit on the power supply. There are some ways to assure this does not happen to avoid the problem of damaging the circuit. Some solutions are based on protection circuits that avoid both switches to be activated at the same time by using logic gates, specific drivers or any other device or circuit that avoids this situation. In some of these solutions there is even the possibility to add a time difference between the deactivation of one switch and the activation of the other, called dead time.

Another kind of issue is that by activating the motor there are some reverse currents and voltages that can appear and may cause damage to the switches. To avoid such damages there are some measures that can be taken such as using fast diodes in anti-parallel with the switches which is the most common (fly back diodes), and the use of TVS (Transient Voltage Suppressor).

2.7 Control of the DC Motor with PWM

PWM (Pulse-Width Modulation) of a signal involves modulation in duty cycle to transport any information through a channel or to control the power supply value delivered to the load. In power delivery, PWM is used to deliver a certain amount of power to a certain load without the losses that would occur if e.g. the voltage was modified by means of resistors. In a PWM system, the switch (IGBT(Insulated Gate Bipolar Transistor), MOSFET(metal-oxide-semiconductor field-effect transistor) or BJT(Bipolar Junction Transistor)) is used to control the current flow, by conducting or not. This is done with a low voltage range, because the instant dissipated power by the switch gate is the product of the current and voltage in a certain moment. That means that no power is dissipated if the switch was an “ideal” switch. With a sufficiently high modulation rate, simple RC(Resistor-Capacitor) filters are frequently used to soften the impulse train in a stable analogic voltage. This method is usually used in DC motors speed control.

To put the concept in perspective lets imagine a simple gate that turns on and off. When off no voltage and power are applied to the load. Conversely, when the gate is on the voltage is the nominal one and therefore the power delivered to the load is maximum. With the control of the time in which the gate is on or off, the average power delivered to the load is controlled and is variable. Because the motor speed is proportional to the applied voltage, the motor speed can be controlled. For example, if the PWM has a duty cycle of 50%, it means that the gate is half of the time on and the other half off and, by consequence, the power delivered to the motor is approximately half of the maximum. The motor speed has a corresponding speed according to that new average voltage level.

2.8 Encoders for DC Motors

An encoder is an electro-mechanical sensor which transform the rotary motion into electric signals. This functionality is usually used for the purpose of quantify distances, speed, measure angles, number of rotations, etc. [22,31]

An encoder is basically composed by a disc with marks, an emitter and a receptor components. Optical encoders use a led as the emission component and a photo-detector as a receptor. These marks have the functionality to interfere with the light that gets from the source to the receptor. This way when the disk is rotating, the photo-detector that connects to the circuit and generates a square signal proportional to the number of marks according to the resolution of the encoder. The resolution of the encoder corresponds to the number of marks on the disk, which corresponds to the amount of square signals. Encoders can have one, two

or more channels, which can be located with different phases, increasing the resolution and allowing to detect the movement direction. This is how incremental encoders work.

The disk can also have multiple levels of marks on the disk for different channels (bits) making it also possible to read the absolute angle of the disk. These are called absolute encoders.

Detecting technologies can be physically of two types. Can either work using a Led as source and a photo-detector as detector (optical encoders) or using a magnet as source and a Hall Effect sensor as detector.

When using a two channel encoder there are different ways to use the signal of the channels to count the increments, and this affects the resolution. Figure 15 illustrates a representation of the output signals.

Simple resolution uses the computing system to sync with one channel each time the signal rises or decreases and measures the value of the other signal at that instant. This gives a resolution of one count per cycle of one channel.

With a maximum resolution approach the system takes full use of the wheel resolution. Each rise and fall from both channels is counted and detected by the computing system. This gives a resolution of four counts per cycle of one channel.

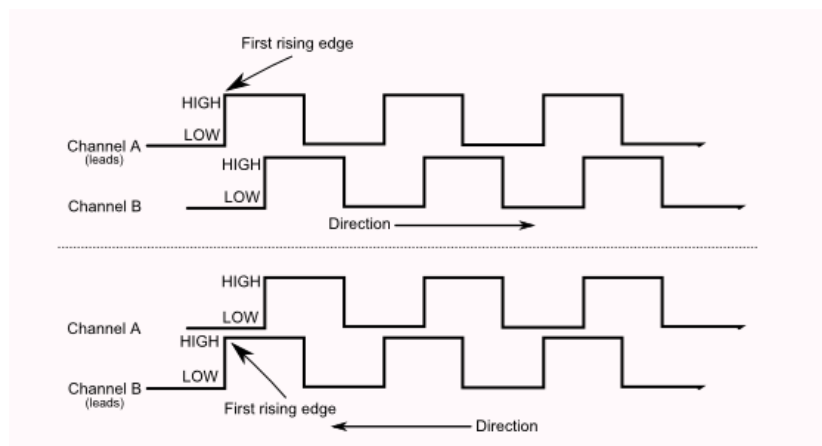


Figure 15: Representation of the signal on the two channels in the two different directions [22]

2.9 Servo Motors

A servo motor is an electro-mechanic system that consists in a rotary linear actuator and has an angular movement proportional to a certain command. As a closed loop feedback control device (as explained in section “Control Systems”), it receives a control signal, verifies the current position and responds accordingly, in order to control the movement going to the desired position. The feedback loop controls this externally, the measurement unit in Figure 10 can either be a tachometer, an encoder or a resolver, depending

on the type of servomotor and application. The most used sensor for servomotors is a potentiometer connected to the servo axis because of their cheapness and reasonable precise and quick behavior. The value of the potentiometer varies accordingly to the motor's angle. In contrast with the DC motors which rotate indefinitely, the axis of the servomotors has a limited and defined freedom. Most of them are limited to 180° degrees, but they are usually precise and quick.

The actuator consists in an electric motor, which usually has a set of gears designed to create a long relation ratio that allows to attain a higher torque.

2.10 Classification and interpretation methods in image recognition

There were many different methods developed for image recognition. They can all mainly be divided in two categories, local based and shape based processing [17]. In general the steps of the image processing are: pre-processing, feature extraction, segmentation, classification of the pixels in classes and interpretation. In pre-processing the goal is to assure reasonable general conditions of the image for further processing, such as brightness and geometry/distortions compensation. Feature extraction is the extraction of properties that can be useful for the next phase and depend on the algorithm used. In this thesis the features extracted were mean and covariance of the pixel distribution. It is a three dimension feature extraction vector for the mean, one dimension extracted from each of the RGB model, and nine dimensions for the covariance (3x3). Segmentation is the process of simplification of pixels with similar properties into “super-pixels” or segments for easier classification and interpretation. Classification is the process of classifying each pixel in different classes with concrete real meanings. For example, a certain pixel in this phase can be classified as a pixel belonging to the class of plant and not any other such as class ground. The interpretation phase is where the image gets real useful meaning, such as the position of the plant for example.

2.10.1 Statistical based analysis - Bayesian Classifier

Many problems in which there are machine learning methods applied can be described as supposing the existence of a particular set of data and a set of different classes and make the question of how to associate the data to the classes. Speaking in a more technical and mathematic language, a particular data x belongs to class C_i . One example of this concept applied to this project is determining for one pixel x if it belongs to the class plant or not plant (which is a composition of non-plant classes).

Equation 5 describes the Bayesian decision theory approaches this problem using the Bayes theory or Bayes rule.

$$P(C|x) = \frac{P(C, x)}{P(x)}$$

Equation 5: Conditional probability of C given x

This describes the probability of C given x, where P(x) is the probability of x and P(C,x) means the joint probability of x and C [24]. Translating this into a practical example, assuming the existence of three classes named Plants, Fabric and Ground. The applied formula would look like in Equation 6.

$$P(Plant|x) = \frac{P(Plant, x)}{P(x)}$$

Equation 6: Conditional probability of Plant given x

This describes the probability of class Plant given the data x. This expression apparently does not lead anywhere as there is no direct knowledge of P(x) and P(Plant,x). The development of this formula would be to dismantle P(Plant,x) into $P(x \vee Plant) * P(Plant)$ and P(x) into $\sum_{C \in [Plants, Fabric, Ground]} P(x \vee C) * p(C)$ by the law of total probability (marginal distributions), ending up with the Equation 7.

$$P(Plant|x) = \frac{P(x \vee Plant) * P(Plant)}{\sum_{C \in [Plants, Fabric, Ground]} P(x \vee C) * p(C)}$$

Equation 7: Conditional probability of Plant given x

In this equation $P(Plant|x)$ represents the posterior probability of class Plants after x data is observed.

$P(x \vee Plant)$ represents the likelihood or class dependent probability, it is the probability of data x if we know that it is a plant. The likelihood is a function of the parameters of a statistical model, given a specific data observed. In other words, the likelihood is a function with a certain model chosen that seems to describe the distribution of the data. This function is created taking the parameters of the corresponding class and the actual data to create the likelihood function. In a sense this can be seen as a comparison of the data distribution form and the theoretical function form with the corresponding parameters. Some examples of models commonly used for the likelihood function are normal Gaussian, Poisson and exponential distribution. Although these functions are the most commonly applied on most of the applications of the Bayesian decision theory, any distribution function can take place, depending on the most suitable case of the type of the real data. In practice this is calculated using the assumption that the given data values are independent from each other, and so like in Equation 8, where x_1, x_2, \dots, x_n represent the pixels in the image.

$$P(x|Plant) = P(x_1, x_2, \dots, x_n|Plant) = P(x_1|Plant) * P(x_2|Plant) * \dots * P(x_n \vee Plant)$$

Equation 8: Probability rule of independent variables

$P(Plant)$ represents the prior probability of the class Plants before any observation was made. The prior probability is calculated based on prior data, training data, which is an overview of the average quantity of each class is usually present. In this case the prior probability of the class Plants is the same as taking into account all the training data, and count the percentage of Plants pixels from the total amount of pixels.

$\sum_{C \in [Plants, Fabric, Ground]} P(x \vee C) * p(C)$ represents the so called evidence, and can also be represented by $P(x|Plants) * p(Plants) + P(x|Fabric) * p(Fabric) + P(x \vee Ground) * p(Ground)$. It does not depend on the tested class and is constant because it is a sum of all possible cases. It does not influence the shape of

the distribution of the posterior probability. It is just a normalized term, which ensures that $P(Plant|x)$ sums up to 1.

Now there are already all the necessary components to apply the algorithm and decide which pixel belongs to the class Plant or a non-Plant class. To achieve that goal, the steps are:

- 1- Compute the posterior probability for each class and compare them to see which one gets a higher probability for each pixel;
- 2 - Compute $P(Plant|x)$, $P(Fabric|x)$ and $P(Ground|x)$;
- 3 - Compare $P(Plant|x_1)$ with $P(Fabric|x_1)$ and $P(Ground|x_1)$, $P(Plant|x_2)$ with $P(Fabric|x_2)$ and $P(Ground|x_2)$ and so on until all the pixels are compared.

For each pixel the higher posterior probability corresponds to the result of the classification for that pixel.

For the calculation of the error associated with Bayesian decision theory, supposing there is a case where there are only two classes, W_1 and W_2 . The chosen class for each data x is the one which has a higher probability $P(W_i|x)$ present in Figure 16. The error of the decision choice for a given data is the value of the post-probability of the other class, $P(error|x) = \begin{cases} P(W_1|x) & \text{if } W_2 \text{ is decided} \\ P(W_2|x) & \text{if } W_1 \text{ is decided} \end{cases}$

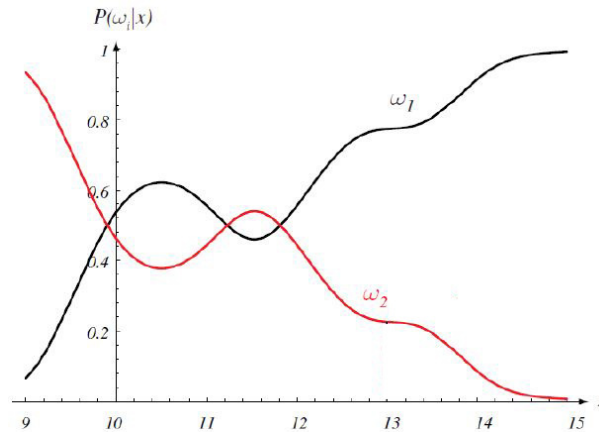


Figure 16: Posterior probabilities functions representation for class W_1 and W_2 given a dataset x [24]

2.11 Edge detection filters

Edge is a frontier between two different regions of color tonality, often used for extracting features. To detect edges and highlight them there is a filter applied on the image. Linear filters soften or highlight certain details of the image, minimizing some noise effects without changing the meaning of the image. Some linear filters can be low-pass band, which soften an image, lowering the values of high frequencies corresponding to fast

changes in color transitions. This filter tends to minimize noises and appears to create the blur effect. An

example of a 3x3 filter would be $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} * 1/9$. There are also the low-pass filters with weighted means,

where the weights are defined according to the distance to the center like $\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} * 1/16$.

High-pass filters are used to highlight details, producing a sharpening of the image, that is, transitions between different regions of colors turn sharper. These filters are useful to highlight edges of objects but they

also highlight noise present in the image. An example of this kind of filter is $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$. These kind of

filters can also be used to highlight edges in a preferred direction of the image. For example this mask

highlights horizontal edges $\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix}$.

The difference between linear and non-linear filters is that non-linear filters change the meaning of the image. Operator Roberts[32] represents a disadvantage because some edges get more highlighted than others

depending on the direction. It consists in the function $a' = \sqrt{(a - d)^2 + (c - b)^2}$, where a is the gray level

that corresponds to the location a. $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$. The mask applied to the image would be $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ for vertical

edges and $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ for horizontal ones.

The Prewitt filter[32] is able to detect edges in more directions with the masks $\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$

and $\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$.

The Sobel operator[32] is able to highlight lines without highlighting isolated points (noise). It consists in

applying two masks, one following the other, $\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$ and $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$.

The Canny operator is better and clearer than the ones previously described but requires more processing power.

Then there are the morphologic filters that explore geometrical properties. For these filters, the masks are denominated structuring elements and appear as ones and zeros. The basic morphologic filters are the

median, erosion and dilatation. To explain this let's take the image $\begin{matrix} 3 & 6 & 5 \\ 2 & 8 & 3 \\ 2 & 6 & 5 \end{matrix}$, apply the structuring element

$\begin{matrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{matrix}$ and calculate the median.

This would result on a value of the operation as a 6 is the median value of the selected numbers of the image by the structuring element [2, 3, 6, 6, 8]. This filter is used to soften the image and eliminate noise without changing the dimension of the image. The erosion morphologic filter creates an erosion of the bright parts, generating darker images. In the previous example of structuring element and image, the result would become a 2. The dilatation filter creates a dilatation of the darker parts of the image, generating brighter images. In the previous example the result of the operation would be an 8.

2.12 Hough Transform

Suppose n points on an image and that the goal is to find subgroups of these points that are aligned in straight lines. A possible solution is to find first all possible lines for each pair of points, then take all the subgroups of points that are located near determined lines. The problem with this procedure is that the number of segments of lines in an image has an order superior to n^4 (number of points n). In 1962 Paul Hough proposed an alternative approach. This approach was initially designed to detect lines and curves, and this method can be used if the analytic equation of the frontier of the objects is known.

The Hough's proposal to detect straight lines in an image is described in Equation 9,

$$y = m * x + b$$

Equation 9

being (x, y) the points in the line and (m, b) the parameters of the line that correlate the points of the different axis. In this case there are an infinite number of lines that pass through (x_i, y_i) , satisfying Equation 10, for different values of 'a' and 'b'.

$$y_i = m * x_i + b$$

Equation 10

However, if the equation is written like in Equation 11, considering the plane mb , denominated parameter space, there is a unique line that goes through the point (x_i, y_i) .

$$b = -m * x_i + y_i$$

Equation 11

A second point (x_j, y_j) is also associated in a line in the parameter space. The straight lines in the parameter space associated to the points (x_i, y_i) and (x_j, y_j) create an interception in mb in the point (m', b') , where m' represents the inclination and b' the point of intersection with the y axis of the line that contains (x_i, y_i) and (x_j, y_j) on the plane xy . Figure 17 illustrates such planes.

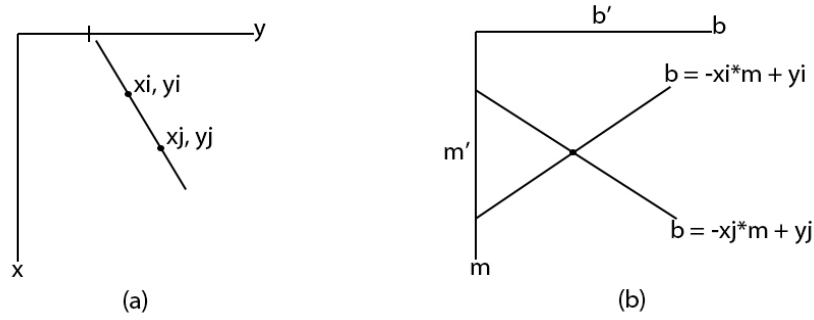


Figure 17: (a) plane xy (b) plane mb

In this way the Hough Transform consists in mapping each pixel of the image in parameters space, which is organized into a n -dimensional accumulator, where n corresponds to the number of parameters. Analyzing each pixel of the image, searching every lines that pass through that point and based on the space parameters map, it increases the position of the accumulator corresponding to the result. This way, it is possible to define a threshold by defining the minimum value of the accumulator that corresponds to a peak in the space parameters map to an object.

The problem of utilizing the general equation in the form of slope-intercepted is that both the slope and the point of intersection are limited, when the line turns vertical, tending to infinite. To overcome this limitation Duda and Hart [18] proposed an alternative parametrization, through the representation of a normal straight line represented in Equation 12 and in Figure 18.

$$x \cos(\theta) + y \sin(\theta) = \rho$$

Equation 12

If θ is constrained to the interval $[0, \pi]$ then the normal parameters of a line are unique. With this constrain each point of xy corresponds to a unique point on the plane $\theta\rho$.

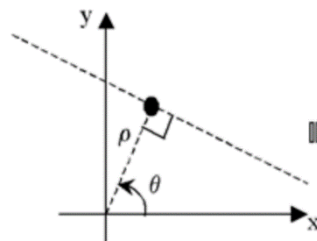


Figure 18: Image domain representation of $x \cos(\theta) + y \sin(\theta) = \rho$ [25]

The next important development of the Hough transform was the application of the concept and algorithm to finding circles, requiring a three dimension parameter space map or array. There are three parameters that index this array and specify the location and diameter of a circle with center $a = (A, B)$ and radius R . The Equation 13 represents this parametrization.

$$(x - A)^2 + (y - B)^2 = R^2$$

2.13 Precision and accuracy

Precision and accuracy are two different parameters that are calculated to characterize the quality instruments. It can be applied to whole experiments and take conclusions about their validity.

Accuracy refers on how close are the measurements from the real value. On the other hand precision refers to how close are the measurements one from another. One way to determine the precision of measurements is to either make measurements in different conditions and check the tendency curve error if the system is linear or through the repetition of the same measurement with the same conditions. The precision is often related to the standard deviation which represents an evaluation of the distribution of the different measurements. In Figure 19 there is a general illustration of a Gaussian distribution with the standard deviation ' σ ' represented. One σ of distance means that 34.1% of the values are located in between the mean of all values and the mean + σ . This means that 68.2% of the values are in a range of 2σ and 94.4% are in a range of 4σ from one another.

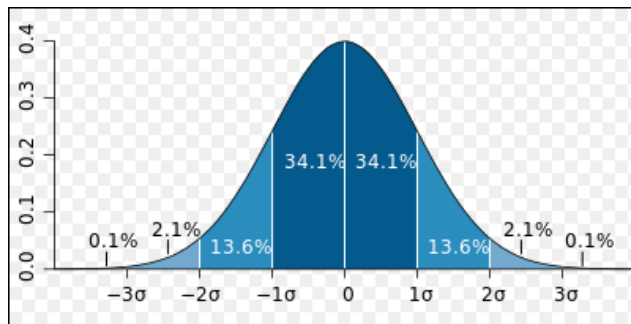


Figure 19: Standard deviation plot [27]

3. System Architecture

The system can be summarized into four main blocks, which have their own functionalities and together create a working robot through the relations between themselves. As represented in Figure 20 there is a central decision making a moving, a 3-axis-bridge and a vision block.

For the purpose of moving the platform above the plants and advancing from plant to plant there is a moving block, which gives traction to the robot and guides it. It is directly connected to the central decision making system and controls the speed and direction at which the robot should move. It requires to be guided and correct the direction as the robot may not always be directly aligned with the plants for the cutting device to reach them. In Figure 20 the moving block relationship with the Central decision making block is characterized as a one way relationship. The data coming on this communication line are the direction and speed commands. This block works in an open loop by itself on a low level because it has not a measure unit to generate and process feedback. However, globally, with the Central decision making and the Vision block, this system works as a close loop system as explained later.

How does the robot know its current moving direction? Where are the plants located on the field? These are all questions that are the responsibility for the Vision block to answer. The main function of this block is to get information about the plants locations on the field. This block is important because the plants are not always in the same alignment, there can be more than one row of plants under the machine at the same time, there may be empty spots on the field with no plants, and besides that, it has to recognize which plants on the field are in proper conditions to be harvested. If there was not any vision block for this purpose, the robot would have to move in a straight rail, the plants would have to be planted with a precision of 100% regarding the distance between them and alignment, there could never be dead plants on the field or damaged plants in recovery and all plants needed to be full grown up at the same time. In reality all these factors are extremely difficult to attain and, even if possible, would need a lot of extra caring and handwork, which is impractical and conflicts with the main goals of this project.

Regarding to the robot's speed and direction, the vision system is responsible to provide the measurement values for the moving block to act on.

In general this block works as an information extractor, basic treatment and interpretation from the field in order to operate and manage the robots behavior. This block is directly connected to Central decision making block, having a data flow communication direction from the Vision block to the Computing block, present in Figure 20.

Even after the robot is positioned above the desired plants it is almost certain that the pointer which carries the tool to harvest the plant does not match the right position, and therefore there is a block composed by a 3-axis-bridge designed for that job. It is called 3-axis-bridge because it can move the pointer in the three space dimensions within the limits of the robot's construction. It has actuators to move the pointer and a control systems to control the behavior. It works in closed loop and is independent from the rest of the robot. In a general way, receives coordinates and moves the pointer on top of the plants in real time. It also can receive the input command to calibrate the axis system, useful at the initialization step. That's why the arrow that relates the 3-axis-bridge and the Central decision making block in Figure 20, points to the 3-axis-bridge.

Finally there is the explanation of the Central decision making block, that is connected to all other blocks. This block provides a certain level of robot's intelligence, being responsible for making the main decisions in order to fulfill the robot's purpose. It receives processed data acquired from the vision system as system input, which are related to the orientation relatively to the plants and the locations of the plants. It manages this data and makes decisions based on it. These decisions affect the other blocks directly, such as the management of the vision system working state, the location given to the 3-axis-bridge and the direction and speed to the moving block.

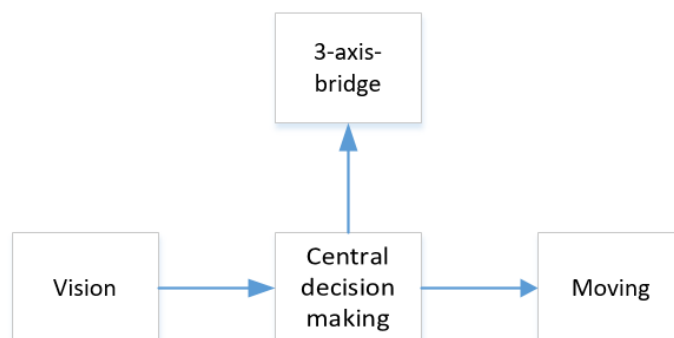


Figure 20: System Architecture Top-View

3.1 Block specification and functionality

The moving block is composed by the components direction and traction, which are composed by two traction motors and two direction motors, respectively connected to four wheels, as present in Figure 21. Regarding the traction component, the motor's speed is proportional to the input signal coming from the Central decision making block. It receives this signal and converts it into a power signal through a power circuit connected to the motor. The direction component has two servo-mechanic systems designed to put the direction motors in the desired angle. The signal to control these systems comes from the Central decision making system.

The Vision block is composed by three main components, as present in Figure 21. The alignment recognition component is composed by a camera pointing forward. This component should provide the robot with the information regarding the robot's relative orientation and position to the line of plants. The algorithm is based on the detected colors alignment. After the orientation is extracted and computed, the information is sent to the Central decision making block.

The robot has two cameras, one pointing forward and another pointing down. The one pointing forward is used only for the alignment recognition component, while the other camera is used by the other components. The next component is the "Image Classification", which is responsible for classifying potential plant pixels from non-plant pixels. This component takes pictures and uses them as input, processes and gets the classified pixels as output. This data is sent and used by the component "Image Recognition". It takes the classified images as input and gets the plants locations as output. It does the detection and recognition of the center of the plants and organizes them into a map of centers. This map is sent to the Central decision making block.

The 3-axis-bridge block has the ability to move a pointer in the three dimensional axis within a certain range. Each of the axis management is independent from each other and has its own control system, as present in Figure 21. The actuators are motors, the horizontal planes axis (xy) motors use a digital designed control system and z axis uses servo-mechanic system. This block receives a certain position as input and has a close loop inside the block to control the motors. This input value combined with the feedback is transformed into a control signal. This control signal powers the motors through a power circuit. In this prototype the z axis servo-mechanic system is coupled to a pen to mark the plants on the ground.

The other components of this block are the calibration systems for the x and y axis. These calibration units are composed by measurement devices placed on one extreme of each axis, that detect when the pointer is passing through that position on the respective axis. When these devices make the detection, the actual coordinates are reseted as zero and serve as the reference origin of the 3-axis-bridge coordinates system. It performs the calibration algorithm when receive the corresponding input command from the Computing block.

Finally, the Central decision making block works as an intelligent decision maker. It interprets all the processed data together and creates context. It makes decisions regarding the robots behavior, specifically the 3-axis-bridge and Moving block, based on the input information received from the Vision block. In Figure 22 there is an illustration of the general data flowchart of the program main block. In summary, when the program starts it begins by calibrating the system, then enters in a loop characterized by its other two main functions. It takes the map, interprets and sends the coordinates to the robot. The next decision is regarding the speed and alignment of the robot, which is made taking into account the received map and the robot's alignment. The map interpretation of this component makes the decision whether there are plants in range or not. The decision regarding the alignment of the robot, is to check the given alignment as from the Vision block as Input and make the appropriate adjustment decision of the angle of the wheel of the Moving block.

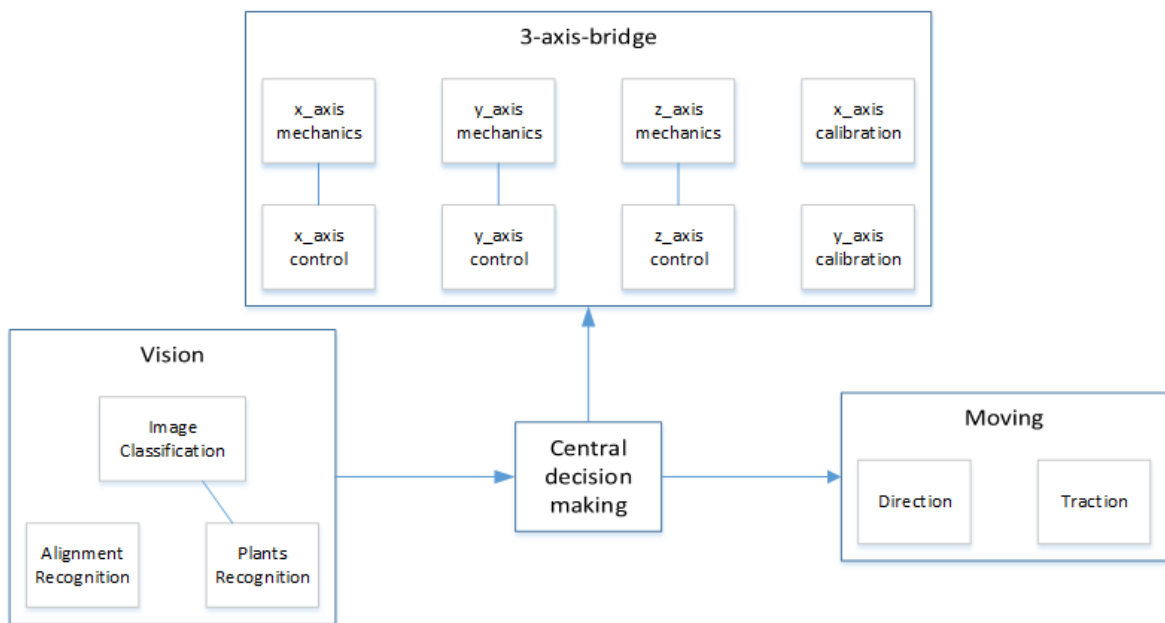


Figure 21: Systems Architecture

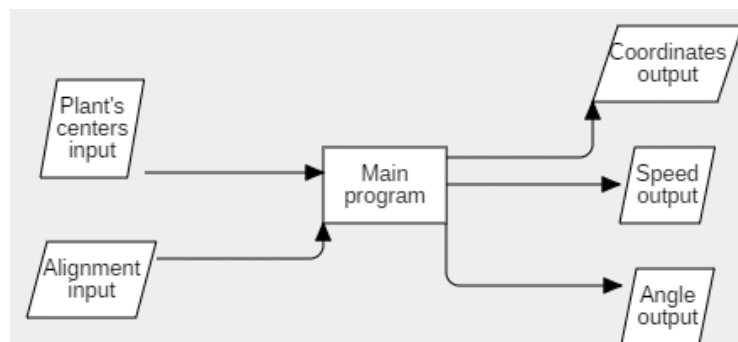


Figure 22: General Central decision making block's data flowchart

4. Implementation

In order to build the real machine it is necessary to build a small scale prototype first to develop the algorithms and make a proof of concept. The development of the prototype is the main focus of this thesis. In Figure 23 there is an illustration of the prototype model, and in Figure 24 there is a picture of the actual built prototype.

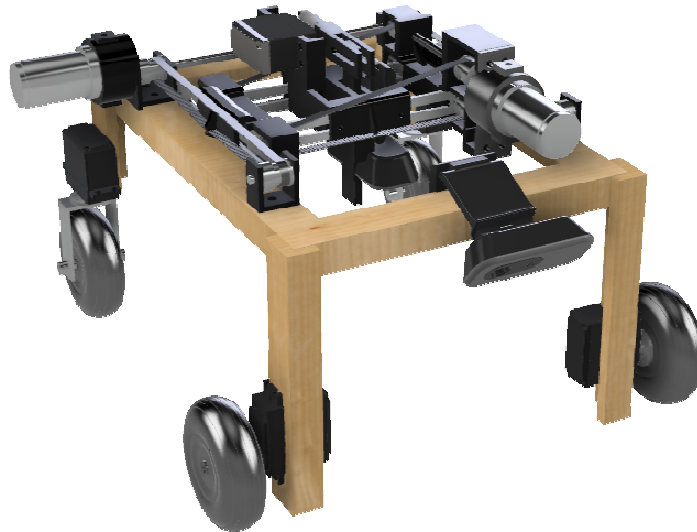


Figure 23: Prototype Model

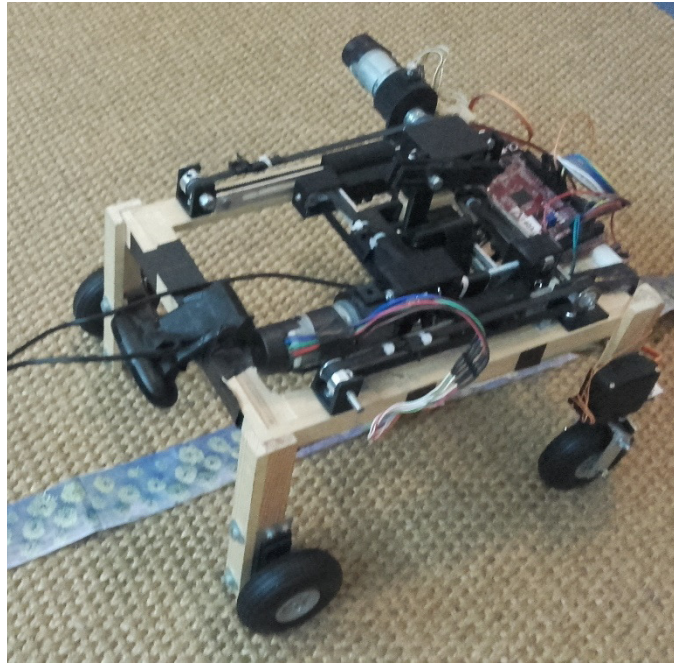


Figure 24: Picture of the prototype on a printed line of plants

4.1 Physical Structure

The physical structure design started with a hand draw on a paper with the dimensions of the base. Figure 25 represents a 3D model of the robot. The base structure is made of pine treated wood. The outer rectangle rim has dimensions of 26.1x19.6cm. The four represented motors are designed to give traction and change direction of the robot. The two traction motors used are DC motors (right side) and the two direction motors are servo motors (left side). The direction wheels used of the aero-modelling kind. The direction wheels support was self-made with aluminum. It was designed in a ‘U’ form with holes to support the wheel’s aluminum axis. The support was furthermore screwed into the servo motor axis. For the base structure was used wood because of its accessibility during the early development stages of this project. Wood is also cheap, resistant enough for the base structure and is easy to mold and work with, allowing also making changes after mounting the pieces in the structure, such as holes and erosion. The direction wheels are positioned in the back of the robot and the traction in the front of it. The direction wheels are positioned in the back because this is a prototype of a model that will have a real scale version too. This is the most common configuration of agricultural light machines because it has better performance in face of irregularities on the terrain.

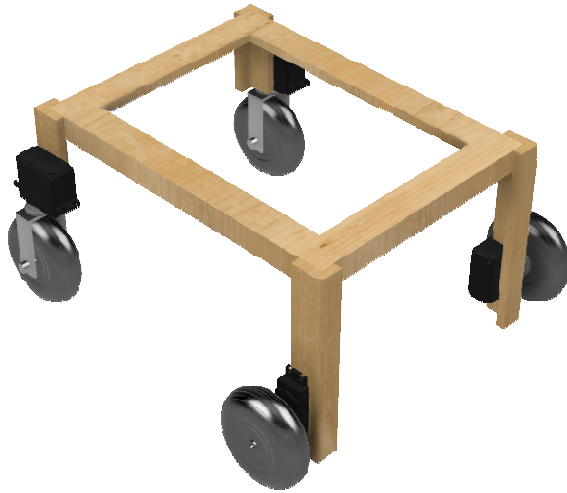


Figure 25: Base structure with the moving motors

In order to build the 3 axis bridge mechanical structure, there was an initial version made of aluminum with steel screws. The main reasons for choosing this material were its extreme accessibility in the early development stages and accessibility of the tools to work with it. The other reasons were its durability, compactness and expected stability and precision. The first version of the 3 axis bridge did not work properly because the stability and precision requirements were not met. The structure had a craft look and would change its form slightly from time to time. The movements between the rails and the axis were also not as smooth as expected. This solution for this phase of the project turned out to be unsuccessful and after a certain waste of time and resources, the project followed a drastic change.

The solution was to introduce the use of the tool 3D printer into the project. It got many advantages relatively to the previous solution such as, the precision of the pieces, custom complex pieces speed of component development and reasonable stability and durability for this prototype's purpose. The pieces, as the entire model itself, were designed in the program Autodesk Fusion 360 and then printed in a 3D printer from Beeverycreative. Not all pieces for the mechanical structure were printed in the 3D printer. The rails of the axis were still made of aluminum because they are thinner and support stronger forces. Using aluminum bars as rails allowed the axis to flow smoothly. Figure 26 represents the final result. In this figure the motors are also represented, one responsible for the movement of each axis. All the black parts are made of plastic, the gears of steel and the rails of aluminum. In Figure 27 there is a representation of a selection of the components that make the x axis. The x axis moves the axis in the forward and backward direction. It has a movement range of 6.5cm with an effective range of 6.2cm. In Figure 28 there is a representation of the y axis. The y axis moves the axis in the left and right direction. Its movement range is of 5.2cm with an effective range of 4.6cm. The gear bands used are made of rubber and connect the motion parts of the bridge. The main motion principle of the x and y axis is based on a motor that has an axis connected to a rubber gear band on both sides of the carrier of the axis. The z axis (vertical) is moved by a servo motor, and it serves only to mark the positions of the pointer on the ground.

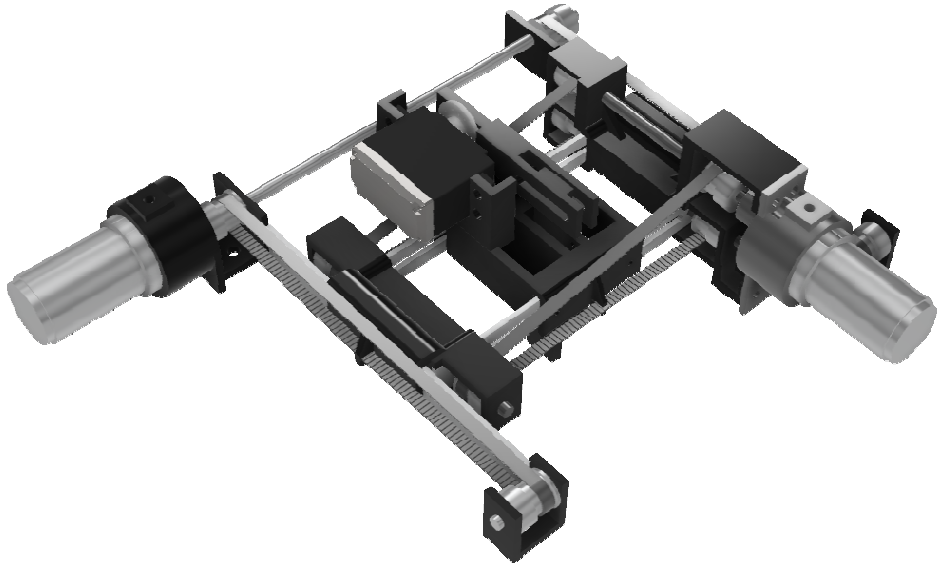


Figure 26: 3 axis bridge structure

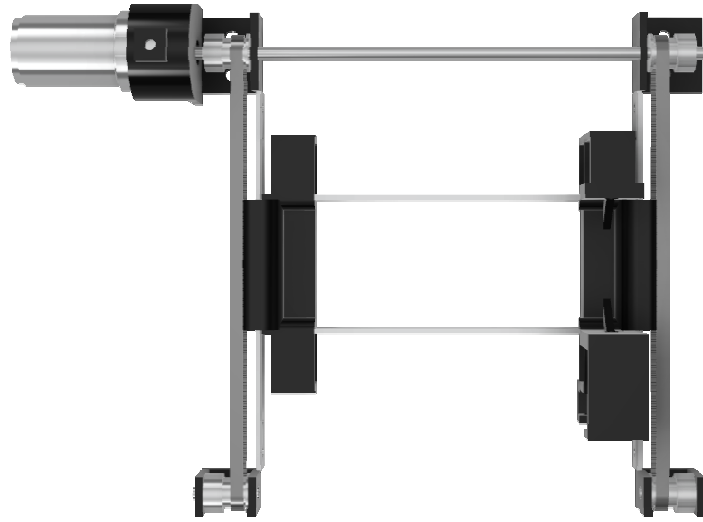


Figure 27: x axis structure of the 3 axis bridge

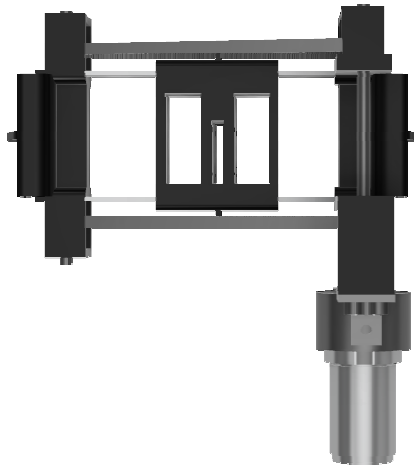


Figure 28: y axis structure of the 3 axis bridge

4.2 Electronic Circuit

Figure 29 represents the general overview of the electronic circuit. Two of the three servo motors represented change the direction of the direction wheels and the other picks up the pointer of the 3-axis-bridge. Servos of the model Tower Pro SG-5010 were used. These servos were used as they are precise enough for the purpose of changing the robot's direction and picking up the pointer. These servos are also very accessible in the economic perspective. They are powered with 5 volts through two of the three pins they have. The other pin is the pin for control of the servo position. These servos are controlled through a PWM signal, receiving a signal of 50Hz which corresponds to a period of 20ms. Typically, according to the datasheet [33] they receive a signal with a TON (time on) that varies between 600 μ s and 2400 μ s, where 600 μ s corresponds to one extreme and 2400 μ s to the other extreme of its range. These servos have a range of 180 degrees. Measures had to be made to determine the angle for each corresponding degree because the motors real specifications are not 100% equal to the datasheet. They have more accurate responses in the middle degrees of the range and less near the extremes.

The project used a Microchip's PIC32 micro-controller, with an output voltage of 3.3V. This values can be lower, being the minimal logical one voltage $V_{oh}=2.3V$. When configuring the digital pins as output the register ODCx register is set to 1, to set the port in open-drain mode, to allow the generation of outputs higher than 3.3V, though the use of a pull up resistor. The calculations for the pull up resistors for the different pins have to be lower than a certain value in order to provide the minimum current that activates the component's input logic to a high state. This value is calculated through the Equation 14.

$$R_{m\acute{a}x} = \frac{V_{CC} - V_{IH(min)}}{I_{IH(max)}}$$

Equation 14

For example, the enable pin on the L293 has a V_{IH} minimum of 2.3V and a I_{IH} maximum of 10 μ A. With a power supply of 5V, this means that the $R_{m\acute{a}x}$ is 270K Ω . There are also other factors to consider, such as parasite capacities, so the value should be smaller, being used a pull-up resistors of 10K Ω , that still requires a relative small value of current from the micro-controller port.

The DC motors are of the same type of the servo motors but without the feedback servo mechanic system which makes them work as DC motors. They are powered through power circuit of the type of H-bridge. Figure 30 represents the used H-bridge that was the L293 which is a quadruple half-H driver. For this project is interesting to use both directions of the motor and therefore each motor used two half-H driver making it a full-H driver H-bridge per motor. The H-bridge for the DC motors one and two are powered with 5 volts for the logic gates and 5 volts for the power supply for the DC motors. The speed of the motors is controlled through a PWM. As the motor's voltage signal changes at a high frequency and motors are an inductive load, two fast diodes were positioned at each side of the motors in reverse in order to discharge the loads and protect the power circuit. The frequency at which this motors operate in this project is 20 kHz, which is a frequency that is not too high nor too low for the circuit. It is not too high because it does not create meaningful parasite impedances in the circuit. It is a frequency supported by the H-bridge and with 20 kHz there is enough time between the Lows and Highs to discharge the motor load and charge again. To control the motor with an H-bridge in this configuration it needs two input signals. One with opposite the value as the other, for example if one pin receives a '1' the other has to receive a '0' to put the motor in work conditions. A PIC32 micro-controller (PIC32MX795F512L) was used in this dissertation. The PIC only uses one pin that sends the PWM signal, so in order to create two opposite signals at the same time, the signal passes through an inverter before one of the H-bridge input pins. The inverter used for this purpose was the HD74LS04P, which has an adequate commuting response to the signal.

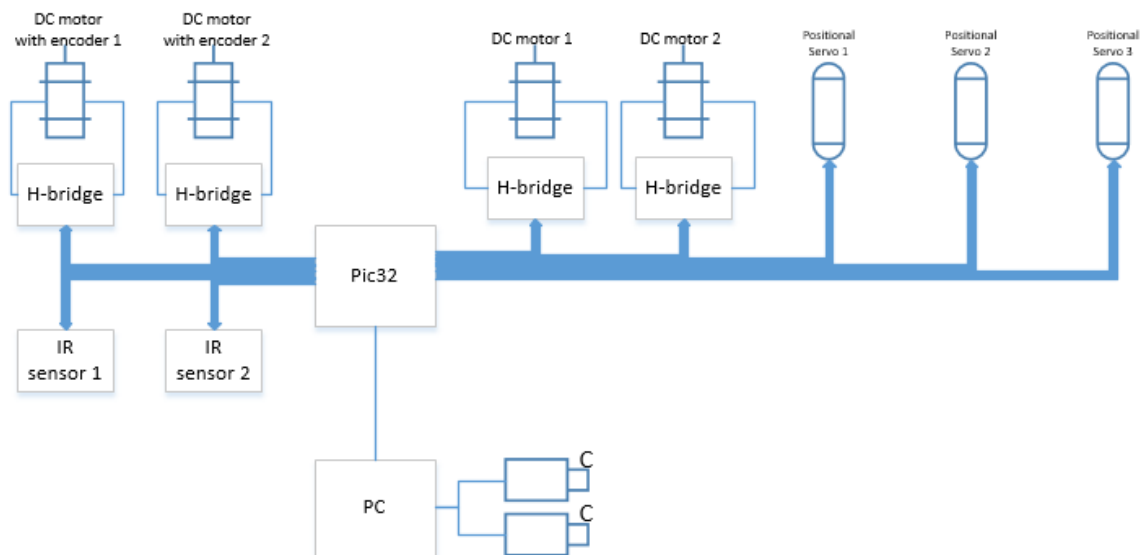


Figure 29: Electric diagram of the overall circuit

The DC motors with encoders have a similar configuration as the DC motors one and two in Figure 29. They serve the purpose of moving the two horizontal axis of the 3-axis-bridge (x and y). They also use one pin coming directly from the PIC and one from the inverter, each one. They are configured in a Full-H driver bridge configuration with fast diodes connected in reverse conduction, as represented in Figure 30. They use a PWM input signal with 20 KHz. These motors are of the model PD3237. The motors have Hall sensors connected to them (UTC KS1816) serving as encoders with a resolution of 14 counts per revolution with two channels. This means they can detect in which direction does the motor rotates based on the values of the two channels. The Hall sensors channels are configured as represented in Figure 31. They are powered with 5 volts on the Vcc pin and the pins Vout A and Vout B are connected to the PIC, which detects the logic value of the channels. The values switch between 0 and 5 volts and are directly connected to the PIC. The PIC's input pins chosen tolerate 5 volts.

The reason for choosing these motors is because they have a consumption of 4W with a big gear ratio reduction, giving the motors enough torque for the purpose of positioning the motors in the corresponding position.

The power to supply these motors comes from the H-bridge which is connect to a logic power supply of 5 volts and a power supply of 12 volts for the power electronics. One motor consumes a current about 500mA, the output current of the L293NE is 1A per channel. With 5 volts of logic power supply the maximum allowed power supply is 36 volts, which means the chosen H-bridge is adequate in relation to the power consumption.

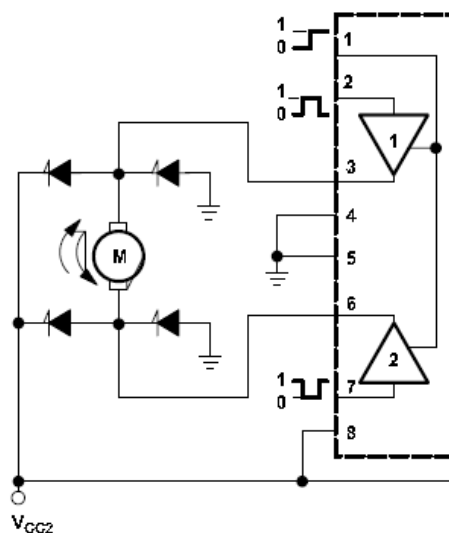


Figure 30: Full-H driver H-bridge configuration with fast diodes in reverse conduction

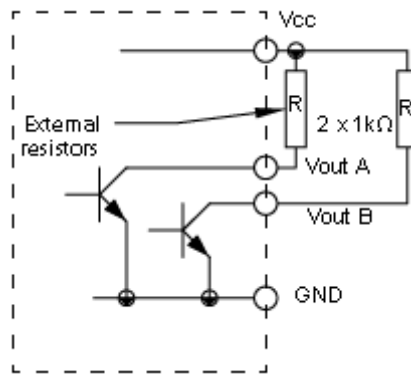


Figure 31: Encoder output circuit

In order to have a reference of the edge of the axis rail there are two infrared sensors, one per each of the horizontal axis. Figure 32 represents the top view scheme of the sensor TCST 2103, where there is a photo diode on the left side and a photo transistor on the right side. When the axis carrier passes between these components the transistor stops conducting current. By powering the '+' pins with Vcc of 5 volts and two resistances connected to ground on the 'E' and 'D' pins, it detects the changes of the sensor's state. Because the sensors are not totally equal, the resistance on the 'D' pin had to be adjusted differently for each one of the sensors, in order to generate a certain voltage above the PIC V_{IH} threshold. Resistances were chosen in order to produce a high state voltage around three volts. The 'D' is connected to a pin on the PIC configured as input in order to read the logic value.

In this circuit all the ground points are connected to the same ground and all the 5 volts power supply pins are connected to the same power supply. The same is for the 12 volt power supply pins.

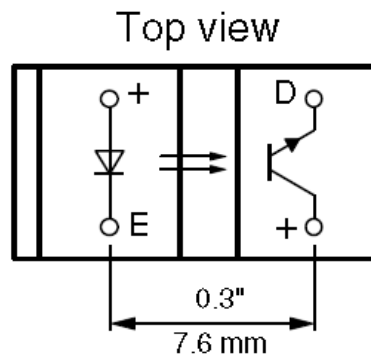


Figure 32: TCST 2103 top view scheme

4.3 Software

Relatively to the software there is a distinct difference between the levels of software. There is the low level coding, which is related to the direct raw input and output signals of the computing system. There is the high

level coding, which is related to making decisions based on the interpretation of processed data. Some parts of the software run on a PIC32 micro-controller and some others on a laptop. The software that runs on the micro-controller is responsible for the low-level functions and was written in the C++ programming language. The software running on the laptop is responsible for the vision system processing and the main high level program. It was written in MATLAB, which is very good for debugging and accessible to develop the algorithms.

4.3.1 Communication between the PC and micro-controller through UART

The communication between the PC and the micro-controller was made via UART serial connection, through an USB cable. There was a simple protocol designed for this communication. The commands are sent in bit frames characterized by a specific initialization and final character. The list of available operations is shown in Table 2. Following the initialization character ‘#’ comes an operation command such as pXXXXXXXX. To finalize the frame the program calculates a checksum based on the value of the least significant byte of the sum of the absolute value of the command XXXXXXXX. This checksum is followed by the character ‘!’ which finalizes the frame. For example if the programs decides to position the pointer on the coordinates (x, y) = (95, 51), the corresponding bit frame would be ‘#p010000506!’.

Command	Purpose	Values range and configuration
#	Initialization character	-
c1	Calibrate command	-
zX	Pointer state	X = [0, 1]; pointer up, down
pXXXXXXXX	3-axis-bridge position	XXXX---- = [0000... 0100]; x coordinates ----XXXX = [0000... 0100]; y coordinates
dXXX	Direction angle	XXX = [-90... 090] degrees
sXXXX	Moving speed	XXXX = [-100... 0100] %
X!	End character with checksum	X = [0... 9]

Table 2: Communication protocol commands

The software on the micro-controller has a circular buffer to manage the incoming commands from the pc. Figure 33 represents a buffer with the size of 20 present in the system. The program is designed to store the incoming bytes from the UART FIFO buffer as they arrive. The buffer has two pointers, one to point the initial position of the unprocessed/unread characters (i) and other to point the last character taken out from the UART FIFO (f). In Figure 33 there are two examples of the two different situations in which the buffer management can encounter. In both cases the number of unread characters is the same, but in the first case

the pointer 'f' is in front of the 'i'. When the pointer reaches the end of the buffer the next position is the first one of the buffer and then the calculation of the unread characters must be computed taking this into account.

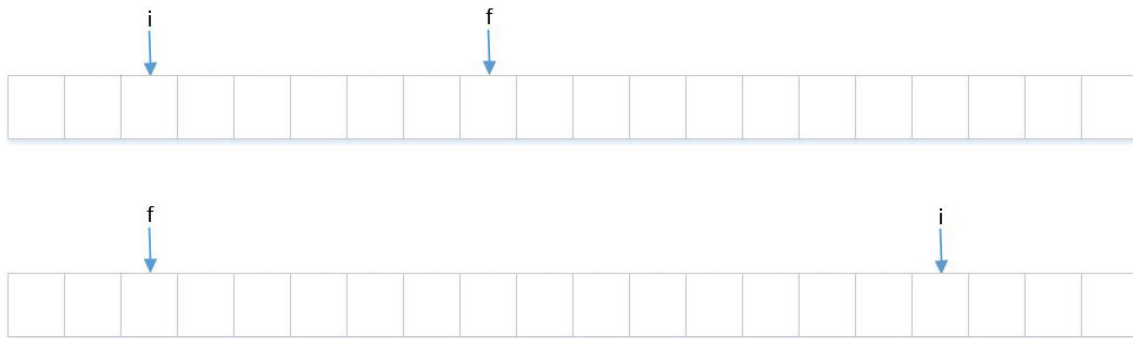


Figure 33: Buffer representation with the initial and final pointers

In Figure 34 there is a representation of the buffer management system in the form of an activity diagram. This system allows the buffer to store all the received bytes and check if there are actions available at the same time. For example, if the micro-controller receives an incomplete command at once, it can check for that case and wait for the buffer to get the rest of the command. In this diagram it starts by reading the content of the UART FIFO byte by byte, or character by character, to a string until it is empty. The next action is to store these bytes in the buffer. It is at this point where the pointer 'f' from Figure 33 is updated. The increment corresponds to the number of new bytes from the string and if the pointer reaches the end position it continues in the beginning of the buffer. When the string is empty, that is, when all new characters were stored, then it calculates the new value of unread characters. This value is important to decide if there are new characters or not and later too as it will be explained soon. If there are no new characters it returns to the reading of the UART FIFO action. The next step is to read the value located in the initial pointer and check if it corresponds to any of the commands in Table 2. If not it updates the initial position pointer by incrementing by one. In case it corresponds it checks if the value of the unread character corresponds to that command, this is important so that in case not, it means the command is not complete. When the software encounters this situation it will not change the 'i' pointer and will wait for the rest of the command to arrive, by checking the FIFO again. In case the number of characters of the command is valid the program will execute it according to the purpose of any of the commands listed in Table 2.

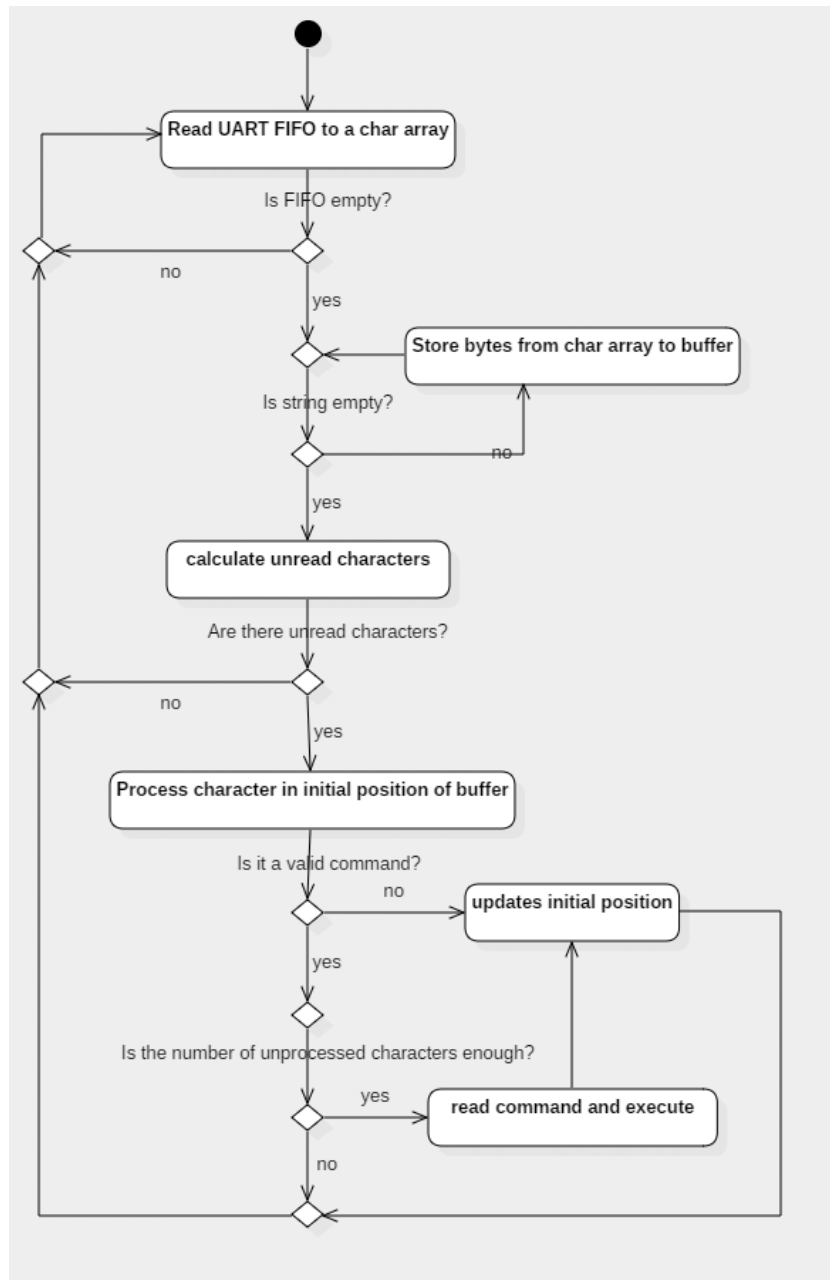


Figure 34: Activity diagram of the buffer management (UML)

4.3.2 Timers setup and PWM generation

The use of the timers of the PIC32 was managed according to the needs of the robot's functions, as listed in Table 3. The DC motors with encoders move the 3-axis-bridge horizontally. To control these two motors there was a PWM generator through the output compare register (OCx) one and two, assigned to timer 2. In

this model of the PIC32 the PR2 is a 16 bits register [26], which has a max value of 65535 ($2^{16} - 1$), as are also the OCx. The duty cycle value of the PWM signal is used to calculate the new variable OCx register value, which determines the motor's speed. In this PIC32 timers use the Peripheral Bus Clock as clock source (fPBCLK), which is set to half of the PIC32 Oscillator clock. The fPBCLK used was 40MHz. The module prescaler divides this frequency by a configurable three bits TCKPS of the register TxCON. The frequency at the output of the prescaler is known and so the frequency at the output of the timer is calculated according to Equation 15, where f_{in} is the frequency at the output of the prescaler, f_{out} is the desired frequency at the output of the timer and PRx is the value of the constant stored on the 16 bits register.

$$f_{out} = f_{in}/(PRx + 1)$$

Equation 15

The value of the PR2 was calculated according to the expression in Equation 16. Timer 2 PR register prescaler value was calculated in order to have enough resolution for the generation of the PWM signal with adjustable duty cycle. With a Prescaler ratio of four and a f_{out} of 20KHz, the resulting PR2 value is 499. A PR of 499 means that the Timer needs 499 pulses to reach the desired period and reset the timer.

$$PRx = fPBCLK/prescaler/f_{out} - 1$$

Equation 16

Timer 3 was assigned to control the three servo motors TowerPro SG-5010 used in this project. The frequency of 50Hz was chosen according to the recommendation of the datasheet. With a Pulse cycle of 20ms and a pulse width varying between 600 μ s and 2400 μ s the angle of the servo is controlled. Tests are described in the 'Results and Analysis' section that study the relation between the angle and the pulse width. The prescaler value is 16. According to the calculations on the Equation 16 the resulted PR3 value is 49999, which is less than its maximum value (65535). The three servo motors use the output compare registers 3, 4 and 5 to generate the PWM signals.

Timer 4 was assigned for the control of the x and y axis motors of the bridge. It has frequency of 1KHz. This is the rate at which the controller of the motors operate, through interruptions. With a prescaler of 4, the PR value is 9999. The functioning of the controller is explained in the sub-section 4.3.4.

There are only five OCx and were all used, so the PWM generation for the DC motors is made by software. Timer 5 was configured with a frequency of 20KHz and the same PR prescaler of four as the timer 2. The timer 5 was configured to generate interrupts and call a function that updated the PR5 register according to its state and the motor speed. In Figure 35 there is a representation of the PWM signal with the two possible states, "pin on" and "pin off". There is a function which converts the motor speed percentage command that comes from the PC into a valid period of time for the timer. At each interrupt the timer PR value is reconfigured according to the value of this variable and according to the state if it is time on or time off. Then the corresponding output pin is switched, the timer is reseted and the state variable is switched.

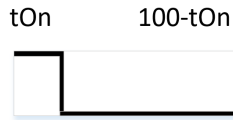


Figure 35: Illustration of the PWM states

Function	Timer	Period
DC motor with encoder of x axis of the bridge	Timer 2	50 μ s
DC motor with encoder of y axis of the bridge	Timer 2	50 μ s
Servo motor for z axis	Timer 3	20ms
Direction servo motor 1	Timer 3	20ms
Direction servo motor 2	Timer 3	20ms
3-axis-bridge horizontal axis controller	Timer 4	1ms
Moving DC motor 1	Timer 5	50 μ s
Moving DC motor 2	Timer 5	50 μ s

Table 3: Timers usage list

4.3.3 Encoder position reading

In order to read the encoders position, there are two interrupt vectors triggered by an external vector, one for each motor x and y. The interrupt is generated on a rising edge transition of one of the encoder's channel. When the interrupt occurs the function read the logic value of the other pin through other pin and increases or decreases the current position by one.

There is a function regarding the encoders position which is the calibrate function. This function blocks the flow of the main program in section 4.3.1. It is important that the robot does not do anything else while the calibrating function is running as there were not any interrupts used for this. In order for the robot to calibrate the motors and position them in the start position, the PIC32 reads the value of the pin connected to the infrared sensor. The robot axis carrier has a small metal plate in alignment with the sensor. When this plate interrupts the way between the infrared photodiode and the phototransistor, the signal changes, and then the position is reseted to zero and the motor stops. The sensors mark the origin of the coordinates system of the 3-axis-bridge system. Further tests are displayed and represented in the 'Results and Analysis' section, such as the measures the length of the track.

4.3.4 Axis motors controllers

The motors' controller is a PI (proportional integral controller). For this particular prototype, a proportional controller would be enough since the system is relatively light and the motor's axis don't get much momentum. However it is a better solution to implement the integral component if the goal is to achieve a stationary error of zero. Timer 4 generates interruptions in periods of 1ms and calls the interruption function. This function is the equivalent of one cycle of the controller calculations, taking in account the input signal value, the stored controller values from the previous cycle, the output value and the controller parameters. This function contains one controller for each of the two motors. The development of the expressions is explained in section 2.4.1 for the PI controller. The parameters that this controller has are the following. A time constant 'h' defined by the period of the timer. A 'Ti' constant which adjusts the gain of the integral component on the control signal. A 'K' constant which adjusts the gain of whole controller. And finally an anti-windup constant 'wd' that manages the anti-windup effect of the integral controller. The integral component has a tendency to create a windup effect which has all the disadvantages and problems explained in section 2.4.1. They were noticed and so the anti-windup mechanism had to be implemented. In this project the anti-windup solution works as a window of error in which the integral component is turned on. When the error is larger than the window value, that is, when the current position is still far enough from the target position, there is no need for the integral component to be on and so it is turned off. There is another group of functions regarding the control of the motors apart from the basic PI and anti-windup. When the robot receives a command to change the axis position, it initializes the variables, activates the timer, converts the control signal into a valid PWM duty-cycle value and determines when the motor reaches the end position. In this project this is a function that blocks the flow of the main program explained in the section 4.3.1. It has a counter that increases when the function is stabilized, and when it reaches a certain value the function ends. This is not a big problem has the robot should not also do anything else in the meantime, in case it receives another command it is stored in the UART buffer and checked out when this function ends. Various results about the motors and controllers are studied in the "Results and Analysis" section.

4.3.5 Pixel classification and training

The pixel classification implementation suffered various changes in the algorithms used. The goal is to classify potential plants pixels from non-plants pixels. By observation there is a significant difference noticed on the textures between plants and plastic fabric areas, and so the idea was to use Sobel filters to detect edges in plants, these are explained in section 2.11. Figure 36 displays the steps from the original image to the classified image. The first is an original image appropriately resized for the image processing. The second is the image in a grey scale. The third is the image with Sobel filters applied. As we can see there is a big difference in the texture between the plants and the plastic fabric areas. The fourth image passed through blur and highlight filters (high pass filter) to give contrast to high contrast areas. The last image are the pixels classified according to a certain threshold based on the previous image after another holes filler. The main problem with this approach was that it is very sensitive to shadows and they mess the classification up. The

part of the plant located in shadows is not classified properly and then the plant does not appear to have a round form. This results in a bad calculation of the center of the plant.

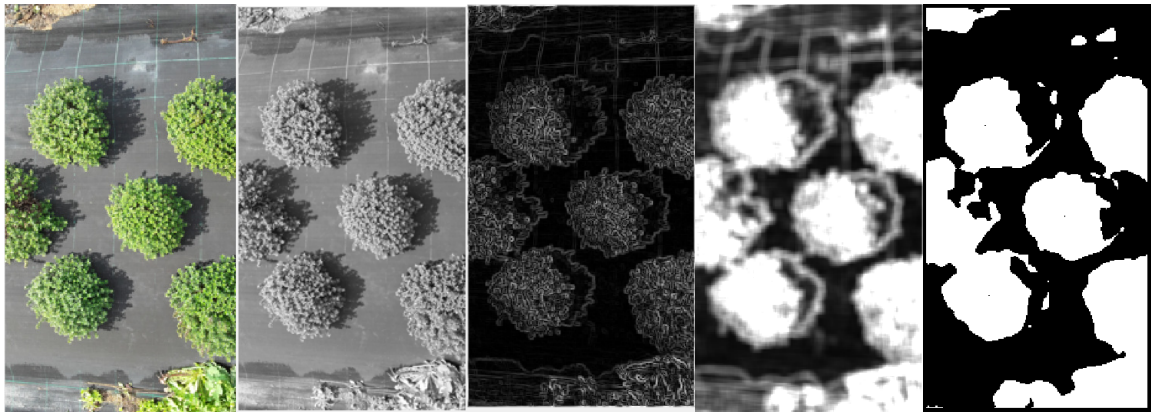


Figure 36: Pixel classification through Sobel filters

The next idea was to introduce two algorithms of machine learning and compare them with the edge detection approach to decide which path is more appropriated and worth to develop further. There are many possibilities of machine learning classifiers, and each one has its own advantages and disadvantages. For this project were chosen two different classifiers to test, the minimum-distance and the Bayes classifier. In Figure 37 there is a demonstration of the different classifiers on the same image. The fourth image in Figure 37 is the edge detection approach for comparison. To build a machine learning classifier the program has to be trained, this is, the image content has to be divided in different classes according to the different kind of objects that can appear on it. The program in the final version was trained with a section of the line with 32 plants. This originated 24700 pixels for class dirt and brown plants, 413000 pixels for the class of valid plants and 818000 pixels for the class plastic fabric. In real the algorithm does not need to be trained with so much training data but there was the concern that it had to have the best collection possible of data, and the training process is relatively quick. How the process follows is to take some sample images and select areas of pixels that correspond to each class and extract three dimensional features, such as mean and covariance of the RGB values of the image. The minimum-distance classifier pixels only uses the means of RGB of each class, and compares them directly with each RGB components of each pixel of the image. The Bayes classifier is more sophisticated, as it uses the mean and covariance to generate Gaussians for each class. The key component of this classifier is the calculation of likelihood values together with the prior probability in order to generate the conditional probability. In Figure 38 is a representation of the implemented Bayes classifier steps in the form of a data flowchart diagram. It starts by converting the RGB into a three dimensional array. The class pixels are the selected pixels in the training process. It uses them to calculate the means of R G and B from the RGB model, and it also calculate the covariance between these components which results in a 3x3 matrix. With these components it is possible to compute the likelihood for each pixel for one class. The prior probability uses the tested class pixels with other class's pixels. With the prior and the likelihood, is possible to calculate the posterior conditional probability for that class. In order to classify a certain pixel, the classifier compares

for each pixel the posterior probability for each class and chooses the one with the highest value. The Bayes approach is a better solution and is the one implemented on the vision system of the robot.

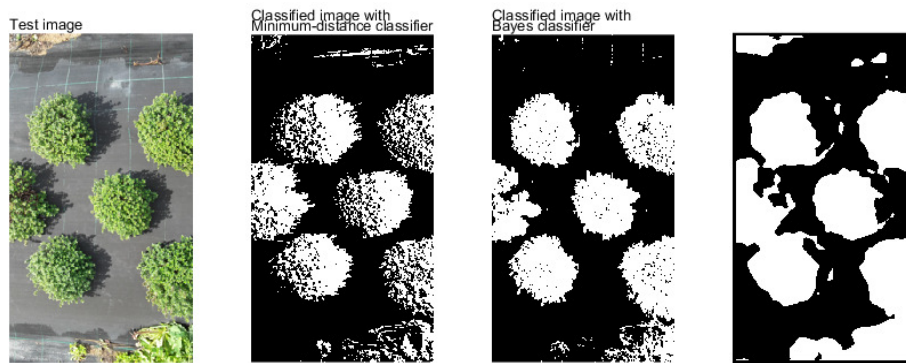


Figure 37: Comparison of the different classifiers

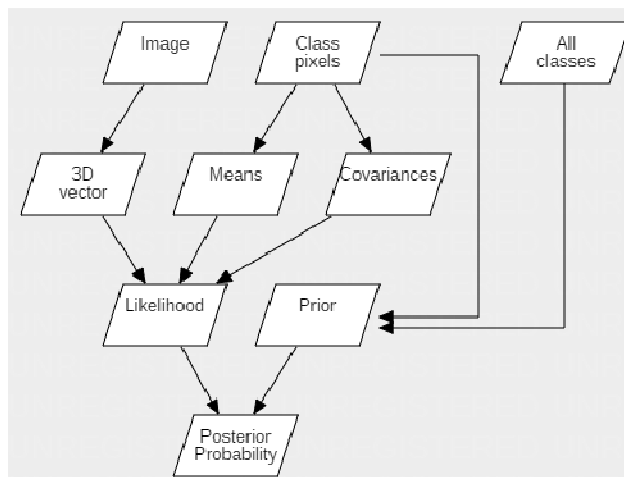


Figure 38: Bayes classifier data flowchart (UML)

4.3.6 Image recognition - progress development

The goal in this phase is to start from the classified image and interpret it in order to get the centers coordinates of the plants. The first problem encountered was to decide if the blobs on the image were actual plants or not. This created another problem, which is how to separate the blobs properly. At first the focus was not much on this second question, and a simple algorithm could easily separate the blobs which are not connected. The cases where the plants touched each other was later dealt with. The ideal plant has a round form, and that was the criteria used to test and decide.

The first approach to achieve this goal is to find the center of the plant represented in Figure 39. This method was not implemented in the final version due to a better solution found for the problem. This first solution uses a sample round blob image and overlaps it on top of the image in various positions. By going through the entire image with spacing intervals and counting the sum values of the overlapping rate, it is possible to generate an array to analyze. The program finds the peaks in this array and determines the position of the

center of the plant. In Figure 40 there is a demonstration of this array. On the left side is the overlap array with the corresponding relative rates values, and on the right side are all the secondary peaks found in the array. The first image of the Figure 39 is the source image to generate this array. The second contains the resulting points that are represented in the right plot from Figure 40, and the third is the original image classified with a small cross marking the calculated center. This center is calculated based on the peaks of the peaks array.

This version was a beginning but not a good approach. Its main limitation is that it only works when the plants have all the same radius and this radius is similar to the reference overlapping circle. It could be developed a process of using the algorithm for circles with different radius but programing this manually requires too much processing for the results obtained. The results for this algorithm were not much reliable for other different testing images.

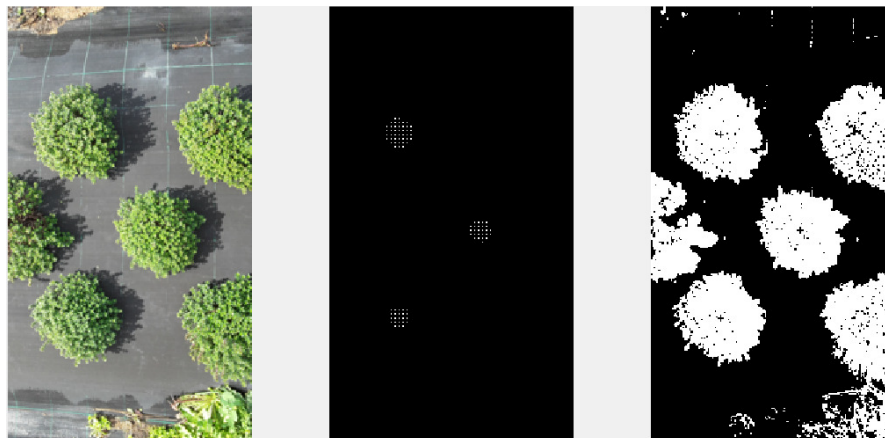


Figure 39: Centers detection

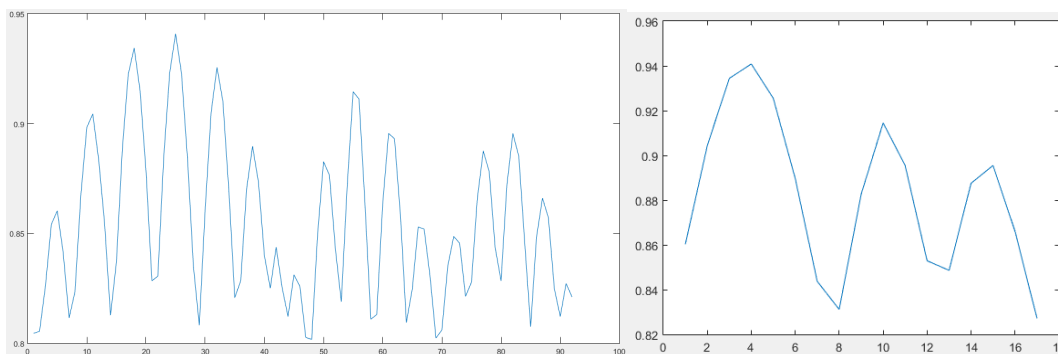


Figure 40: Peaks of overlapping rate

The next idea originated more reliable and versatile results. It was to validate the blob's shape through a different approach. This will be explained through a real example starting from the image in Figure 41, as classified testing image. The algorithm separates the blobs, and in Figure 42 is the analysis of one of the blobs. The image is separated in two arrays, one for each axis, these arrays being the sum of each column/line of plant pixels. The concrete interpretation of these arrays is the shape of the blub, looking from the sides of

the image. In order to find the most similar curve the goal is to determine the most likely radius of these lines. This is achieved by calculating the differences array between the line and a sinusoidal curve with a certain radius. In Figure 42 the program tried out all the possible radius from a constrained range and found out that the most likely radius is 34 for y and 35 x axis line. The blue line represents the differences array which is the absolute error value between the sinusoidal curve and the blobs curve/line for this axis. The minimum value of the sum of the absolute error makes up the absolute error in the graphic. The relative error is a normalization of the absolute error according to the plant's area with the calculated radius. The relative error is the one that determines if the blob is round enough. The center in these cases is the peak of the blob's curve line. This solution solved the problem in a better way, but did not worked by itself, as there was still the problem on how to separate the blobs if they are connected. How did the program knew if the blob that was being tested was one or more plants? This is its main limitation and therefore it needed the help of another algorithm to do the job. Another important limitation is that in case the blob appears to have an irregularity, the calculated center wouldn't match the real center. Further results discussing more possible cases for this shape classification system are in the "Results and Analysis" section in detail.

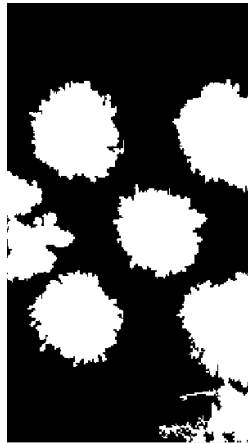


Figure 41: Starting image

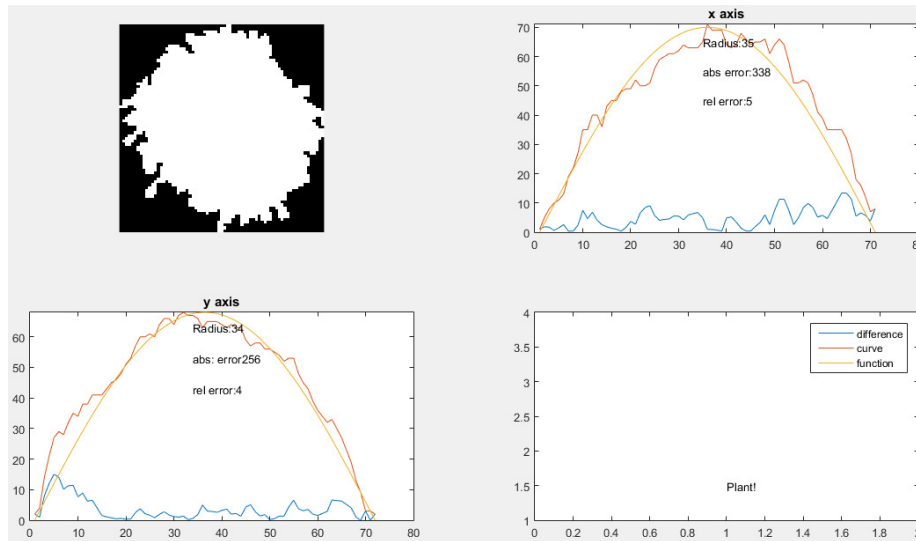


Figure 42: Analysis of a blob's shape in a valid plant case

The next approach was to use the Hough transform algorithm for circles, whose results are displayed in Figure 43. How the Hough transform algorithm works is explained in section 2.12. This algorithm seems like the right path to follow, but just by itself has many problems as it can recognize circles in places where there should not be any circle, and also happens that it detects circles inside others. This study will be discussed in the “Results and Analysis” section.

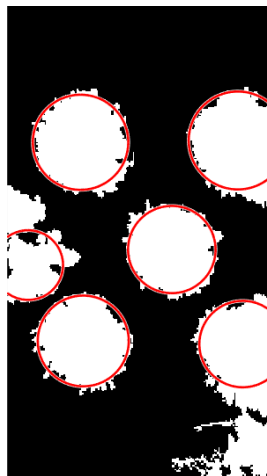


Figure 43: Circles recognition through Hough transform

The solution to overcome this limitation of the Hough transform algorithm is to combine it with the previous shape validation system. The shape validation was used as post validation of the recognized circles by the Hough transform algorithm. An example of this solution is illustrated in Figure 44. The blue circles are the blobs recognized as plants and the red ones are the ones validated as non-valid plants. Another criteria to validate the plants is if the center appears inside of the square marked in the image. It is important to make this distinction because even if the circle is assigned to a plant, the plant may not be complete and there are

cases where this leads to wrong results. Those will be covered in the “Results and Analysis” section. In the image, the x marks the center of the circles and the number below, the relative error. An important advantage of this solution is that with the Hough transform the problem that the shape validation algorithm had of not having a proper method of separating connected blobs is eliminated. The Hough transform algorithm returns more information, such as the center and radius of the circle, which allows to select an area around the center for validation.

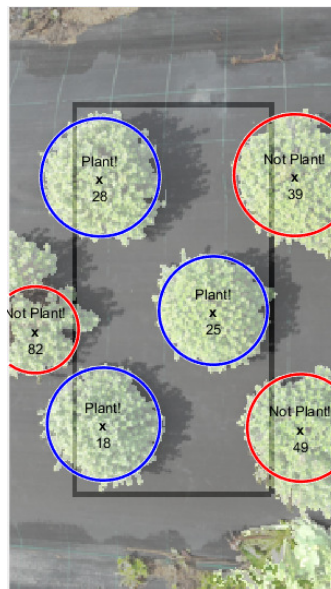


Figure 44: Combination of the Hough transform with the developed shape validation system

4.3.7 Image recognition – implemented

The implemented algorithm is described in the following paragraphs, starting by titles in bold text, which represent a certain function or step.

Find circles. The program uses the classified image and applies the Hough transform algorithm to find the circles with a specified range of radius. This function returns the centers and radius of the detected circles.

Separate circles. This function separates the individual detected circles into individual images. These images contain the blobs of the plants in image.

Clear blobs. A small step is introduced after obtaining the individual circles in images, which is to clear possible noise. This function treats two specific possible cases. Sometimes the circles detected by the Hough transform aren't always in the center, but even in these cases the circle information can still be useful to get to the real center position. In Figure 45 there is an example of this case. The red circle in the first image is the initial detection, which obtains the image in the middle image. The results after this process allow to get to shift the red circle to the blue circle and to prepare this individual image for the next step, such as the third image. The other useful case that this function solves is exemplified in Figure 46. Here the goal is to clear disconnected blobs from the main blob, which usually belong to other plants nearby. This is an

important feature of the preparation of the image to validate the form of the blob and the center of the plant. Although this has the limitation that it only excludes connected blob, connected blobs will result in a higher error which can make the plant validation more delicate. In Figure 47 is an example where this function has no effect on the detection circle and lets the noise pass through.



Figure 45: Blob centralization



Figure 46: Blob clearing, case 1



Figure 47: Blob clearing, case 2

Find radius and calculate the centers. After the blobs are separated in individual images is time to analyze the blob's shape. In order to determine a measurable value which allows to decide if the blob is acceptable, there are a set of properties that have to be extracted from the image. The measurable value is the absolute error between the plant's curve's shape with a sinusoidal curve function for both axis. The shape curve of the plant is obtained by summing the 2D image/matrix rows and columns, which make the y curve and x curve respectively. In Figure 48 there is a representation with the plots of the various comparison curves for different radius. The curves in the sinusoidal form are the functions in which the red line is compared with. They vary with the radius from 30 to 60 pixels. The red line is the blob's shape's curve. The irregular curves on the bottom correspond to the absolute value of the differences calculated between the red line and the

sinusoidal curve. In Figure 49 there is the radius likelihood function, which is calculated by the sum of each one of the differences lines. There is a minimum value that corresponds to the most likely radius of the blob's shape assuming that it is a circle.

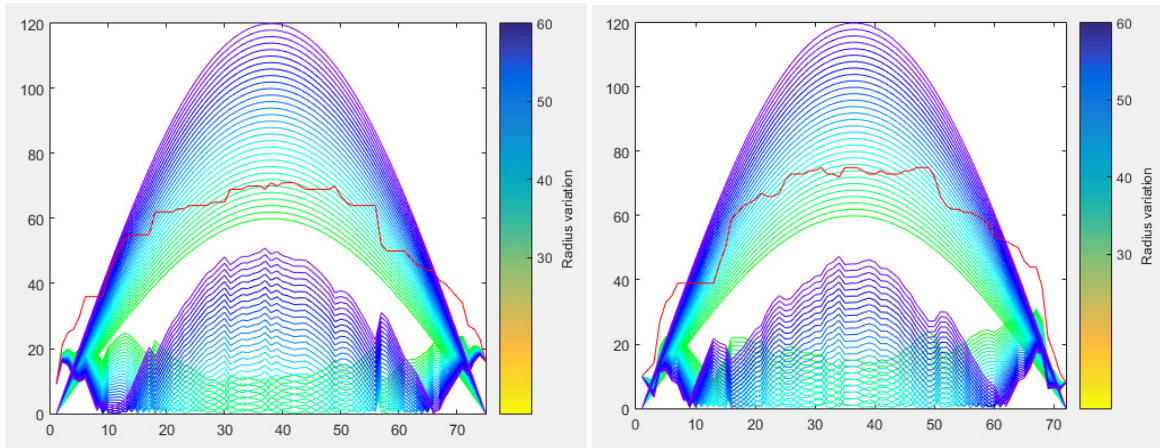


Figure 48: Absolute error function computing, x axis on the left and y axis on the right

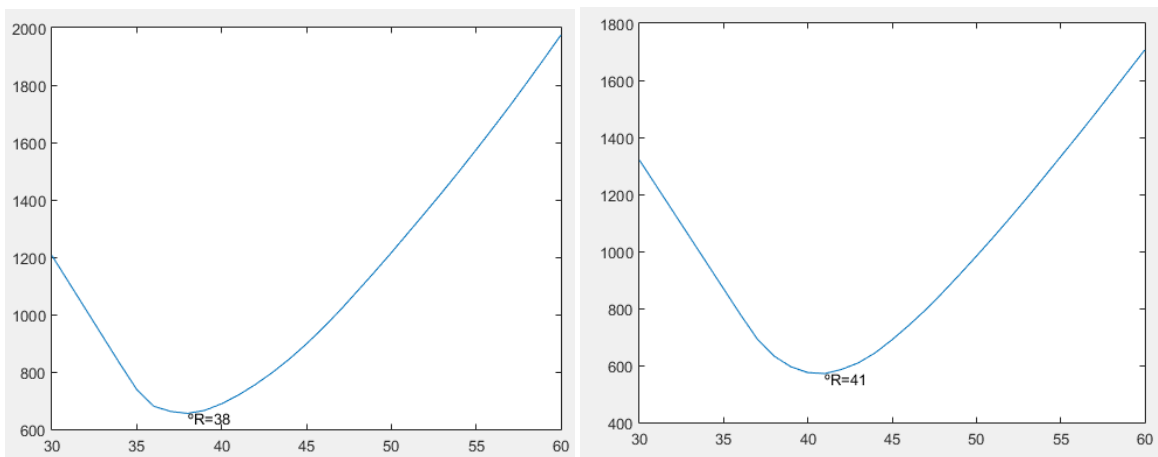


Figure 49: Radius likelihood function, x axis on the left and y axis on the right

Calculate errors. The assumption that the blob's shape is a circle in order to determine the most likely radius returns also the corresponding absolute error associated with that radius. In this step the program calculates the relative value which is a normalization of the absolute value according to the area of the blob. In Figure 50 there is the blob's representation correspondent to the example in Figure 48 and Figure 49. In this figure there is only the functions with the lowest error associated. The total relative error is the sum of the relative error of both axis.

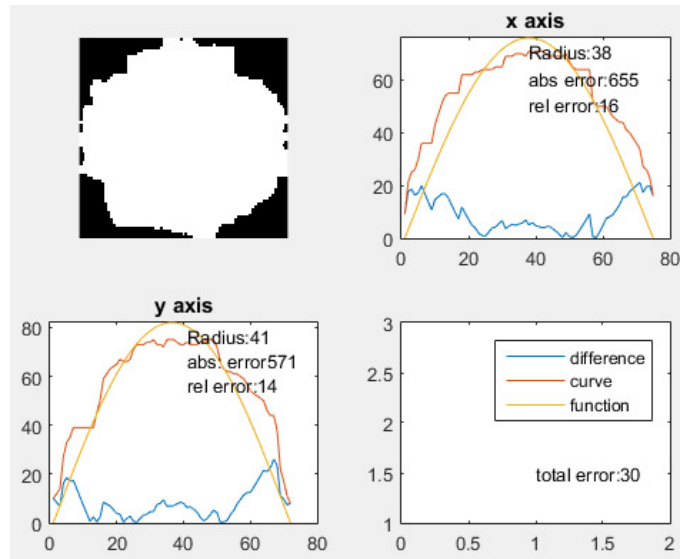


Figure 50: Analysis of a blob's shape

Decide the acceptable errors. The plants blobs with a corresponding level of total error are validated as plants. They are added to an array of plant's centers for further processing.

Detect overlaps. After the program obtains a list of all validated centers as blobs with the minimum requirements to be considered plants, often happen overlaps between circles. In the second image in Figure 51 there is an example of three circles that correspond to the same plant and overlap each other. The overlaps have to be above a certain threshold to be consider a conflict.

Exclude the uninteresting overlapped circles. All the centers that correspond to conflicts are compared with each other according to the relative value. In a conflict the circle that corresponds to the blob with the lowest relative value survives the exclusion process and the others are deleted.

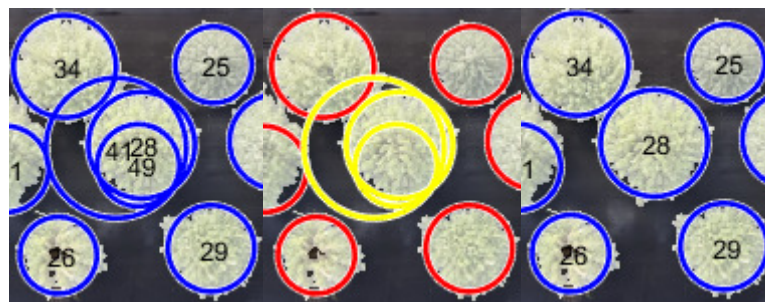


Figure 51: Circle conflict example, case 1

In this process sometimes there is another conflict which is when there are circles with the same relative error value, as Figure 52 represents. In this case happen, most of the time when neither of the circles is properly centered with the plant. The solution in these cases is to calculate the average center between the two or more circles in conflict. In the third image, in blue is the calculated circle.

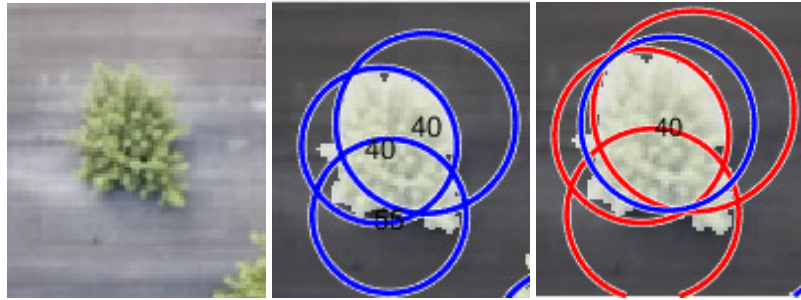


Figure 52: Circle conflict example, case 2

4.3.8 High level program (Top view)

This part of the software is the main decision unit which gives the necessary intelligence to the robot. The initial approach revealed some limitations, reason why it was modified at a second stage. In this first approach the camera that points downwards was positioned in front of the robot and not on the axis pointer. The plan was to take pictures, recognize the plants and add them to a map of plants. The program would have the ability to detect the motion between frames and update the plants positions on the map accordingly, in order to detect when they reached the bridge range behind the camera. In Figure 53 there is a scheme of the position of the camera in relation to the robot. As the robot advances from left to right, the camera takes pictures, recognizes the plants, stores the information into a map and tracks the motion of the robot. When the robot advances enough, that is, when the plants map is shifted enough, it marks the plants. Figure 54 and Figure 55 show an example of a simulation of the mapping algorithm in two different moments. The sample image was compiled from a video from a real field. The camera filmed along a line of plants, and from this video the appropriated frames were selected and then compiled into a long picture. The end result of this compilation originated a line with around 300 plants in total. In Figure 54 and Figure 55 image is a section of this line of plants, the black box represents the camera and the gray line represents the range threshold of the bridge. The blue squares with numbers represent the recognized centers on the map. The red crosses represent the marked plants. As the camera advances the coordinates of the centers are updated according to the robot motion. Note that the motion value on the corner of the figures represents the motion between image recognition, not the total motion.

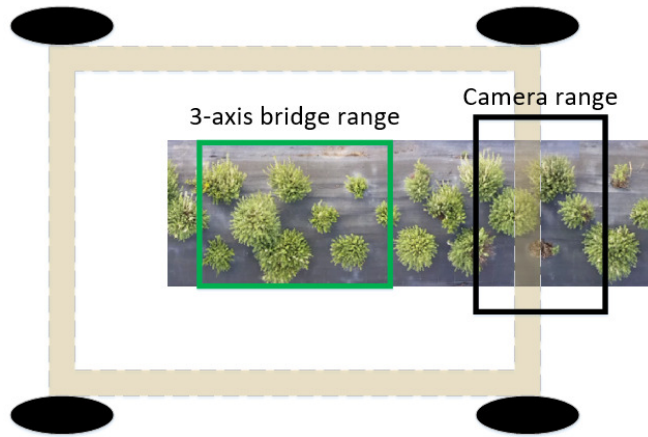


Figure 53: Scheme of the first solution for the camera position and mapping idea (top view)

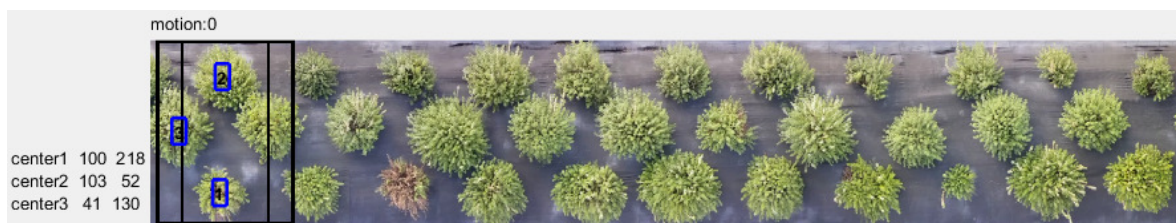


Figure 54: Simulation of the mapping algorithm with compiled loaded picture part 1



Figure 55: simulation of the mapping algorithm with compiled loaded picture part 2

As this simulation worked fine, the next step was a simulation with real pictures from the robot's camera on the printed plants, which was very similar to the previous simulation. As the robot advanced manually, the pictures were stored for the development of the image compilation, but this time the compilation would be done by the robot. In Figure 56 the system seems to be able to track the motion correctly, until the algorithm was tested while the robot's main program was running. The motion tracking system stopped working properly when the plan was executed in this environment. This is because when the robot was taking pictures periodically while running the code showed a tracking error that did not allow the tracking of the motion. The error was too big to be considered acceptable. The motion tracking algorithm detected motions of 1-4 mm when the robot was standing still and taking pictures periodically. The pixels of the images between each frame would have many brightness and color small changes, which confused the motion detector. These results are studied in the "Results and Analysis" section with more detail.

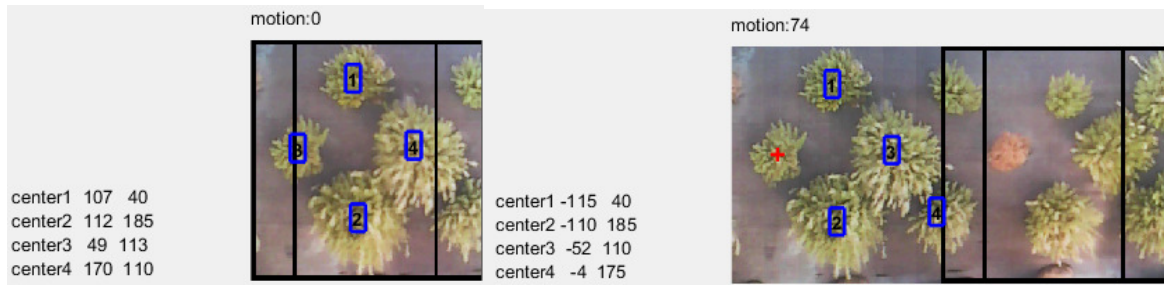


Figure 56: Simulation of the algorithm with loaded actual pictures from the robot on the printed plants

The solution for the problem was to replace the camera position. The new camera position is located on the 3-axis-bridge pointer. This approach was not initially considered as there are other limitations to consider, such as that the real life version recognition system would have the harvesting blade head in the sight of the image and it could create wrong recognition results. Another limitation was that the size of the camera is proportionally too large in comparison with the prototype pointer, and so the bridge was not designed to hold a camera. The pointer carrier had to suffer small changes in its structure in order to fit the camera and it ended up in a good solution. Figure 57 shows the scheme that represents this second solution camera positioning concept.

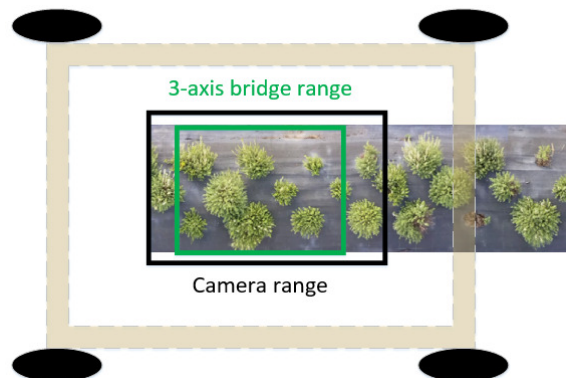


Figure 57: Scheme for the second solution for the camera position (top view)

In Figure 58 there is the basic activity diagram of this version of the main program. When the robot is turned on, it starts by making the connection between the laptop and the micro-controller. By initializing the connection the software on the PIC is restarted automatically with all variables assign with default values.

The next action is the calibration of the axis. It is important that the robot knows where the axis positions are and for that, it sends the calibration command to the controller. The concrete action taken is to send the calibration command to the PIC32 'c1'. The robot will rotate the axis motors in the negative direction until they reach the reference point at the beginning of the rail. This reference point is considered as the origin of the coordinates system.

Next, the program enters in the first main loop state. Here it moves the robot forward while taking pictures of the ground and when there are plants on range it gets out of the loop and stops the robot's movement. In order to move the robot the PC sends the command 's0060', which means run at a speed of 60%.

There is one position that the 3-axis-bridge range overlaps the camera range at best. In this stage the pointer carrying the camera is positioned on this spot for better detection. It sends the command 'p00500000' to the pic. The best position for recognition is when the camera is located in the center of the bridge and as the camera is right on the side of the pointer, the pointer has to be in position 0 on the y axis and 50 of the x axis.

After it stops ('s0000'), it is in position to harvest, so it takes another picture with the robot stabilized. The extracted centers from this picture are now filtered with the aim of having a list of plants that are in range of the bridge.

Then the program enters in the second main loop where it points all the plants in range. It starts by selecting the center with lower x coordinates from the centers array, that is, the plant in the back. Then it positions the pointer on top of the plant, lowers the pointer by sending the command 'z1'. It simulates a pause (harvesting time) and elevates the pointer again through the command 'z0'. The current center value of the array is also deleted. The out condition of this loop is when there are no centers left.

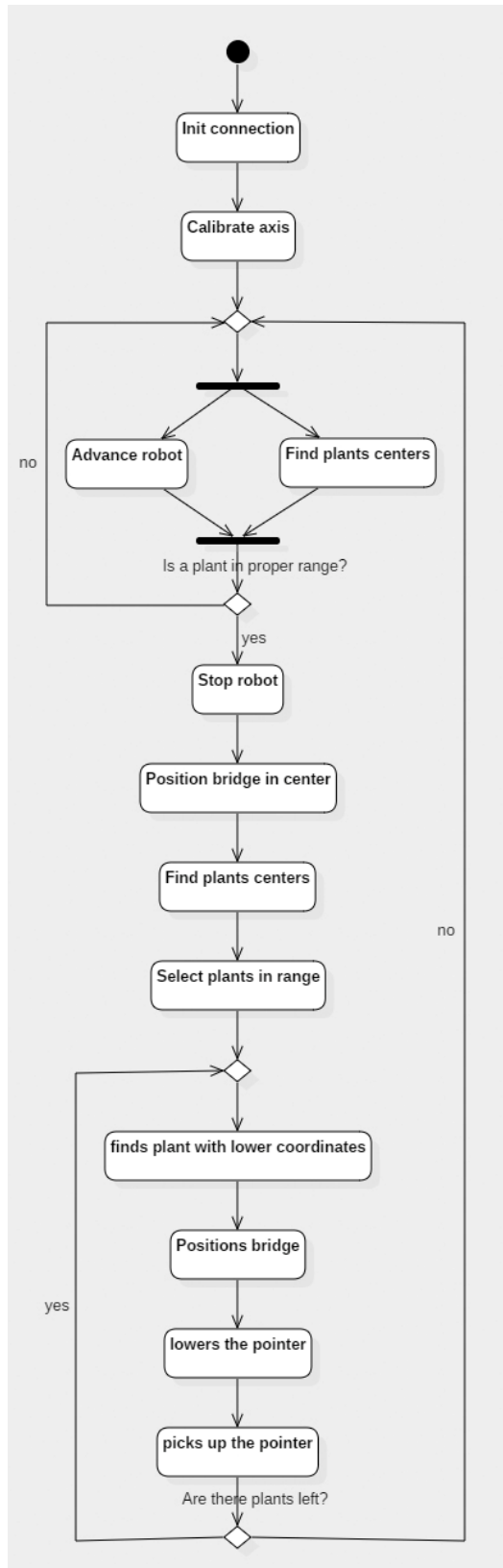


Figure 58: High level main program activity diagram (UML)

4.3.9 Alignment recognition

At the end of the project, the alignment recognition module was developed and tested. Due to time constraints, this module was not included in the main program but instead developed as an independent module. For a set of images taken from robot, they were processed and the alignments angles were extracted.

The camera positioned on the frontal part of the robot is meant to take pictures in order to extract the robot's alignment in relation to the line of plants. The camera is positioned at -49 degrees down from the horizontal plane. In order to explain the process, let's take a real example to visualize it better. In Figure 59 there is the starting classified image. The white pixels are pixels classified as potential plant's pixels and the black belong to the non-plant's class. The information in 2D, as in the image, is not in a format hard to process and has too much useless information. To extract the interesting information there is an array that is calculated through the analysis of each line of the image individually. In the regions where there are plants, usually there is at least one interval of the line that has a larger amount of consecutive plant's pixels. That region corresponds to the plant on that line with more impact. In order to extract this information it finds the peak with the largest width and adds the middle point of that region to the array. In Figure 60 there is an example of the line number 20 of this image with various widths. The combined points calculated this way generate the array demonstrated in Figure 61. This method discards the center of mass of different plants on the same line and generates some error regarding this case. But it is also a way to ignore noise on the side of the line and in between the plants. In average, with all the points combined, this error is very small. The straight line on the figure is calculated through the linear regression of all points. In Figure 62 there is the final result of this process. With some calculations it is possible to convert the inclination into degrees.

In Figure 62 there are also values of angles written, such as the recognized angle of the line in the image and the real angle of the image. Although, these angles don't correspond to the robot's alignments yet. They correspond only to the alignment of the line in picture. In reality the image is distorted because of perspective and needs an adjustment to find the real alignment of the robot in relation to the line of plants. After measurements and some analysis it is possible to determine this adjustment, those results correspond to the other two values in the Figure 62, described as 'corrected'. Further explanation and measurements are described in the "Results and Analysis" section.

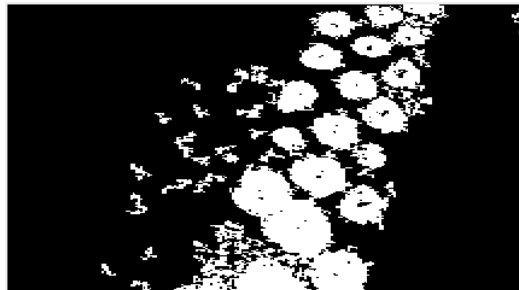


Figure 59: Classified image for alignment recognition

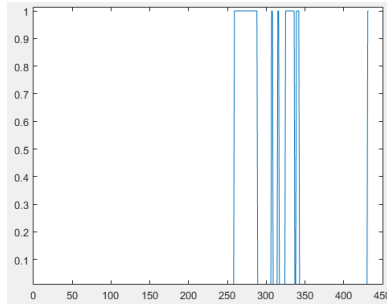


Figure 60: The plot of one line values of a classified image

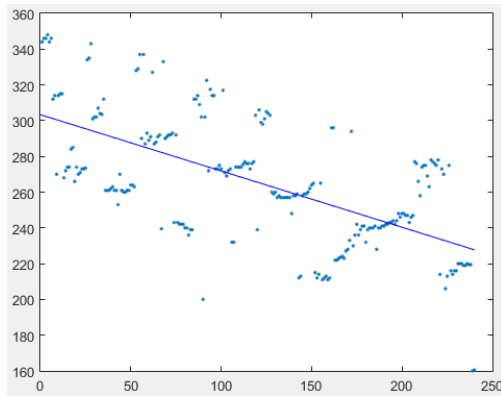


Figure 61: Demonstration of the array with the largest peaks

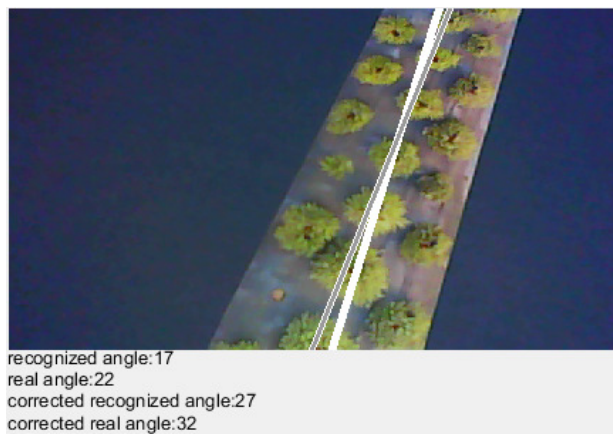


Figure 62: Original image with the recognized alignment drew with a white line and the real in grey

5. Results and Analysis

This section is dedicated to validate and explore the robot's specifications through the study of its functionalities. In order to take conclusions about the specifications of the robot, there were measurements and the analysis of the data documented. During the development phase there are some different kinds of results that were not included in "Implementation" section. It describes specifications such as precision, accuracy and proving that the function work.

5.1 Buffer explicit example results

The management of the buffer system is explained in section 4.3.1. Here there will be three examples with different possible cases that can occur in the program represented in Figure 34.

In Figure 49 there is a recording of the program's behavior for the first example. There were some delays and prints introduced on the program for the visualization of the demonstration. In this example the goal is to send the frame '#s0050z1d0603!'. The frame has 14 bytes. The UART buffer has only eight bytes and so the 14 bytes have to be split into two frames each time the program reads the content of the UART buffer, '#s0050z1' and 'd0602!'. The initial conditions of the buffer are when both pointers 'i' and 'f' are 0, which is the first position. As it is represented in Figure 49 it starts by reading the content of the UART buffer, which is '#s0050z1'. These bytes are added to the buffer and the 'f' pointer is updated to 8 which points to the position after the last byte. There is a variable that counts the number of characters that were not processed, called 'unprocessed'. It only processes one command at a time per loop. In this loop it finds the character '#', which marks the beginning of a frame and resets the variable sum, whose purpose is going to be explained soon. The pointer 'i' represents the current byte being processed and is updated, therefore the 'unprocessed' variable is set too. Next, the second part of the frame is introduced and is added to the buffer. There is a variable called 'sum' which counts the sum of each incoming byte in ASCII values. It is used later to calculate the checksum and check if it matches with the sent checksum. In this case the sum of all characters is 733 and therefore the checksum is 3 in this system. It matches with the sent value right before '!'.

```
COM5 - PuTTY
Read UART buffer:
Buffer string: #s0050z1
Buffer: #s0050z1,i: 0, f: 8, unprocessed: 8
->command: #
i: 1, buf[i]: s sum: 0, unprocessed: 7
Read UART buffer:
Buffer string: d0603!
Buffer: #s0050z1d0603!,i: 1, f: 14, unprocessed: 13
->command: s0050
i: 6, buf[i]: z sum: 312, unprocessed: 8
Read UART buffer:
Buffer string:
Buffer: #s0050z1d0603!,i: 6, f: 14, unprocessed: 8
->command: z1
i: 8, buf[i]: d sum: 483, unprocessed: 6
Read UART buffer:
Buffer string:
Buffer: #s0050z1d0603!,i: 8, f: 14, unprocessed: 6
->command: d060
i: 12, buf[i]: 3 sum: 733, unprocessed: 2
Read UART buffer:
Buffer string:
Buffer: #s0050z1d0603!,i: 12, f: 14, unprocessed: 2
i: 13, buf[i]: ! sum: 784, unprocessed: 1
sent checksum: 3, calc checksum: 3
checksum correct!
i: 14, buf[i]: sum: 733, unprocessed: 0
Read UART buffer:
Buffer string:
Buffer: #s0050z1d0603!,i: 14, f: 14, unprocessed: 0
Read UART buffer: █
```

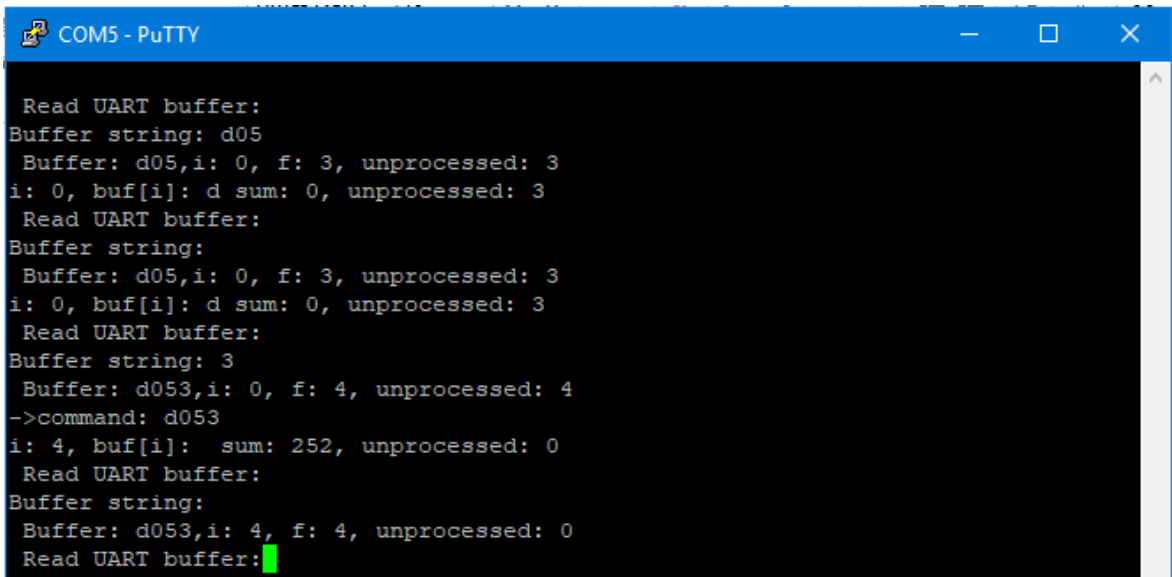
Figure 63: Buffer management example 1

In Figure 64 there is the second example which demonstrates the cases where there are numbers without any command assigning them. This example pretends to show the case where the pointers reaches the end of the buffer and also when the value of the calculated checksum does not match the sent one. The goal is to send the frame '012345678901#p002001206!'. This frame consists in a set of nonsense numbers as a starting offset to reach the end of the buffer and later a proper frame with the 'p' command for positioning the bridge. The frame is divided into '01234567', '8901#p00' and '2001206!'. After the third part of the frame being introduced, the number of the sum starts counting. During the reading and processing of the command. The pointers return to the beginning again, substituting the values stored previously there and reuse the buffer. Note that the calculation of the unprocessed characters and the process of the command is not affected despite they aren't position linearly in the buffer.

```
COM5 - PuTTY
Read UART buffer:
Buffer string: 01234567
Buffer: 01234567,i: 0, f: 8, unprocessed: 8
i: 1, buf[i]: 1 sum: 48, unprocessed: 7
i: 2, buf[i]: 2 sum: 97, unprocessed: 6
i: 3, buf[i]: 3 sum: 147, unprocessed: 5
i: 4, buf[i]: 4 sum: 198, unprocessed: 4
i: 5, buf[i]: 5 sum: 250, unprocessed: 3
i: 6, buf[i]: 6 sum: 303, unprocessed: 2
i: 7, buf[i]: 7 sum: 357, unprocessed: 1
i: 8, buf[i]: sum: 412, unprocessed: 0
Read UART buffer:
Buffer string: 8901#p00
Buffer: 012345678901#p00,i: 8, f: 16, unprocessed: 8
i: 9, buf[i]: 9 sum: 468, unprocessed: 7
i: 10, buf[i]: 0 sum: 525, unprocessed: 6
i: 11, buf[i]: 1 sum: 573, unprocessed: 5
i: 12, buf[i]: # sum: 622, unprocessed: 4
->command: #
i: 13, buf[i]: p sum: 0, unprocessed: 3
Read UART buffer:
Buffer string: 2001206!
Buffer: 206!45678901#p002001,i: 13, f: 4, unprocessed: 11
->command: p00200120
i: 2, buf[i]: 6 sum: 501, unprocessed: 2
Read UART buffer:
Buffer string:
Buffer: 206!45678901#p002001,i: 2, f: 4, unprocessed: 2
i: 3, buf[i]: ! sum: 555, unprocessed: 1
sent checksum: 6, calc checksum: 1
CHECKSUM ERROR!
i: 4, buf[i]: 4 sum: 501, unprocessed: 0
Read UART buffer:█
```

Figure 64: Buffer management example 2

Note that in a real scenario this frame splitting problem does not happen because the frames are short and consist only in one command in between the '#' and 'x!'. The longest frame sent is '#pXXXXXXXXx!'. This frame has 12 bytes but this is a serial communication, which means that the bytes come one after another by the proper order. The program is constantly reading the UART buffer and stores them in the secondary buffer, and does it in a faster rate than the rate that which the bytes arrive. In Figure 65 there is the case where only part of the command arrived on the PIC (d05) and is now waiting for the remaining (3). The command 'd' requires three bytes to complete the command. In this demonstration the frame is sent through the command line and had to be split.



```
COM5 - PuTTY
Read UART buffer:
Buffer string: d05
Buffer: d05,i: 0, f: 3, unprocessed: 3
i: 0, buf[i]: d sum: 0, unprocessed: 3
Read UART buffer:
Buffer string:
Buffer: d05,i: 0, f: 3, unprocessed: 3
i: 0, buf[i]: d sum: 0, unprocessed: 3
Read UART buffer:
Buffer string: 3
Buffer: d053,i: 0, f: 4, unprocessed: 4
->command: d053
i: 4, buf[i]: sum: 252, unprocessed: 0
Read UART buffer:
Buffer string:
Buffer: d053,i: 4, f: 4, unprocessed: 0
Read UART buffer:█
```

Figure 65: Buffer management example 3

5.2 Direction servo motors

Each one of the servo motors has a certain angle error. This generates some imprecision in the control of the direction in which the robot moves. In Figure 66 there is the setup for the measurements represented. The goal for these measurements is to specify the behavior for each one of the servo motors in order to generate the appropriate signal to control them. The photo shows the setup after the measures have been done. Note that the black cable under the wheel on the picture was not on the paper during the measurements. The robot was supported above ground so that the wheels are suspended. The torque of the motor is sufficiently high for its function on the robot, and so the fact that the motor has load applied or not does not change the results significantly. All supports were fixed together with duct tape to the table and robot, so that it does not move during the measurements. A paper was carefully positioned in a way that allowed to know the alignment of the paper with the robot. For 19 different values of pulse width tested, the resulted angle was marked on the paper by a dot where the pointer extension landed. Each one of the 19 pulse width values was tested for different movements with different rotation distances. Each measure has a minimum of five measurements with different rotation distances. Then the center of the axis was measured in order to connect all the measures to the center. After that, the lines were measured with a protractor with a resolution of one degree. This process was done for the two direction servo motors. In Figure 67 and Figure 70 there is a picture of the paper sheets with the angle measurements. In Figure 68 and Figure 71 there is the treatment of the data regarding the average relation between the robot's wheel's angles with the pulse width. In this graph there is also the standard deviation for each one of the measurements, represented as vertical bars with endings. In Figure 69 and Figure 72 there are the graphs for the standard deviation for each one of the measurements. These results show that the servos are linear but have a certain deviation from its average point. The mean of the standard deviations is 2.8 degrees for the left servo and 3.5 for the right one. Relating this to half of the

range of 180 degrees it results in a precision error of 3.1% for the left and 3.9% for the right. The main factor that affects the resulting angle values is the distance where the servo comes from, that is, the starting angle. When the servo changes are small is often the case that the angle does not change. In case they are large the servo tends to stop on the other side of the average angle for this pulse width. According to the datasheet[33] the servo is sensitive to pulse width from 600ms to 2400ms and the angle ranges from 0-180 degrees. This is not exactly accurate because the servo's angle range is larger than 180 and is sensitive to pulse widths outside its specified range. However as the results demonstrate, the standard deviation increases on the extremes of the servo's range, especially for lower pulse width values. Comparing the two servo motors, the motor on the left side has a better behavior than the one on the right side, as it appears to have smaller deviations in general. In Figure 69 and Figure 72 there is an odd result of deviation for the extremes values of the graph. In reality the deviation is also as large for this values. The angle measured is influenced by the previous position in which the servo was. It is not possible to test the extreme position angles by starting the motion from a position outside of the servo's range. Therefore the deviation of these measurements are smaller.

Regarding the direction servo motors, the robot is able to receive a command from the PC with a certain angle and position the direction wheels in the desired angle.

Its main limitation is the accuracy of the angle because of the deviation that depends on the starting point.

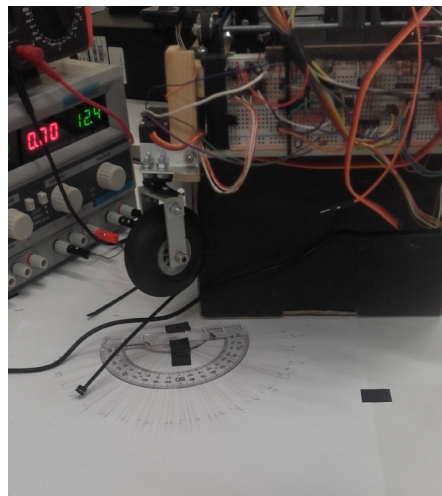


Figure 66: Setup of the measurements of the servo's angle

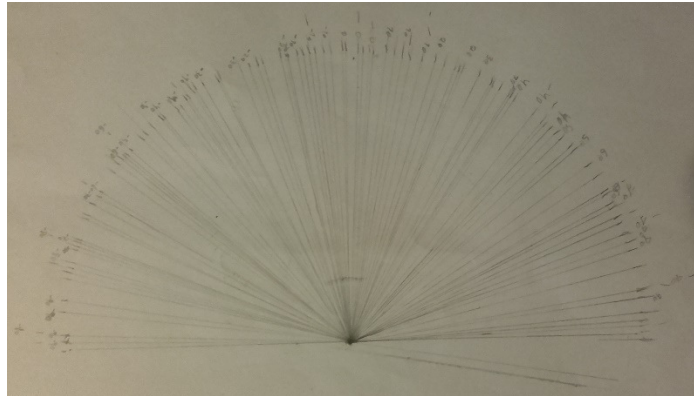


Figure 67: Resulted paper sheet with all the measurements for the left side servo motor

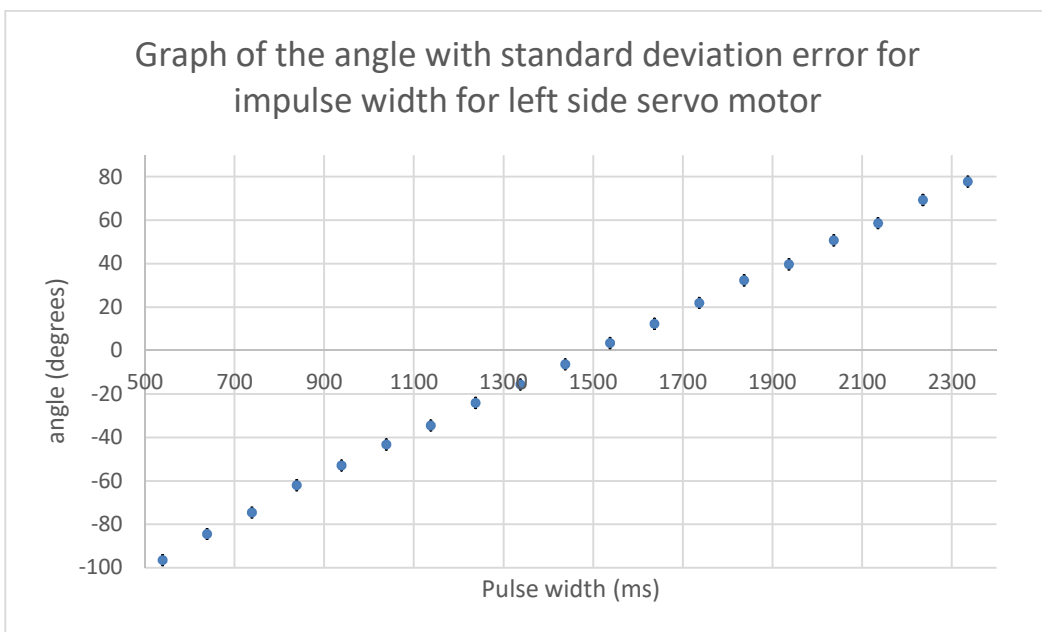


Figure 68: Graph of the angle with standard deviation error for impulse width for left side servo motor

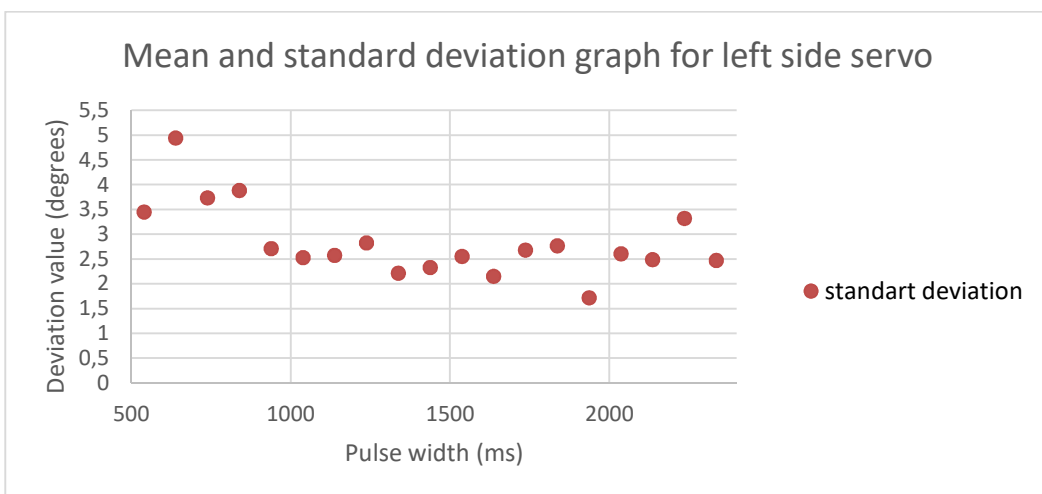


Figure 69: Standard deviation graph for left side servo

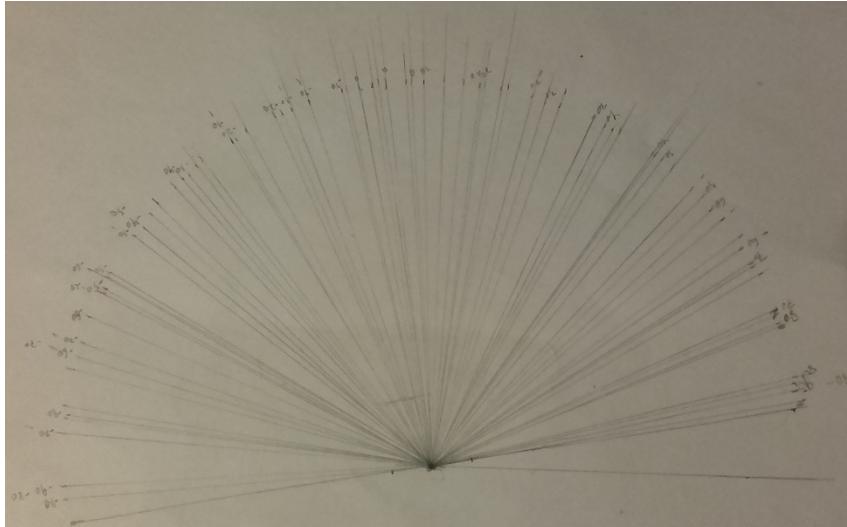


Figure 70: Resulted paper sheet with all the measurements for the right side servo motor

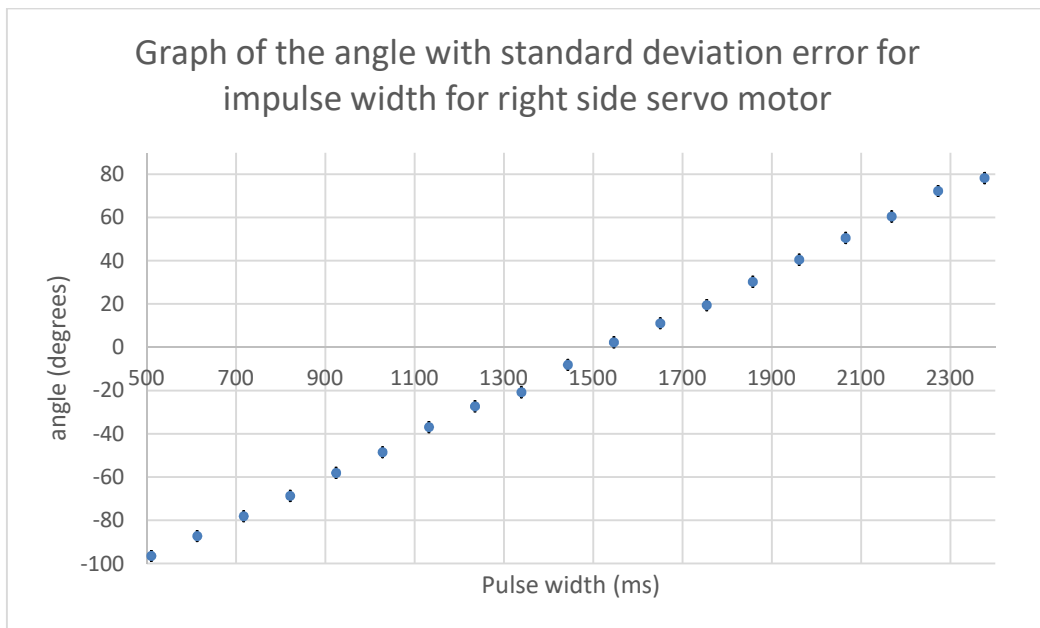


Figure 71: Graph of the angle with standard deviation error for impulse width for right side servo motor

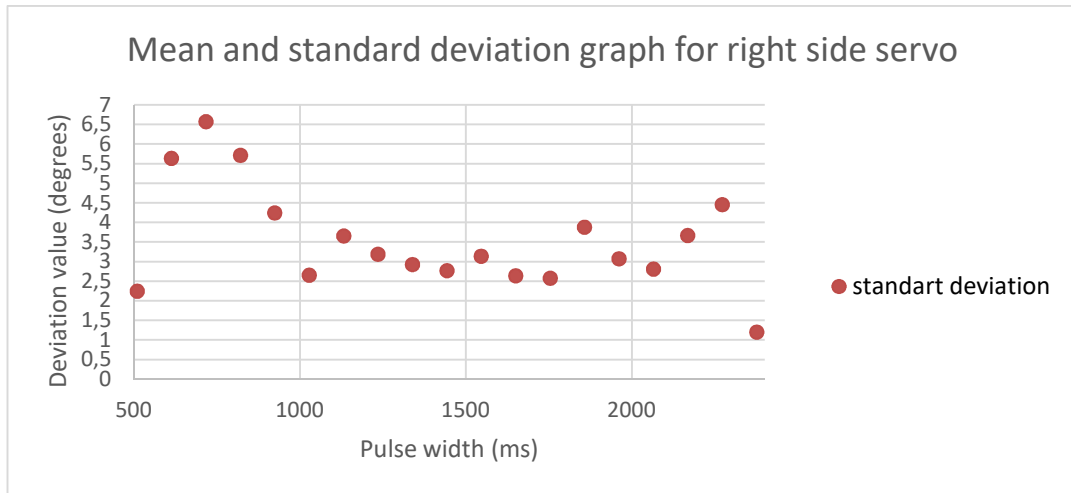


Figure 72: Standard deviation graph for right side servo

5.3 DC motor for the robot's traction movement

When it comes to implement the DC motors for traction of the robot, there are some measurements that had to be made in order to validate the implemented solution. The data was acquired by suspending the wheels in a stable position and count the time for 10 rotations. The time was equivalent to 10 rotations in order to minimize measurement errors, such as the reaction time for controlling the timer and motor speed deviation. This number is appropriate to the motors, relatively to their rotating speed. The measurements data acquired were the applied duty cycle on the motor, time to complete 10 rotations and average voltages in both of the motor's terminals. When the duty-cycle is at 50% the motor stops, as the average voltage is 0V. Therefore the measurements only start with a duty cycle of 75%, as it corresponds approximately to the point where the motor starts rotating. In Figure 73 and Figure 74 there are the graphs that compare the relation between the applied duty cycle with speed and applied voltage. According to the results, the relation between the duty cycle and the voltage seems to be linear. This matches with the expectations as the signal has a rectangular and the duty cycle represents the time that the signal is on high and low state. In the same graphs there is also the relation between duty-cycle and speed, which have a non-linear relationship. In Figure 75 there is the relation between voltage and speed, which is nonlinear. This matches the expectations as DC motors are highly non-linear, as there are many phenomenon involved. In Figure 75 the left motor presents a bigger nonlinearity.

Each motor has different specifications that relate the input and the output, but they had to be connected to the same generated signal. This is because there is only the Timer 5 available for the generation of this signal and the PWM had to be generated manually because all the Output Compare pins were already used. The motors specifications aren't much noticed on the results as the difference is not too big. For the purpose of these motors on this project, this is acceptable as this is not as important as for the other motors.

For this part of the project the robot is able to apply a PWM signal with a certain duty cycle on the motors and they make the robot move at a certain speed straight forward.

Its limitation is the small difference between each motor's speeds for the same signal which causes errors in a long track. However this limitation is compensated by the direction correction component.

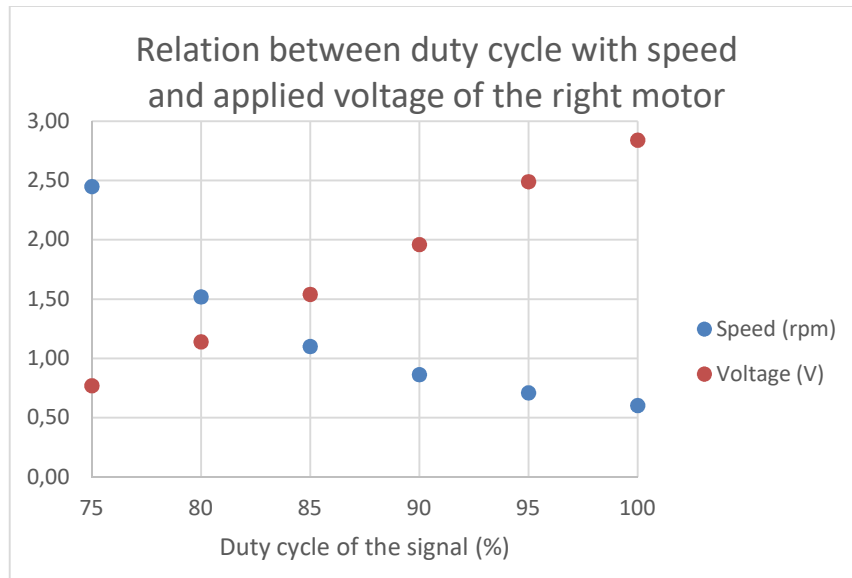


Figure 73: Graph relating the duty cycle of the signal and motor's speed and voltage for the right motor

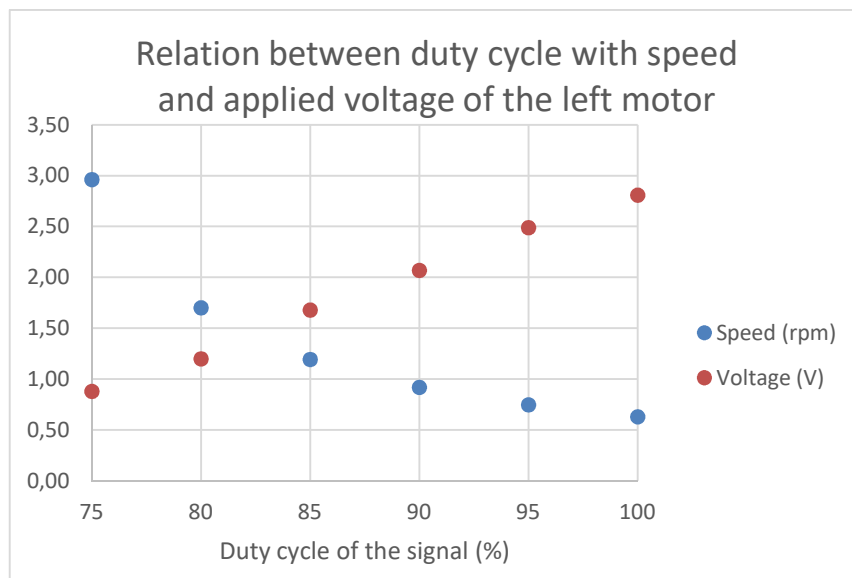


Figure 74: Graph relating the duty cycle of the signal and motor's speed and voltage for the left motor

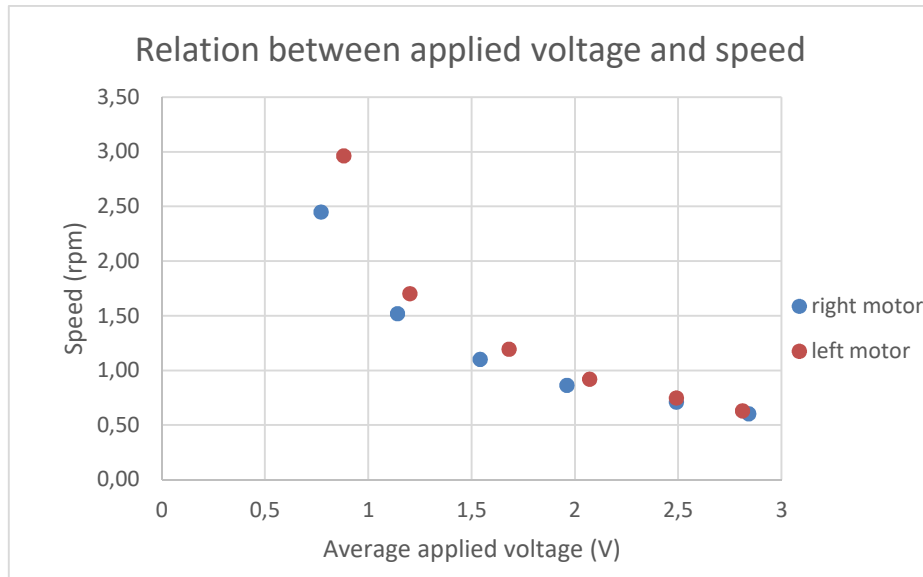


Figure 75: Graph comparing the speed of both motors in relation to the average applied voltage

5.4 Axis length according to encoder's pulses

At the part of the project that required to apply the controller and have a notion of the length of the track, there are some measurements to specify it. The measurements represent the number of pulses that the encoder generates while the motors rotate from one extreme to another. Each one of the axis has a mark positioned on a safe margin from the complete physical length. The mark for the x axis is located at 6.2cm from the carrier position when is on the reference infrared sensor. The mark for the y axis is at 4.6cm. There was a small program that rotated the axis to the direction of the sensor, then stopped and printed the counts of the encoder on the PC screen that were sent via UART, then return back for the next measurement. Before the each measurement the axis carrier was manually adjusted to match the mark position. There are 60 measurements with this process for each of the axis motor. The x axis counted in average 104.6 pulses, and the y axis 98.3. The need for that large amount of measurements is because the position was adjusted manually and therefore there is an associated random error.

Later it was tested that both axis supported around 110 pulses maximum and so the chosen axis bridge range was 100 pulses for both axis.

5.5 Controllers and axis motors

Right at the beginning of the development, to understand the motor's behavior regarding its reaction and speed to the applied voltages, there is a test that measures the rise time of the speed when applied a step signal of 12V. There are 20 measures for each motor without load. The resulted average rise time is 293ns for the motor x and 251ns for the motor y. These values mean that the reaction delay of the motor and the applied signal is insignificant.

The controllers were designed and implemented, after that they were tested. The controller's parameters were adjusted accordingly to the response and control signals. First the proportional parameter was adjusted to a value that caused the quickest response. The aim is to use the proportional component to cause an influence over medium and larger error values and use the integral component to adjust the end position in order to eliminate the error in the stationary position when the error is very small. In Figure 76 and Figure 77 there are examples of the motor's response for short, medium and long distance movements for proportional controller with gain of 3 and 6 respectively. The lines represented are the different control signals, 'yin' represents the target position, 'y' the current motor's position, 'iy' the integral component, 'ky' the proportional gain and 'uy' the control signal. In this case the integral component is turned off. The proportional controller with the gain of 3 has almost no influence when the error is small and almost manages the long distances well (>30 counts). The controller with a gain of 6 cause the motor too much speed in long distances overpassing the target position, and still has little effect on close distances. The control signal 'uy' saturates at 50. This value plus 50 mean the duty cycle of the PWM signal that defines the applied power on the motor. That means that the signal 'uy' ranges from -50 to 50 which corresponds from 0 to 100 of duty cycle.

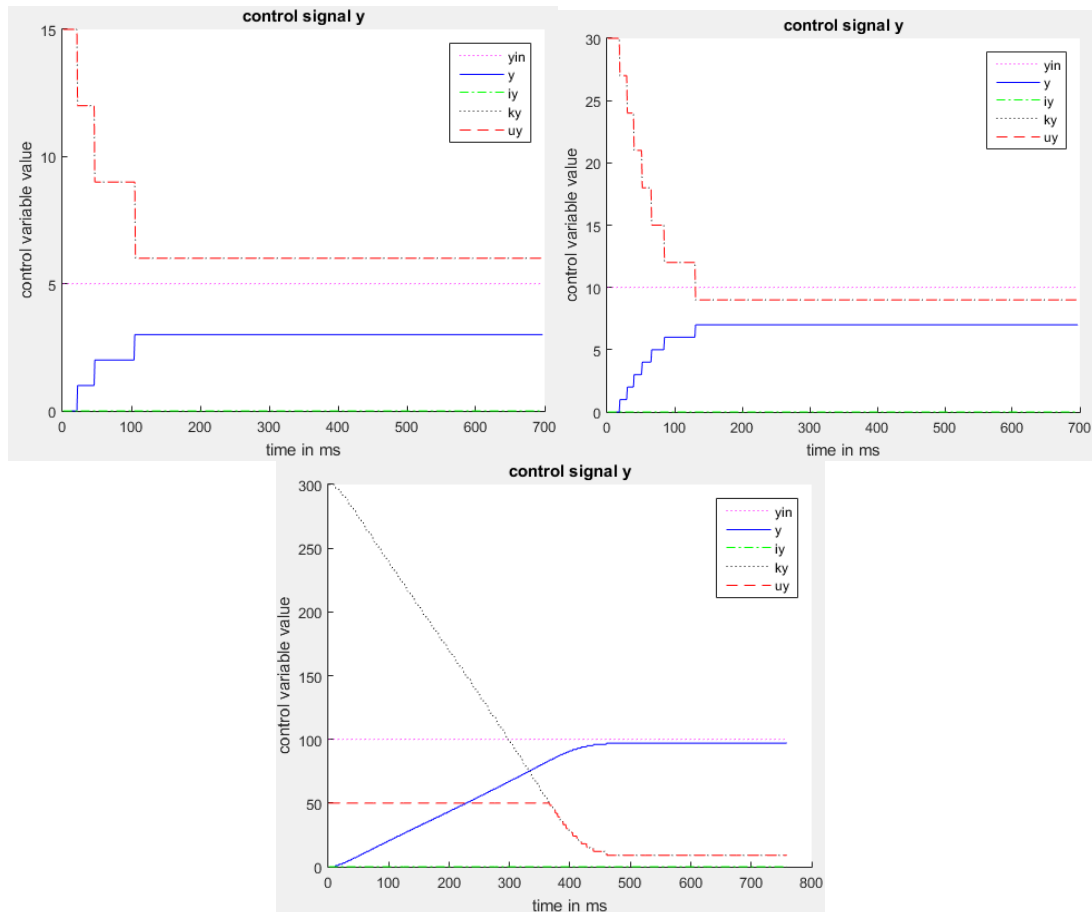


Figure 76: Motor's response to for a movement of 4, 10 and 100 counts, respectively, for proportional controller with gain of 3

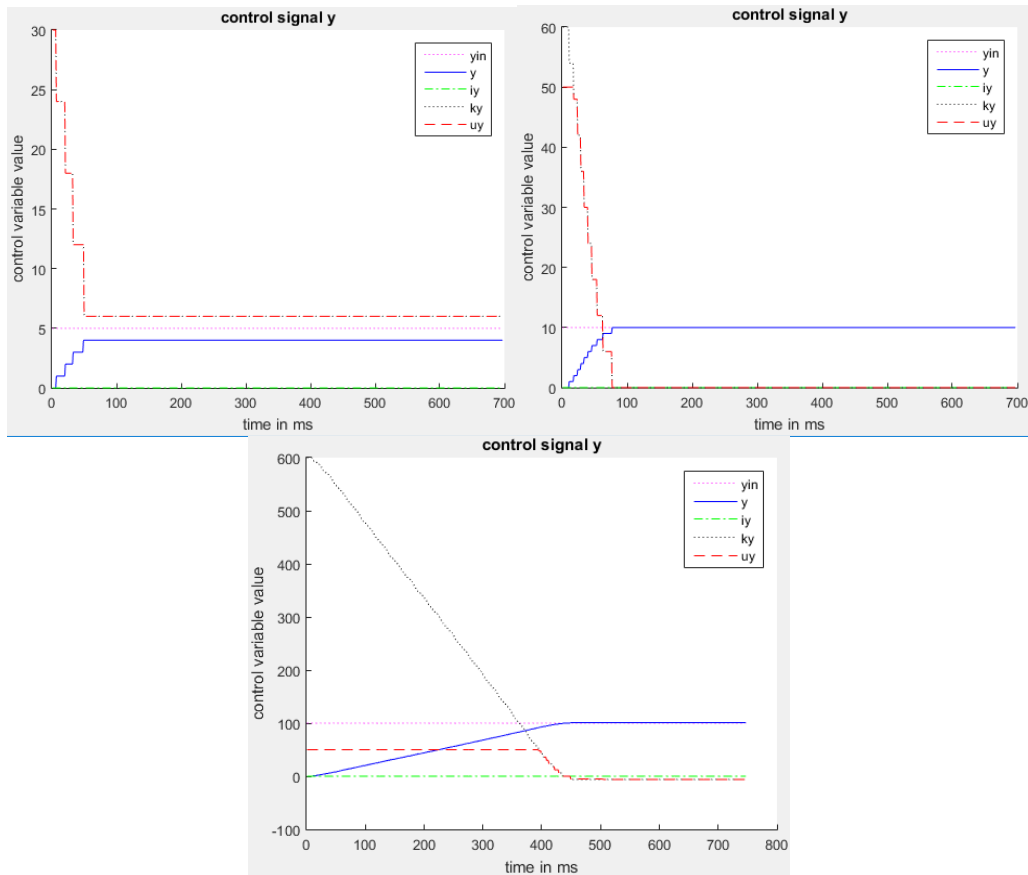


Figure 77: Motor's response to for a movement of 4, 10 and 100 counts, respectively, for proportional controller with gain of 6

During the tests there were some situations where the controller was unstable, such as Figure 78 represents. 'K' is the gain, 'ti' the integral constant, 'y' the target position, 'wd' the anti-windup value. As observed in previous results the gain of 3 could be appropriate. With 4 the integral influence is too high, which causes the system to be unstable. To turn it stable again either the gain 'K' has to be decreased or 'ti' increased.

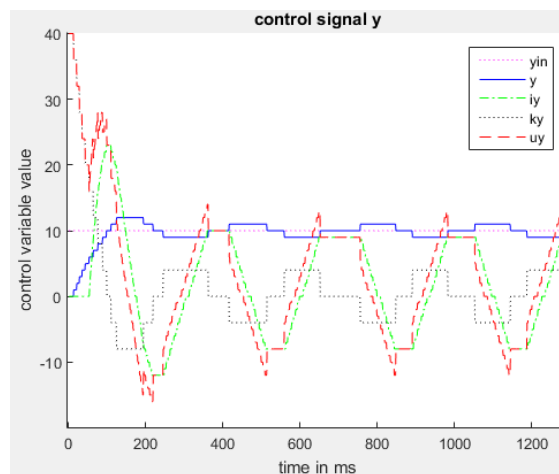


Figure 78: motor's response for the parameters $K=4$, $t_i=0.02$, $y=10$, $w_d=10$

The variable w_d is the window that limits the behavioral of the integral component, serving as the anti-windup system. When there is no anti-windup, the integral component over increases causing a large overshoot in the motor's response, as represented in Figure 79.

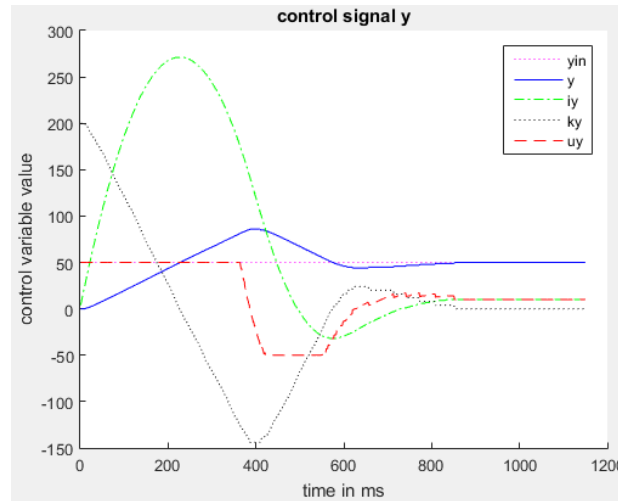


Figure 79: motor's response for the parameters $K=4$, $t_i=0.09$, $y=10$, no w_d

The parameters were adjusted to the final values of $K = 4$, $T_i = 0.09$ and $w_d = 5$. With this type of controller there was a compromise that had to be done between the response in longer and shorter distances. For many different combination of parameters, there is always one distance that has a better reaction than the other. The chosen parameters were considered to minimize the overall time that the motor takes to stabilize for all distances. In Figure 80 there are the results of the motor's response for very small, middle small and large distances. The parameter ' w_d ' was chosen according to the criteria that it should only have influence when the proportional controller had very little error, that is, for very short distances (<5 counts). At an error value of 5 the proportional factor is equal to 20, and the motor is already rotating at a slow speed. The minimum control signal required to make the motor move varies depending on the current speed and on the specific position of the track. This detail is analyzed next.

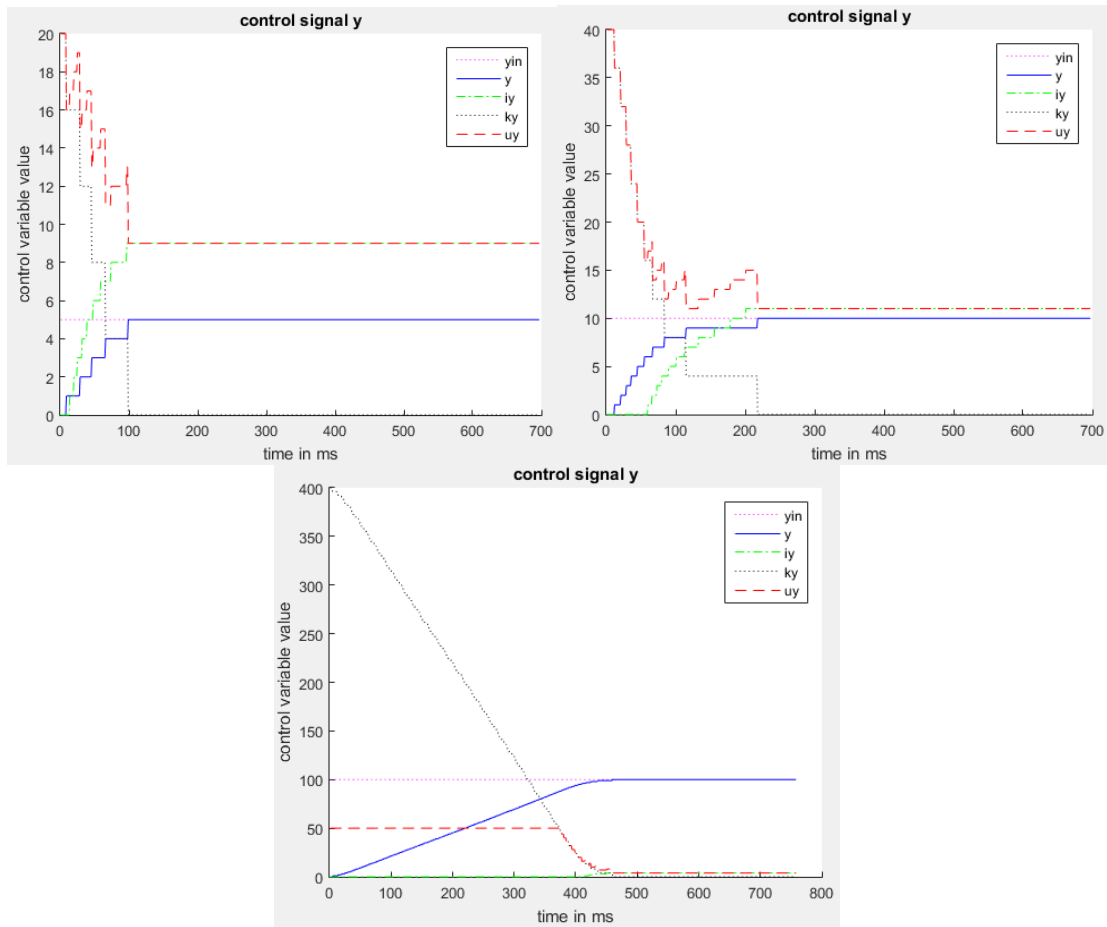


Figure 80: Motor's response for the parameters $K=4$, $T_i=0.09$, $w_d=5$

The time that the axis motors take to move different distances was measured. The measurements consisted in recording the motor's response curve along the axis position range in intervals of 5 counts. The graphs were analyzed and the information extracted was the corresponding time that took to achieve the zero state error and the one error state for each position. One error state points to the moment where the curve achieves one count of error. In Figure 81 and Figure 82 there are the results represented in the form of a graph. The set of data displayed in green dots represents the measurements when the motor behaves linearly. It serves as a reference of the motor's maximum speed in relation to the other sets of data. It appears that there is a significant performance difference between the time that the motor takes to achieve the zero state error from the one state error. This difference is displayed in Figure 83. In general the motor's response has an s curve form with an approximately linear region in the middle. In larger distances the control signal is saturated and the motor operates at maximum speed. In Figure 83 the case on the right side appears to have a larger delay to make the final adjustment and change from the one state error to the zero state error. As we can see in each case, the control signal needed to create the motion for the final adjustment is different. On the left case, the motor increases the final count when the control signal is 8 and on the right side it needs to wait for the integral component to increase the control value up to 15. This happens due to the irregularities of the system

that can cause different frictions in different positions. This explains the difference of some points in Figure 81 and Figure 82. Through the linearization of the green set of data it can be concluded that the motor x rotates at 0.223 counts per millisecond at a maximum speed (1/4.4843). The motor y rotates at 0.242 counts per millisecond at maximum speed (1/4.1321).

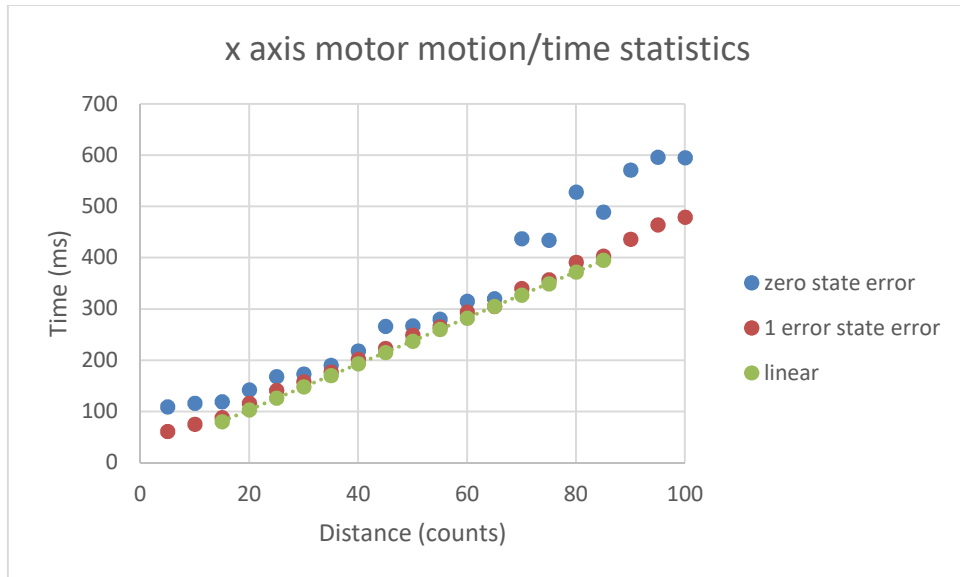


Figure 81: x axis motor motion/time statistics

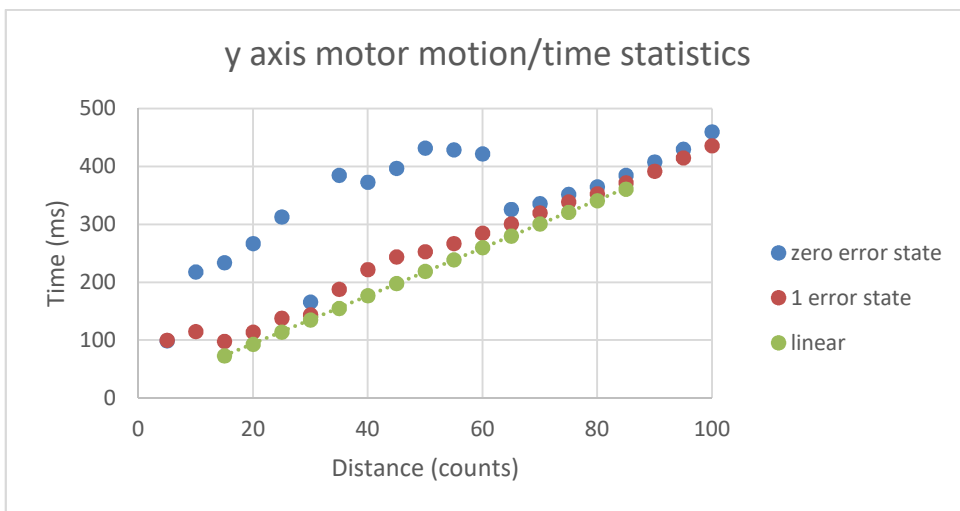


Figure 82: y axis motor motion/time statistics

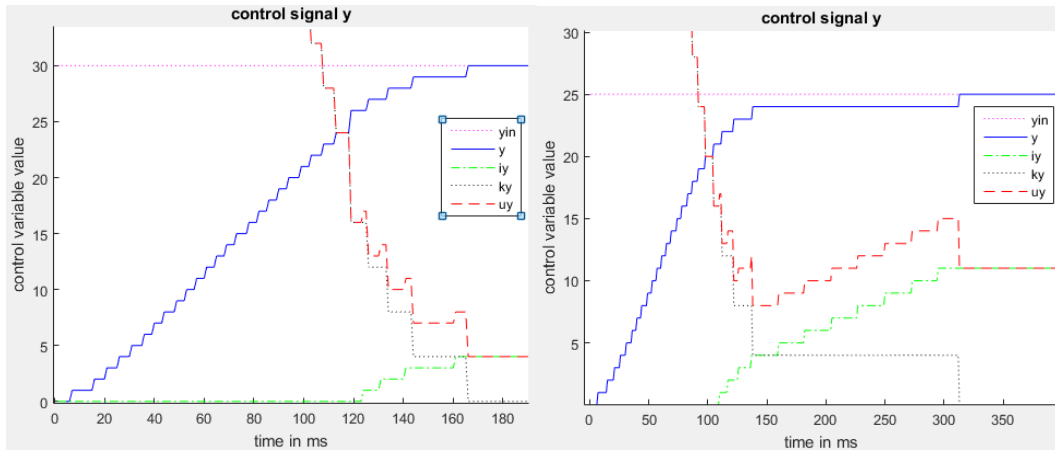


Figure 83: Motor response curve in two different cases

5.6 3-axis-bridge specification

The bridge is a composition of several components and each one of them has to be measured individually before making any conclusion about the system. There are two main reasons for these measurements. One is to determine the best conversion between pixel detected on an image and coordinates sent to the bridge. The other is to specify and validate the bridge behavior. We have a coordinates system in pixels, one in bridge coordinates and another in millimeters. To determine the relation between pixels-coordinates there is an individual study to specify the relation between pixel and millimeters and another to for the relation between millimeter and coordinates.

Figure 84 illustrates the setup used for this experiment. A millimeter sheet, with a resolution of one millimeter, was under the robot fixed on the table with duct tape. The robot's wheel were also blocked by pencils that were fixed by duct tape to the table. It is important to assure that the robot does not change its position during the measurements.

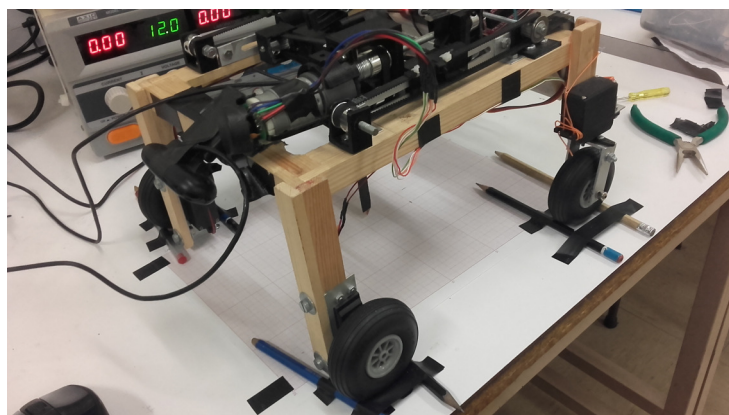


Figure 84: Setup for the 3-axis-bridge measurements

The measurements that specify the relation pixel-millimeter consisted in taking a picture with the robot's camera in the current position and measure the relation. In Figure 85 there is the view of the robot with the

used picture. The same picture was used to measure the two relations. In MAYLAB the pixels were measured in intervals of 10 mm of the picture of the sheet. It resulted in two 2D tables in which the means for each the pixels corresponding to the same x and y value in the millimeter scale resulted in the graph in Figure 86. The main source of error in this experience is the distortion of the camera depending on the area. Areas located to the extremes appear slightly smaller in the picture. The practical results are that the linearization is an approximation of the results but fails in predicting this factor.

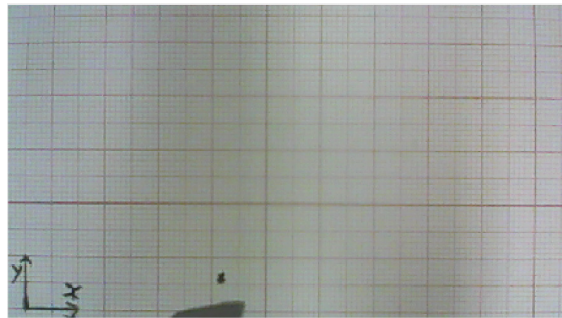


Figure 85: Picture to study the components relations pixel-millimeter and coordinate-millimeter

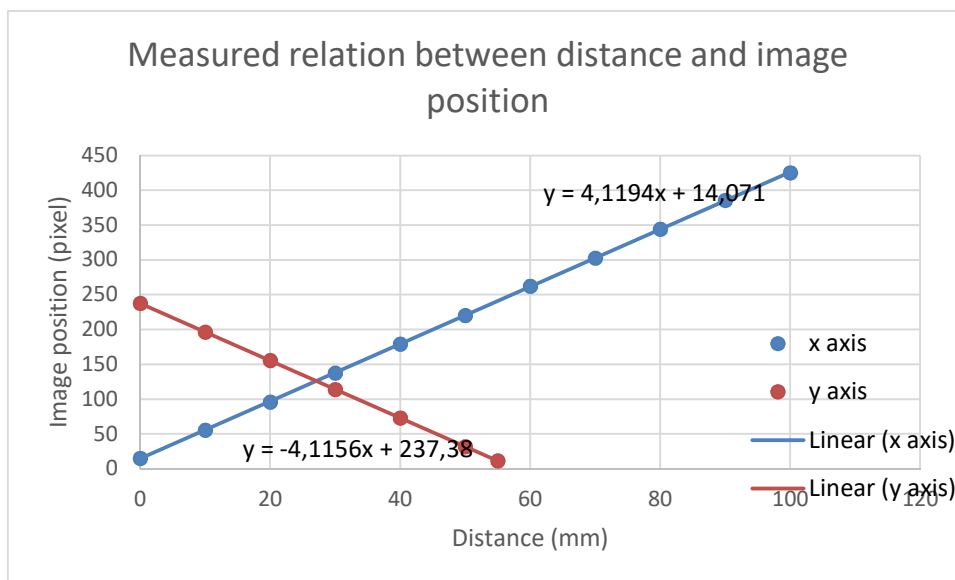


Figure 86: Measurements results for pixel-mm relation

Next is the specification of the relation coordinates-millimeters. Here is the main source of the error and so, the measurement experiment was more detailed and careful. The experience consisted in moving to each coordinates position in intervals of 10 starting the motion from different positions. The position where to the pointer comes may have impact on the results and therefore it is important to take every position into account. In Table 4 and Table 5 there are the raw results of the measurements. The first column represents the starting position and the first line the target position. The remaining contents of the table are the measured distances on the millimeter paper. In these tables, all possible combinations were used to guarantee the most reliable results. In Figure 87 there is a representation of the results in the form of a graph for both axis. The

standard deviation errors for each value are also represented there by small lines close to the points. These variations can be seen more clearly in Figure 88 and Figure 89. It is possible to conclude that different positions in the bridge cause different precision error. This may be due to the irregularities of the mechanical structure mainly. Right before the measurement there were several irregularities detected, some of them were corrected and other were slightly improved. Another conclusion is that the x axis provides a larger precision error source than the y, with a mean of 0.67 mm for y axis and 1.00 mm for x axis. The x axis is the one that moves in the direction forward/backwards and is mounted on the base structure. The y axis move laterally and is mounted on top of the x axis.

y	0	10	20	30	40	50	60	70	80	90	100
0		11	16	19	23	27	30	36	41	44	49
10	7		14	20	24	29	31	34	40	45	48
20	7	11		18	24	29	31	36	41	45	48
30	6	11	15		22	27	31	36	41	45	49
40	6	11	15	20		27	32	36	40	45	48
50	7	11	16	19	24		32	36	41	46	49
60	6	10	14	20	24	28		36	41	44	49
70	7	11	15	19	23	28	32		40	45	50
80	7	12	15	19	23	28	31	36		45	50
90	6	10	15	19	24	27	33	36	40		50
100	5	10	14	19	22	27	30	36	40	45	
mean	6,4	10,8	14,9	19,2	23,3	27,7	31,3	35,8	40,5	44,9	49
SD	0,663 325	0,6	0,7	0,6	0,781 025	0,781 025	0,9	0,6	0,5	0,538 516	0,774 597

Table 4: Raw measurements of the relation coord-mm for y axis

x	0	10	20	30	40	50	60	70	80	90	100
0		8	17	23	29	35	40	46	53	59	62
10	6		17	23	29	36	41	47	53	58	62
20	5	12		23	28	36	42	47	53	58	63
30	7	13	18		29	34	40	46	53	58	63
40	6	12	18	24		34	40	46	52	58	63
50	7	12	19	25	29		40	46	54	58	63
60	5	14	20	26	29	35		45	51	58	64
70	6	13	18	25	31	37	41		51	57	62
80	7	14	20	26	32	37	41	46		57	63
90	6	12	18	25	32	36	43	48	52		63
100	5	11	18	24	31	37	42	48	53	58	
mean	6	12,1	18,3	24,4	29,9	35,7	41	46,5	52,5	57,9	62,8
SD	0,774 597	1,640 122	1,004 988	1,113 553	1,374 773	1,1	1	0,921 954	0,921 954	0,538 516	0,6

Table 5: Raw measurements of the relation coord-mm for x axis

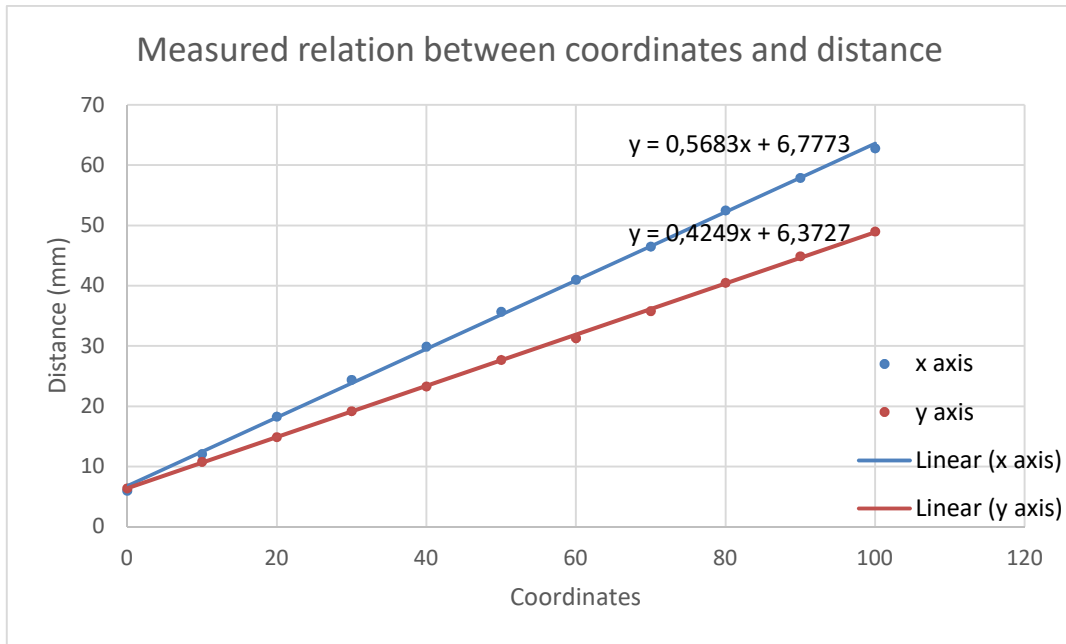


Figure 87: Measured coord-mm relation

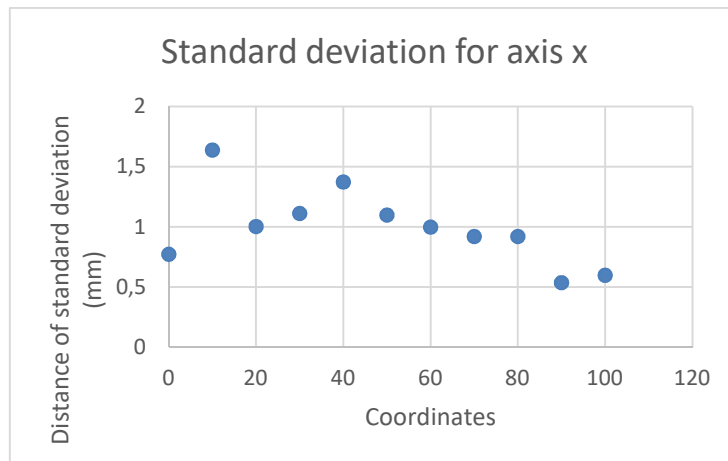


Figure 88: Standard deviation for axis x measurements

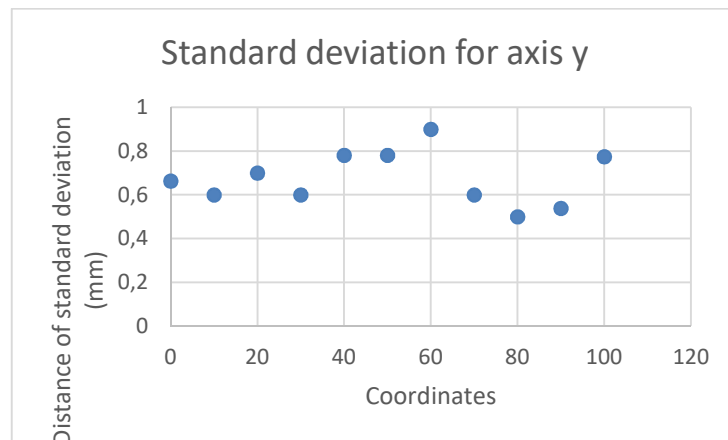


Figure 89: Standard deviation for y axis measurements

The dot in the bottom of the picture in Figure 85 represents a known measured point across all the units systems. This dot was used to relate the data from both of the experiments and produce a joint relation with a simple linear expression for each axis. Before starting to validate the final results there is another source of error that has to be taken into account, the z axis. The z axis is a very simple mechanism but has some error too. In a set of 24 measurements, the pointer was moved up and down. The result is a standard deviation of 0.43 mm. Is not a high error but it adds to the final error.

For the validation of the 3-axis-bridge system a new measurement experience was done. The setup of the experienced was reproduced the same way of the previous ones, and Figure 90 presents a picture of the new millimeter sheet used. The rectangle represents the estimated range of the bridge considering the resolution of 10 mm. The experience consisted in getting the desired pixel coordinates on the pc, convert them into coordinates, send the command and measure the results in millimeters. In Figure 91 and Figure 92 there is the results for the comparison between precision and accuracy, for each axis. The number of measurements are the points illustrated in the picture that are inside or on the limits of the rectangle box. Note that in this particular experiment it was considered precision as the standard deviation, and accuracy as the difference between the target and the measured distances. The mean of the precision error measured for x axis is 0.82 mm, accuracy is 0.77 mm. The mean of the precision error of y axis was 0.57 mm and accuracy of 0.5 mm. It can be confirmed again that the x axis appears to have a larger imprecision than the y axis. The same conclusion can be made about the accuracy. To put the precision into perspective, an average plant has 5.87 mm on this prototype. This means that the relative precision is 14% for the x axis and 10% for y axis. In relation between precision and accuracy there can't be any conclusion made as they represent different characteristics of the system. They are displayed in the same graphics to see exactly that and also to save space as they both have close units range. It would be expected that the accuracy line had more linear appearance, although it is not the case in this experience. The main reason on why this is not the case is the same main reason on why the precision varies along the rail. The bridge structure is not perfect and has some physical irregularities. Also, from experience to experience the accuracy and precision can vary for the same point on the floor. This is because there can be a slight error of the calibration system. The moment where the calibrating sensor is intercepted by the carrier and it resets the origin point can vary, although the error is in the order of one count pulse of the encoder, which corresponds to 0.57 mm on x axis and 0.42 mm on y axis. These values correspond to the equations in Figure 87.

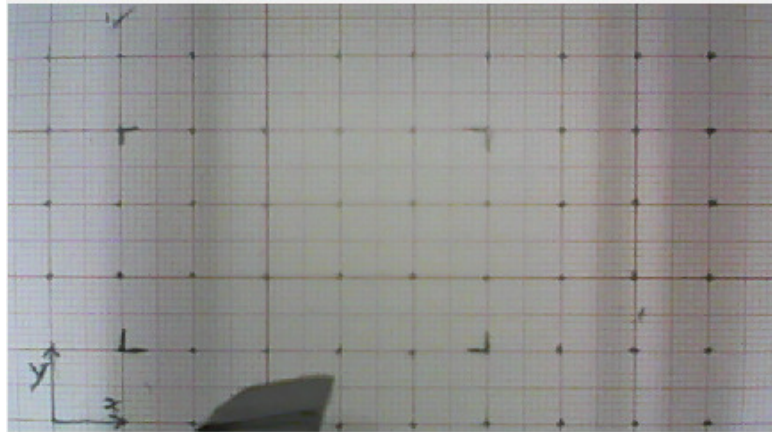


Figure 90: Millimeter sheet used for the coordinates validation

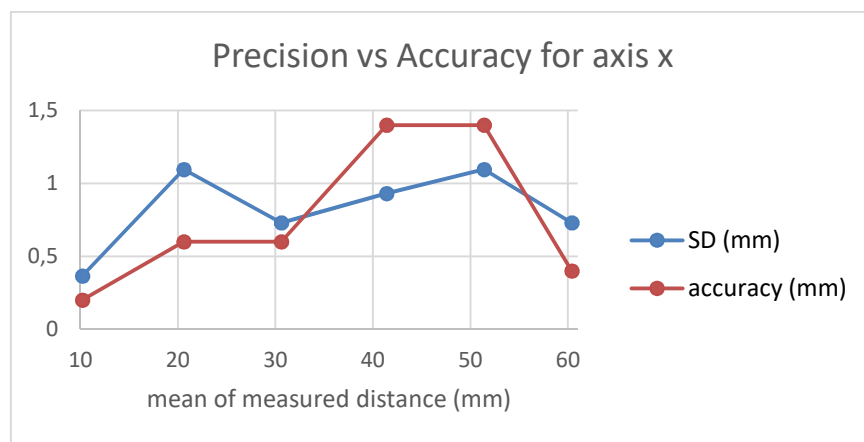


Figure 91: Precision (SD) vs Accuracy for x axis

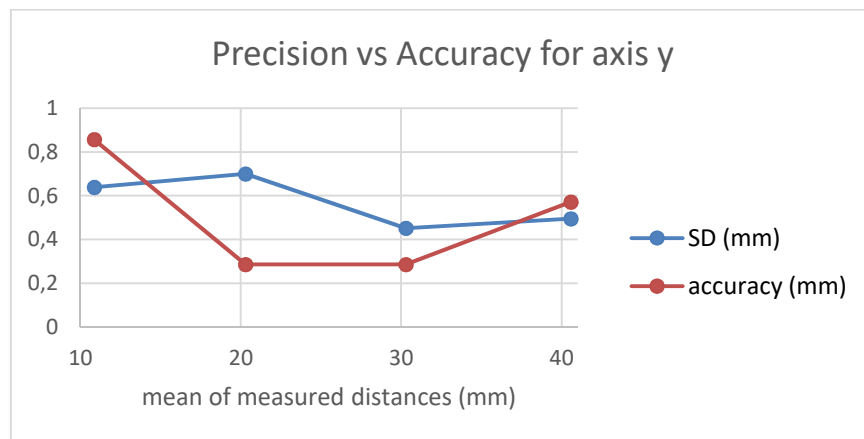


Figure 92: Precision (SD) vs Accuracy for y axis

5.7 Image recognition – shape validation

In the shape validation algorithm, there are cases in which the tested plant is round and is validated as plant. In other cases the plant may be round enough but is too small to be validated. A solution to deal with this

problem would be to limit the minimum value of the radius tested. This would cause a large calculated error, as Figure 93 demonstrates. In this case, the absolute error is large because its calculation is based on the comparison of the plant's curve with a curve function with radius of 30, when in reality, looking at the graphs it is around 6 and 10 for the different axis. A later improvement was a better normalization though the relative error. In these example is dividing the absolute error by the length of the curve array, which in common sense is approximately the diameter of the plant. The last improvement was to divide by the plant's area based on the calculated radius. This is better because the absolute error varies in quadratic way, as the area but the diameter varies in a linear way instead.

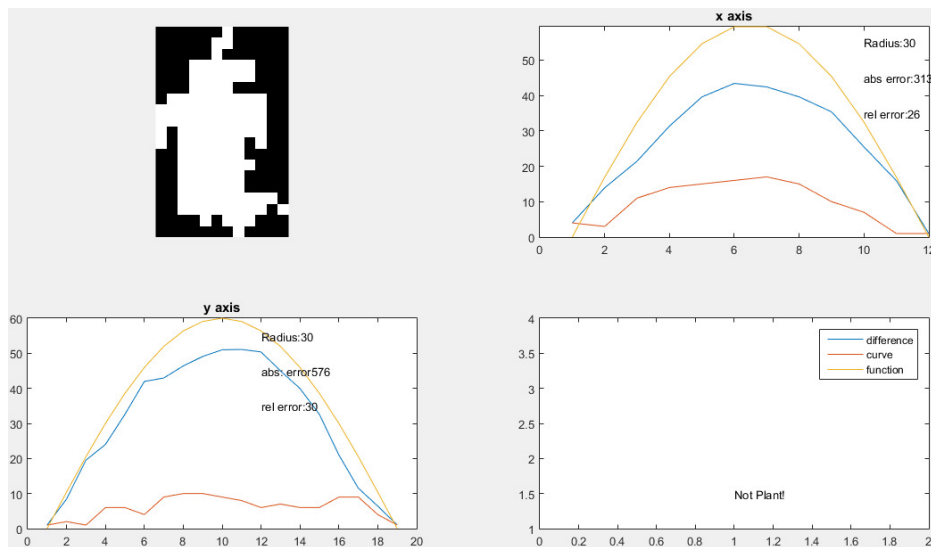


Figure 93: Analysis of a blob shape in an invalid small plant case

In Figure 94 there is an example of a validation of an invalid plant. The blob is not round and therefore the differences array is too high and the errors too.

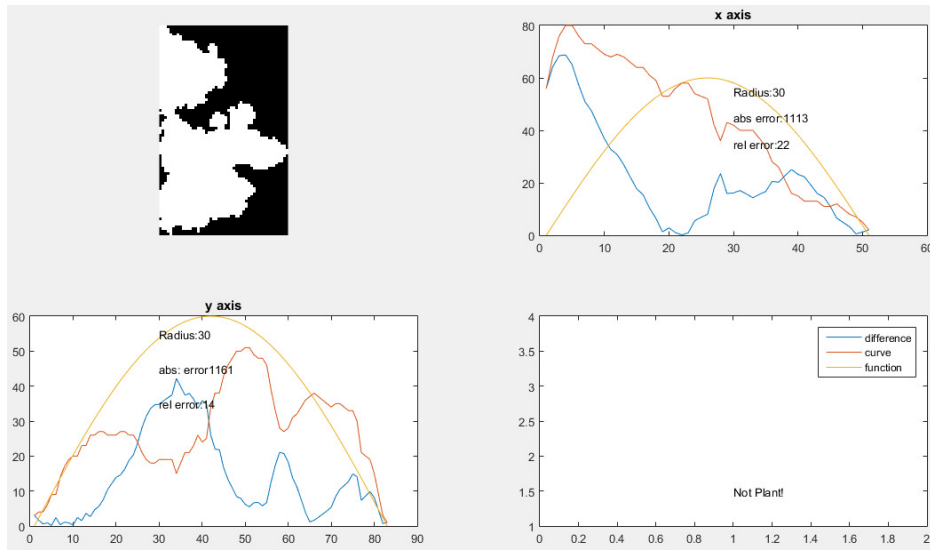


Figure 94: Analysis of a blob shape in an invalid plant case

In Figure 95 there is the next interesting case which represents the main limitation of this system when it works just by itself. This is a blob of a tea plant connected to a weed plant. They are both classified as plants pixels because the color is too similar to distinguish. By introducing this blob into the shape validation program, the results are classified as a non-plant because the blob is no round. In this stage the threshold to decide if it is a plant or not, is the sum of both relative errors that has to be less than 20.

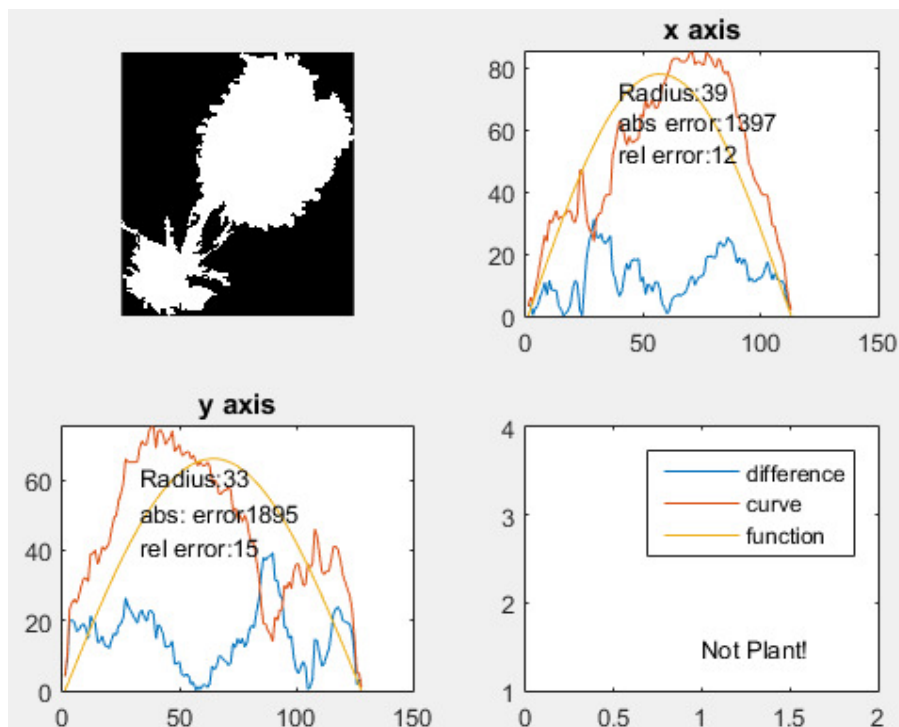


Figure 95: Analysis of a blob shape in a false negative case

5.8 Image recognition – results with loaded images

In order to test the accuracy of the algorithm, the program was loaded with images of lines of plants, such as the examples from Figure 96, Figure 97 and Figure 98. These examples represent sections of a line of plants in different health conditions. The first image of each figure is result of a middle step of the recognition for that image. The blue circles represent the validated circles as plants before the overlap conflict exclusion process. The red circles represent the discarded circles for having an error too high to be considered, plus the used circles before a center adjustment. This adjustment was an improvement implemented over the first version tested in section 5.9. The adjustment consists in recalculation the center of the plants instead of using the centers values returned by the circle finder algorithm. The second image of the figures represent the final result with the recognized centers marked with a black '+' symbol.

The recognition measurements were made across 222 plants that were considered as valid plants, throughout eight of this images. There are very few cases that the recognized center is not right, this is, it is indistinguishable from the human recognition. But there are some cases that a shift in the determined position happens, for example the bottom middle plant in Figure 98. The accuracy error, this is, the total mean shift deviation from the real center of the plant and the recognized calculated was two mm. These pixels were converted into mm to have the same measures as the results from the robot. This scale relation considers that the width of 45 mm, which is the same as the printed images on paper. The precision error is not possible to measure with loaded images as the recognition algorithm always gets the same results for the same images, which leads to a precision error of zero. The recognition success rate obtained was 99.95%. In these 222 examples there was one case of a false positive, that is, a recognized plant that should not have been recognized. This case is present in Figure 99. This is a case where a good plant dried out most of the area and is recovering on just one side. The current program is not able to differentiate such cases. In a real situation this can't happen as the harvester would collect non-good material besides destroying the plant. The possible solutions are to design the algorithm to detect these cases or the farmer has to remove the plant and substitute for a new healthy plant.



Figure 96 Image recognition results example 1

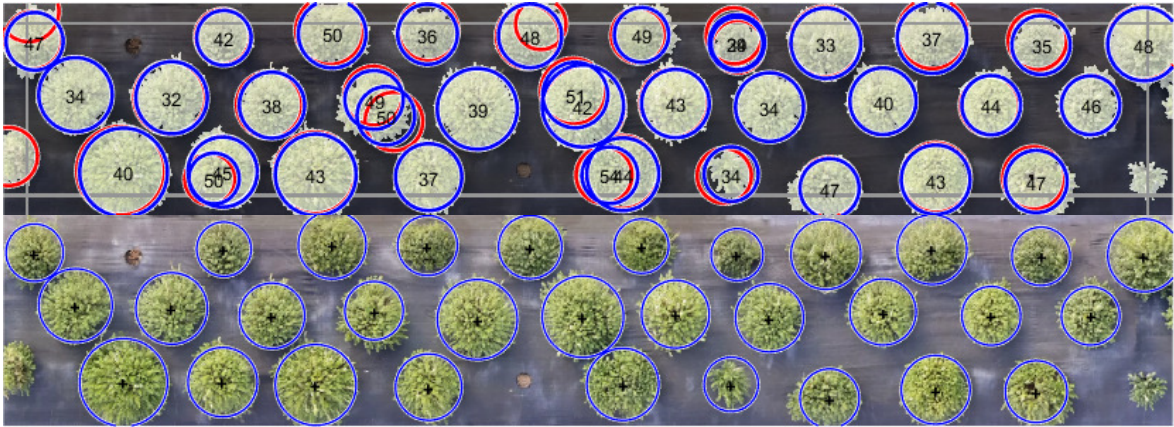


Figure 97: Image recognition results example 2

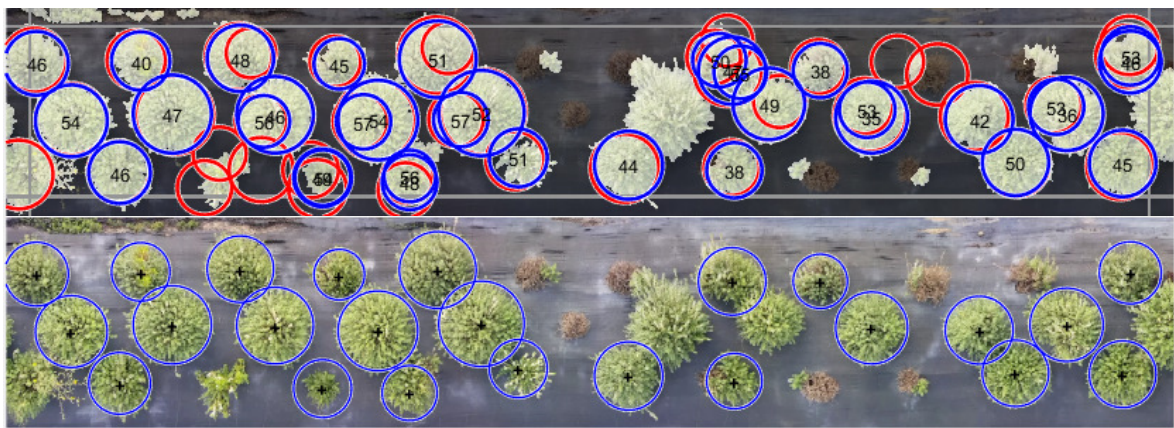


Figure 98: Image recognition results example 3



Figure 99: Image recognition, false positive example

5.9 Image recognition – results on robot version 1

In order to validate the robot’s recognition capabilities there are some features that have to be evaluated through experimental results and the analysis of them. There are two versions of these results because in between these results the robot’s behavioral was improved and many small bugs were fixed. Note that there are two versions of results of the Image recognition final results. The second version are the measurements after several improvements.

For the precision measurements the two images in Figure 100 were printed and glued together, as they are the same image split. The precision measurements consisted in letting the robot go over and mark the recognized plants with a pointer. The program stopped each time the pointer was down so the pointed position could be marked on the paper. After the robot points to all recognized positions in one cycle, to simulate that the plants were marked, there was a paper sheet covering them. This way the robot could advance again until it detects new plants in range. This experience was repeated eight times with new printed plants of the same line. The marked positions were measured with a ruler with a minimum scale of one mm. The obtained average standard deviation for each plant was 2.88 mm. The average radius of a plant on the paper is 5.87 mm. This means that the error is 49% of precision of a plant. The average radius was measured from 72 plants with a ruler of with one mm of resolution. The error is extremely high in comparison with the size of the plants, and thus there is a motivation for the improvement and the second round of results.

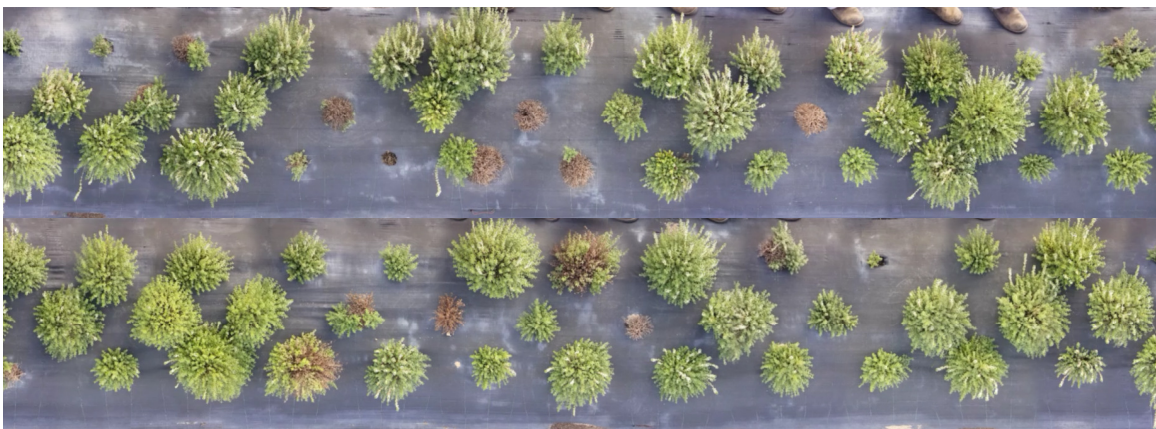


Figure 100: Printed images with plants used for recognition

The accuracy experiment used the setup illustrated in Figure 101. The robot advances over the track and the plants are marked manually as they get pointed out by the robot's pointer. The used paper sheet has a width of 4.5 cm, two meters length with 198 recognized plants.

For the measurements of the accuracy it was used a ruler with a higher resolution (0.5 mm). This paper sheet has regions with more and better quality plants and others harder to recognize. Each plant was processed by the robot once and the pointed spots were marked manually. Figure 102 shows one section of the used sheet after measurements. The red dots are the human considered spots as centers and the blue are the ones marked by the robot. The differences between the blue and the red dots were measured for each axis. The accuracy value was calculated using the Equation 17 with the means of x and y differences as input. The result is 1.35 mm, which is 23% of the average radius of a plant. In Figure 103 there is the histogram that represents these results. It is noticed that the y axis values are more condensed around 0 and the x values are more distributed. In average the axis y is more accurate than the x with 0.24 mm of standard deviation for x, and 1.32 mm for y. Another value reference is the mean of the absolute deviation from the exact value, this is, the mean of Pythagoras expression applied to the measured differences. This results in 2.53 mm. There is the histogram

that represents these errors in Figure 104. The majority of the marked centers are located in a distance of 2 to 2.9 mm of the real center. It is possible to determine the precision through this experience too, assuming the center of the plants as static correlated references. This represents the precision across different environments. It is calculated through the Equation 17, with the standard deviations of x and y differences as inputs respectively. The result is 2.94 mm, which is slightly higher than the one calculated in the precision experiment. This is due to the various environments and to the uncertainty of the measurement instrument.

The plants above a certain radius and in reasonable conditions were all considered as accepted recognizable plants. Out of the 194 plants 94% were recognized.

$$\text{combinederror} = \sqrt{x^2 + y^2}$$

Equation 17: combined error (Pythagoras)

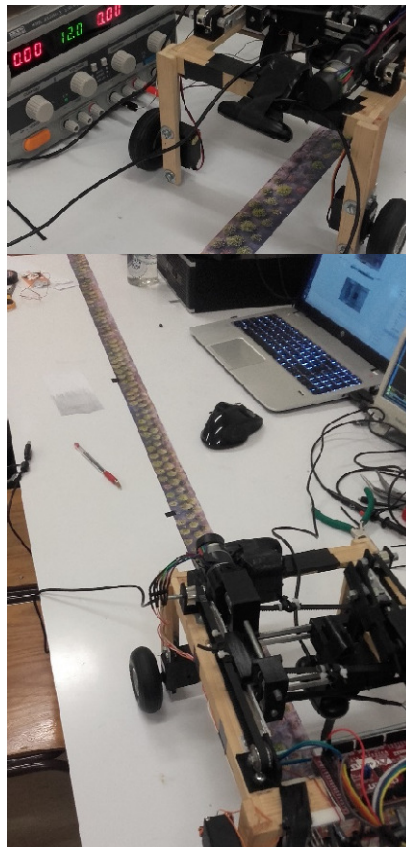


Figure 101: Setup for the accuracy measurements



Figure 102: Sheet's segment with the recognized results

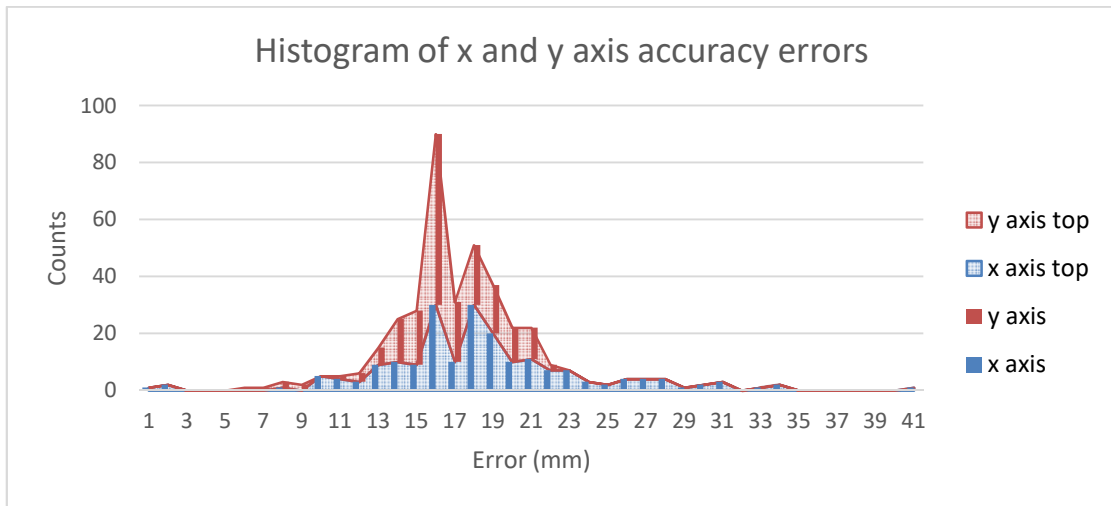


Figure 103: Histogram of x and y axis accuracy errors

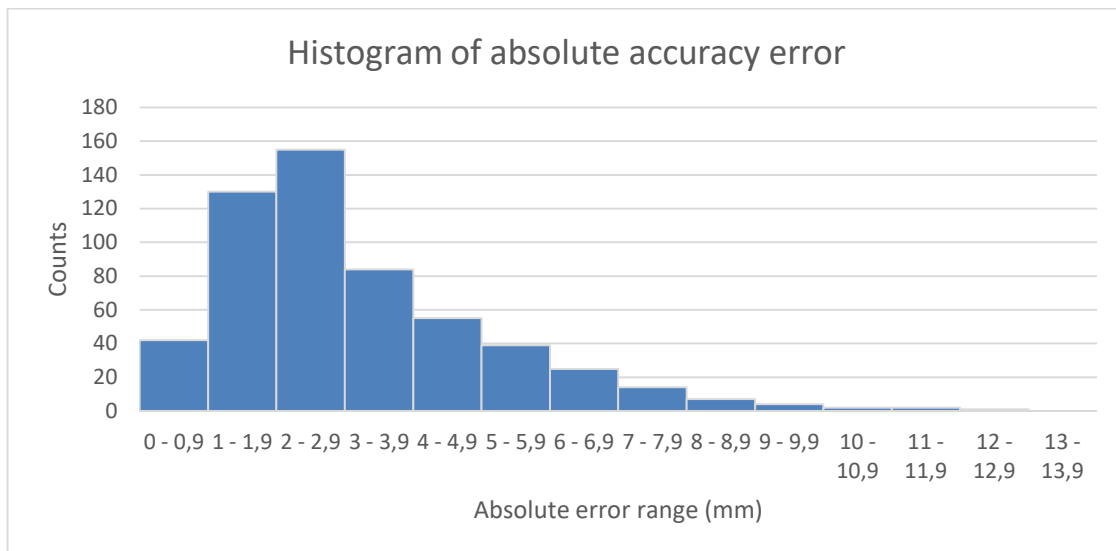


Figure 104: Histogram of absolute accuracy error

5.10 Image recognition – results on robot version 2

Here there is the second version of the robot's results. The main differences between both versions is that the 3-axis-bridge study was made in more detailed way, the recognition software was improved, the bridge

mechanical structure had some imperfections fixed or improved and it was noticed that the recognition system gets better results with higher light intensity.

The experiment consisted in letting the robot run its program over a printed line of plants and mark the pointer's position manually for further measurements. For the precision measurements there were made with 14 printed equal segments of plants. Each segment with a length of 24.7 cm and 28 recognizable plants. The total number of measurements to determine the final precision was 381. The setup used is displayed in Figure 105. Each sheet was duct taped to the table, along with two papers on each side with black stripes. These stripes were designed to assure that the robot does not classify areas right next to the paper as plant's pixels. The paper sheet covering half of the line is designed to simulate the marked points after the robot marks all the plants in the current area, before advancing for the next. In Figure 106 there is an example of the recognition results. The area inside the black lines represents the bridge range. The area inside the gray lines represents the valid area for recognized plant's centers. In order for a plant to be marked by the bridge it has to be located inside both areas. The image of the figure represents the recognized plants ready to be marked. The text on the sides represent this list of plant's centers coordinates in the pixel scale on the left and on the bridge coordinates scale on the right.

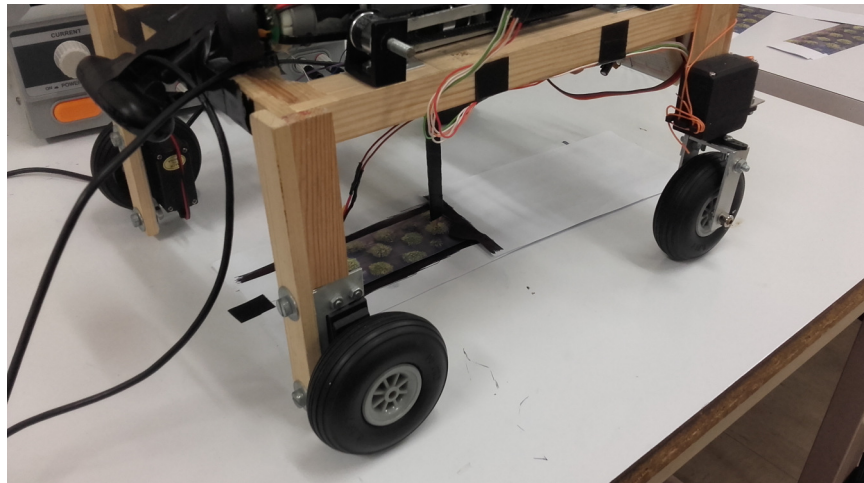


Figure 105: Setup for the final precision and accuracy results

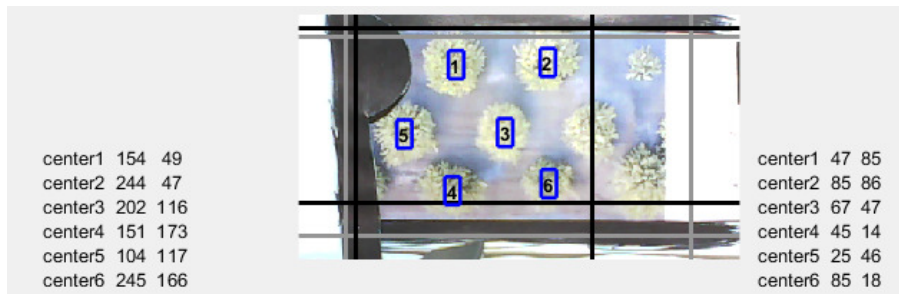


Figure 106: Program recognition results display example

During the experiments it was noticed that the brightness affects the results significantly, as Figure 107 demonstrates. On the left there are the results with normal lab brightness conditions and on the right there are the results with extra brightness. The extra brightness was achieved by pointing a phone's flash on the paper sheet. On the left case, the brightness exposure parameter was already defined to the maximum sensibility. On the right case, the sensibility was decreased a little because with the extra brightness it did not needed to be defined to the maximum value. Not only in this example but in general, it was noticed that in normal conditions the plants on the bottom side of the image had a tendency for a lower success recognition rate. Note that in extra brightness conditions, the white paper and table are recognized as plant's pixels. In the classification system there is not any class for table pixels. With more brightness the result is that the table pixels features are closer to the plants pixels than from dirt or plastic fabric.

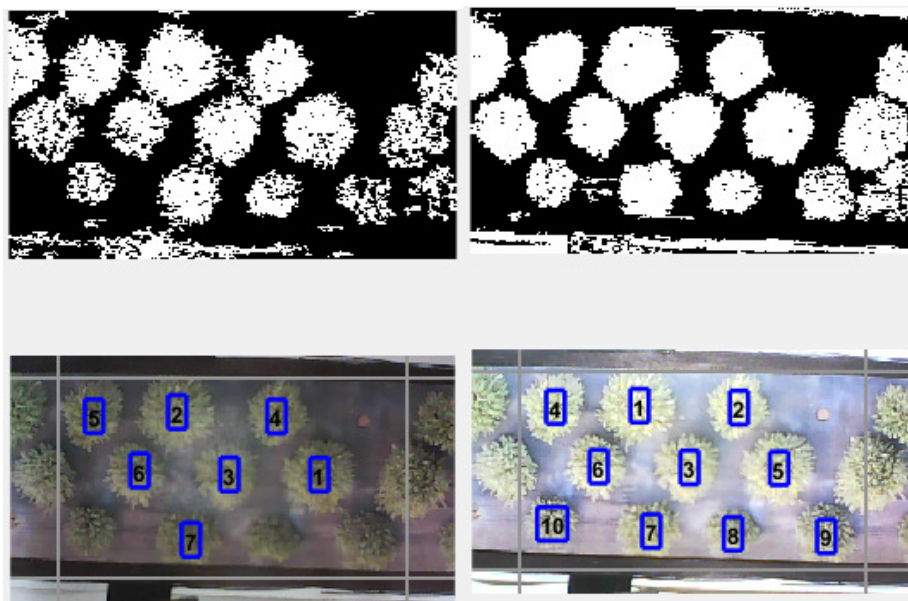


Figure 107: Comparison of the recognition and pixel classification between normal lab brightness and extra brightness conditions

After the experiment the distance marked centers were measured to the same reference position with a ruler with one mm of resolution. The measurements for the x axis and for y axis make in total 762 values (2×381). For each one of the 28 plants there was calculated the value of the standard deviation for x and y measurements. The mean of these imprecisions was 1.31 mm for x axis and 0.78 mm for y axis. The combined error can be calculated through the expression in Equation 17, which results in 1.55 mm. This value is relatively better compared to the radius of the plant, which results in a ratio of 26%.

The measurements for the accuracy experiments the ruler used had a higher resolution of 0.5 mm. The setup used is similar to the precision measurements, but the used paper sheet has a width of 4.5 cm, two meters length with 162 recognized plants. The conditions and explanations for the calculation of the values were described in section 5.9, so their explanation aren't going to be repeated. Therefore the results will be displayed in Table , with the discussions following it.

Accuracy	0.54 mm
Accuracy %	9.2 %
Accuracy of (x ,y) axis	(0.38, 0.39) mm
Absolute accuracy mean	1.46 mm
Accuracy deviation (precision)	1.68 mm
Recognition success rate	95%

Table 6

In Figure 108 there is the histogram representing these results. It is noticed that the y axis values is more condensed around 0 and the x values are more distributed. In average both axis have a similar accuracy.

Figure 109 presents the absolute accuracy deviation histogram. The majority of the marked centers are located in a distance of 1 to 1.9 mm form real center.

The plants above a certain radius in reasonable conditions were considered recognizable plants. In this version of the program the recognizable radius threshold chosen was slightly higher. This choice was made because the program was changed from the first version to the second to be stricter. The images used to test the program on the computer had slightly less length in the image pixels loaded on the PC program, therefore the predefined radius threshold was considered proportionally larger. Out of the 162 plants 95% were recognized.

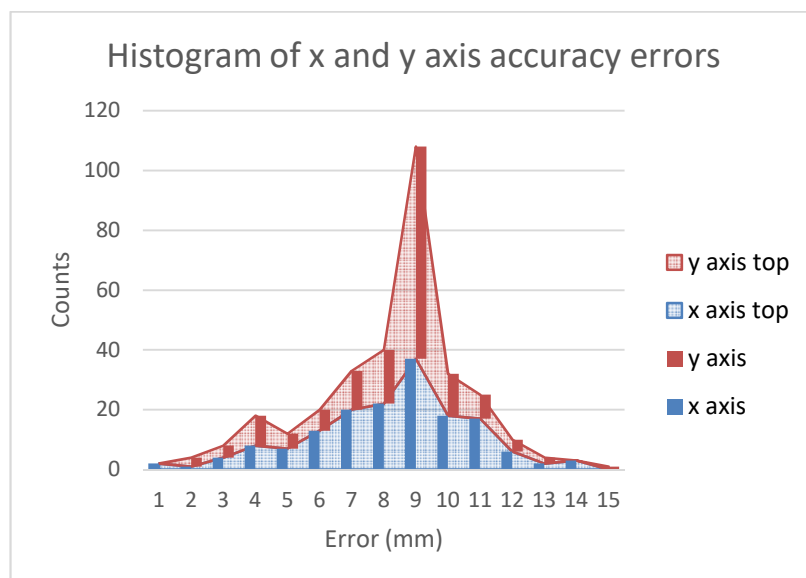


Figure 108: Histogram of x and y axis accuracy errors

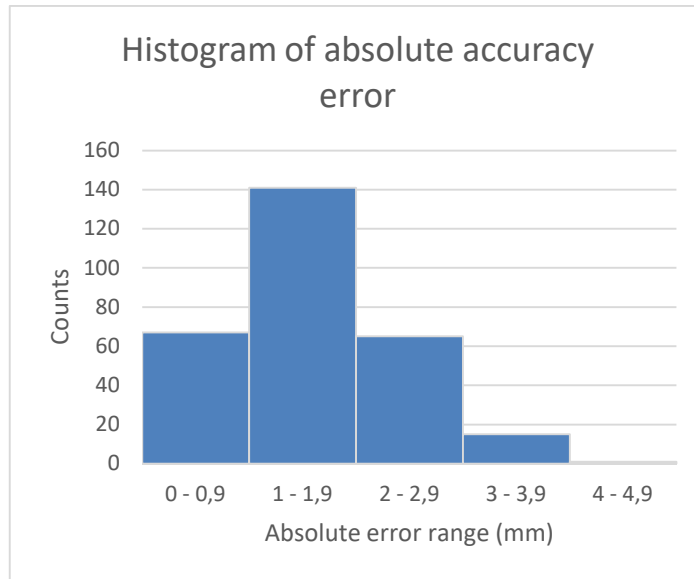


Figure 109: Histogram of absolute accuracy error

5.11 Visual odometry - motion tracking

As it was described, the visual odometry solution for tracking the robot's position did not work for this system. The slight changes in brightness and the quality of the camera are possible reasons for it to fail. The motion algorithm could be refined but there was not enough time to invest in a solution that is most likely to produce many errors anyway. It was also not a good solution for the real machine later as it is more sensitive to the environment and its irregularities. To measure the odometry error there was a comparison between the accumulative expected estimation motions with the accumulative calculated motions, which the results are in Figure 110. The experiment consisted in two parts. The first part consists in taking pictures with the camera on the robot and marking on the table the positions. These pictures were processed and compiled with the program on the computer, similar to the example in Figure 56. These represents the data of the accumulative expected estimated motion. The second part consist in running the part of the program that calculates the motion, takes pictures and move the robot at the same time. The program waited for a manual keyboard signal to take a new picture and calculate the motion. While the robot was in motion this signal was given when it passed above the previously marked positions on the table. The program with the loaded images previously loaded before running, calculated a total motion of 51 pixels along 19 pictures. The second program calculated a total motion of 17 pixels. This experiment was repeated several times and the second program always returns very different values, as Figure 111 represents. The final results of the different repetitions were 17, 24, 9, 19 and 12 pixels for the same distance, which is a very big difference between results. The first set of data is reliable because we know this program worked with loaded images. We know that it works with loaded images because the differences from a shifted image on the computed, result in a very close motion from the taken pictures manually and in static position and then loaded. The composed image of another example of this algorithm was displayed previously in Figure 56.

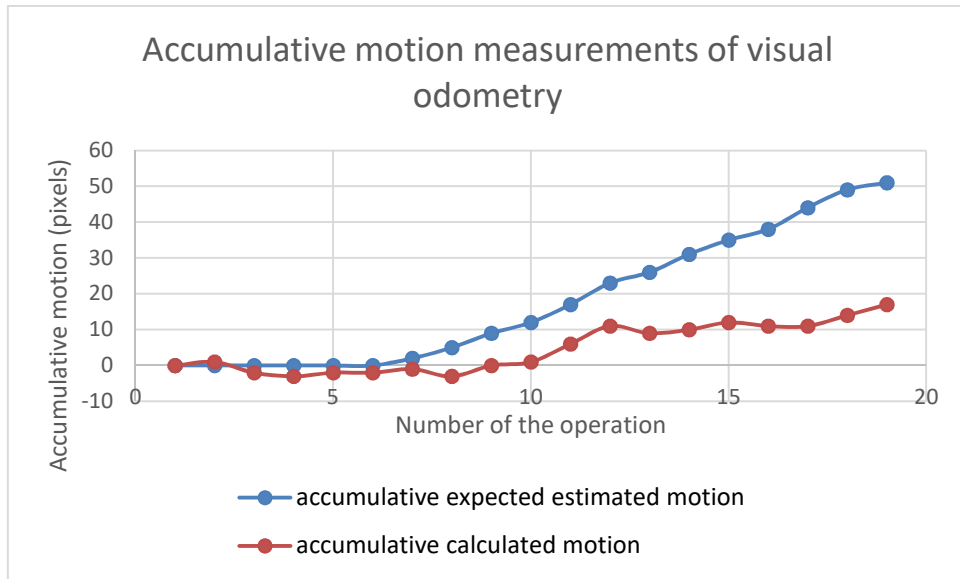


Figure 110: Accumulative motion measurements of visual odometry

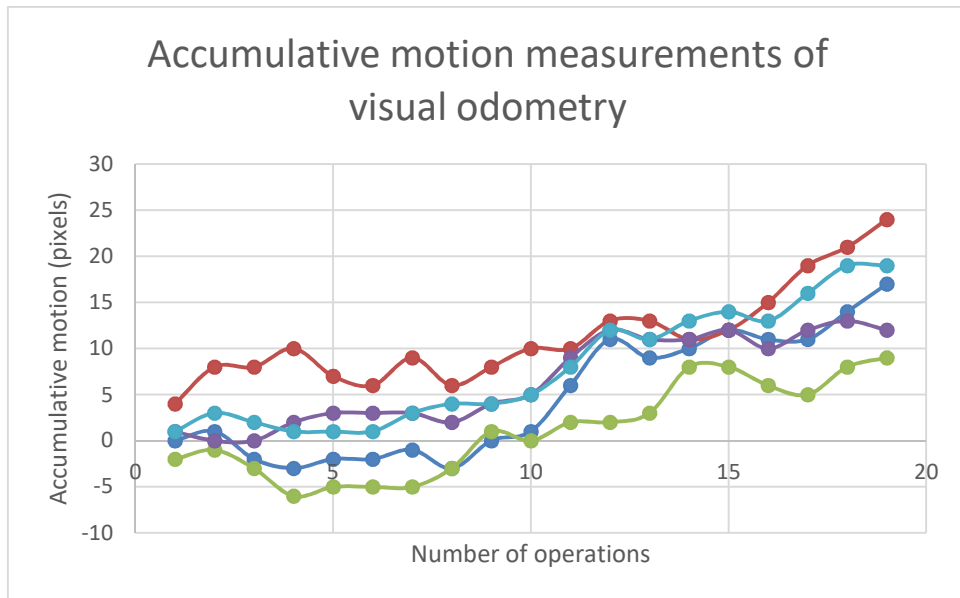


Figure 111: Accumulative motion measurements of visual odometry measurements

5.12 Image recognition – time efficiency

The processing time of the image recognition algorithm has to be measured too. The conditions for this measurements were made in a laptop with a processor Intel® Core™ i7-4700MQ CPU @ 2.40GHz. The operating system was windows 10 64 bits in the program Matlab. The measurements were made while the computer was working at 4% of normal CPU usage.

The images chosen for this measurements were all different and had the equivalent size of a picture taken by the camera positioned on the robot. In 24 images the average recognition time is 659 ms per image. The standard deviation of the measurements calculated is 38ms.

In the development of this project the processing efficiency was not the focus at all. The goal was to develop an algorithm that could recognize the plants in images properly. The programming language Matlab is a high level language with practical libraries to develop an algorithm, although they often aren't efficient regarding the processing time. Future developments can include to reformulate the code in order to make it more efficient. The MATLAB code can also be translated to other more low level language, such as c++, for example.

5.13 Alignment Recognition

In the "Implementation" section the general process was described. Here the purpose is to validate the process results, to explain in what does the validation consists of and how. There are three sets of measurements to define different type of specifications, such as precision, accuracy and the study of the image distortion and determine the correction adjustment.

It is reasonable to start with the study of image distortion as it influences the precision and accuracy values. In order to study the precision the robot was positioned in several positions above the line of plants, with the same alignment but with different lateral shifts. The result of seven measurements is in Figure 112. The values used for the measurements were the real values of the line's alignments. The pictures taken were not all with the robot positioned exactly at the same angle, and so it may cause some error in the linear regression, reason, why the points aren't all aligned.

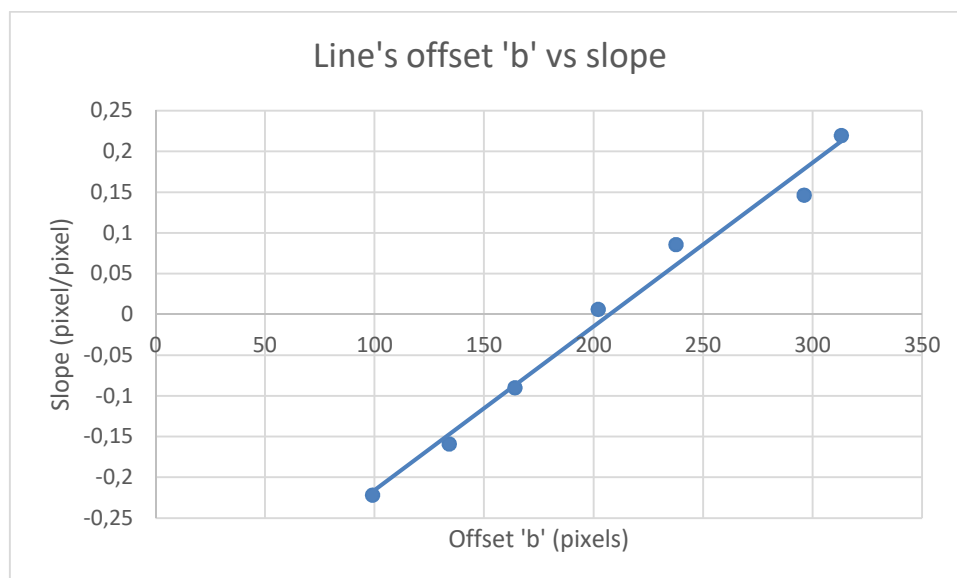


Figure 112: Line's offset 'b' vs slope

The result of this of this linearization generated a value 'mc', which is the proportional correction constant, and the 'bc', which is an offset correction value. The expression used for correction is expressed in Equation , where 'm' is the slope value to correct and 'b' the value at which the line intercepts with the origin. Since the distortion depends on the lateral position, it makes sense to make the adjustment proportionally to the line's offset too.

$$angle \in degree = Atan(m - (b * mc + bc)) * 360 / (2 * pi)$$

Equation 18

In Figure 113 there is an example of the correction algorithm implemented. The image is shifted laterally but still aligned with the plant's line in real. In real recognizing an alignment of -10 degrees is wrong, although the correction formula corrects it to zero.

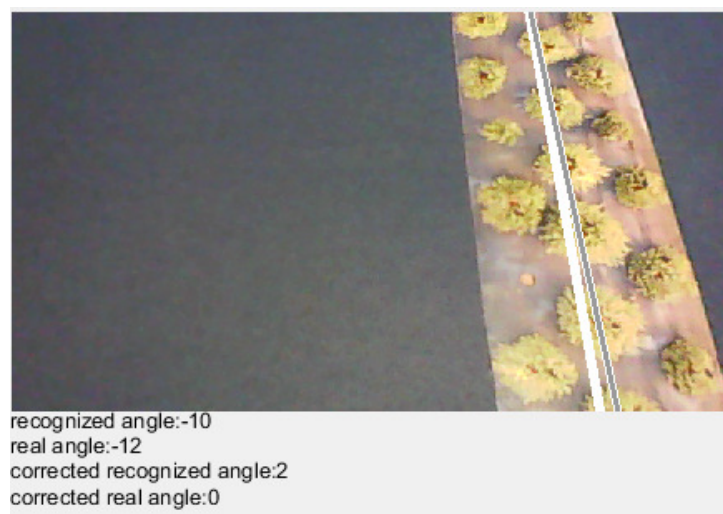


Figure 113: Angle correction example.

Next comes the study of the precision of the robot's alignment software recognition. This experiment consists in, with the robot and the printed pictures fixed in a static position, take and process several images of different sections of the line of plants. There were 12 measurements of different sections of plants each, with 12.5 cm each. As experiment reference, the average corrected value of the real angle position was 2.7 degrees, with a standard deviation of 0.5 degrees. The recognized average alignment was 1.8 degrees with a standard deviation of 4.9 degrees. Possible reasons for existing a precision error may be because the plants aren't located uniformly on the field in different sections, thus the correction value can have some impact, and the nature of the algorithm in general.

The accuracy experiment consisted in taking account the results of two different experiences and combine them with the same weight to the calculation of the accuracy error. The first one considered was the same as the precision experiment but different values were considered. The accuracy error of this component is the average of the differences between the corrected real angles and the corrected recognized angles. This results in an accuracy error of -0.31 degrees. The second component consisted in varying the alignment of the robot

and process the images for the same section. This experiment resulted in an accuracy error of -2.15 degrees. Figure 114 presents a graphic relating the accuracy error with the recognized angle. It is noticed that the accuracy error tends to be positive for positive values of alignment and negative for negative alignments. It is suspected that this is caused by the correction, because when a line inclination is slightly different from the real, the corrected value that corresponds to the robot alignment may be more different. This happens because the value of the offset is different too and that same inclination may have an even larger difference in reality. To visualize this error we can take a look at Figure 113 again. This difference is clearly visible in Figure 115, which has a more random variation regarding the angle. Note the angle variation is related to the side at which the offset is located, because the pictures used to perform this experiment also followed this relation. The accuracy error without the alignment correction results in 0.36 degrees, which is much less than -2.15 in absolute value.

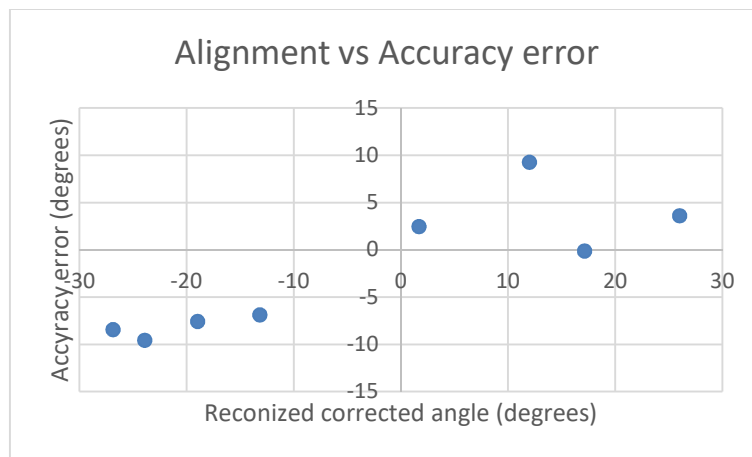


Figure 114: Recognized corrected angle (alignment) vs Accuracy error

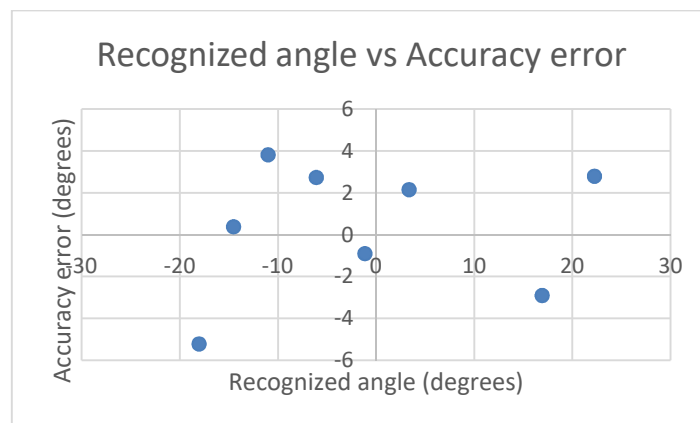


Figure 115: Recognized angle vs Accuracy error

6. Conclusions

In overall the development of this project included a wide range of different areas and features that had to be, designed, implemented and tested, in order to build a working prototype.

The prototype ended with the following capabilities:

- It has the capability to communicate commands between the robot and the micro-controller;
- It has the capability to move the robot at a certain speed straight forward through the traction motors movement;
- It has the capability to pull the pointer of the bridge up and down with a precision of 0.47mm ;
- It has the capability to position the direction wheels in the desired angle with a precision of 3.1% for the left and 3.9% for the right servo;
- It has the capability to recognize the robot's alignment with the line of plants, with a precision of 4.9 degrees, and an accuracy of 2.15 degrees
- It has the capability to position both axis motor's in the desired position, accurately;
- It has the capability to position the pointer on the desired position on the robot, with a precision of 0.82 for x and 0.57 mm for y axis;
- It has the capability to calibrate the axis motors position;
- It has the capability to classify images from potential plant's and no-plant's pixels;
- It has the capability of recognizing the plants with a success rate of 95%;
- It has the combined capability of covering areas with printed plants and point to the recognized plants with a precision of 24% (1.55 mm) and an accuracy of 9% (0.54 mm).

One limitation that this project had in the thesis context, is the amount of total work for the available time. There is not any individual component developed in this thesis that is new to the world and too much advanced, but the amount of different components to develop, implement and validate is large and consumes lots of time. Anyway it was my idea, choice and responsibility.

Further developments on the prototype are the fully implementation of the robot's alignment into the main program in order to introduce the capability to adjust the robot's direction. The processing optimization of the recognition software can be developed. The next continuity is to develop the real scale version which is actually going to harvest real plants and is the main end goal and motivation for this project. Giving the capability to harvest delicate MAP with the combination of the benefits of a human's and machine's harvest. Its end advantages are better usage of the plantation area in a field, better leaves quality and economically

more profitable, than the current solutions available in the market. By time as the number of workers that want to work in the field are decreasing while the manual labor costs are increasing. This raises the value of the delicate plants in the market. This is a perfect opportunity for the introduction of this robot into the market and make a difference. The real scale robot has a size of two by two and a half meters with adjustable widths. It includes a special harvesting blade which cuts the plant in the round form. It includes a leaves sucking system, that suck the cut leaves into a bag. It may be powered by a generator at first for development and further by batteries and solar panels. In the end, it is possible to develop a localization system through GPS and google maps where the farmer can select an area for harvesting.

Note that in this prototype project the scaling between the prototype components and the real scale projected prototype was not proportionate. It is not possible to dimension the components proportionally as the physics in a greater scale work different as for a smaller scale. Some components have also a minimum size to be workable, such as wheel, band wheel, rubber band, rail dimensions of the 3-axis-bridge, motors, etc. In the real scale prototype the relation between the range of the 3-axis-bridge system and the size of the robot has to be better, that is, there is going to be less unused space of the bridge size in proportion. This happens because the size of the basic structure will increase more than the size of the motors and technical components.

Another conclusion related to the real scale prototype is related to the motors and cameras used. The limitations of the used motors is related to the DC motors for traction and to the servo motors. The DC motors as implemented don't allow tracking of the movement which could be useful. The servo motors used have a precision that is not enough for a final product. The camera used for recognition takes pictures from only one perspective for recognition. The plants that aren't right under the camera don't appear round in the camera has there is some distortion and the plant itself can have a higher branches that hide the side borders and distort the perceived plant shape.

7. Bibliography

- 1- <https://prezi.com/fpfumhuy60yq/stages-of-agriculture/>
- 2- <https://mahtabrasheed.wordpress.com/2012/11/14/steps-a-farmer-performs-and-what-information-is-required-at-each-step/>
- 3- <http://epam.pt/produtores/>
- 4- <http://www.anaheimautomation.com/manuals/forms/encoder-guide.php#sthash.PUREhxTI.dpbs>
- 5- <http://www.robotoid.com/appnotes/circuits-quad-encoding.html>
- 6- MAP-EXPO, Global marketplace for Medicinal & Aromatics Plants
- 7- <http://www.sweeper-robot.eu/>
- 8- <http://octinion.com/products/harvesting-series/strawberry-picking-robot>
- 9- D'Esnon A Grand, et al., Magali: a Self-Propelled Robot to Pick Apples, American Society of Agricultural Engineers 1987
- 10- Harrell R C; Adsit P D; Munilla R D; Slaughter D C, Robotic picking of citrus, 1990
- 11- Duda, Hart et al. 2000
- 12- Peilin Li, Development of a Bi-camera Machine Vision System to Identify the Citrus Fruit for Automatic Harvesting System, 2014
- 13- Rabatel G; Bourelly A; Sevila F; Juste F, Robotic Harvesting of Citrus: State-Of-Art and Development of the French Spanish EUREKA Project, 1995.
- 14- Ceres R, et al., Design and implementation of an aided fruit-harvesting robot(Agribot), Industrial Robot, vol.25, no.5, 1998
- 15- Muscato G; Prestifilippo M, A prototype of an orange picking robot: past history, the new robot and experimental results, 2005
- 16- Peilin Li; Sang-heon Lee; Hung-Yao Hsu, Review on fruit harvesting method for potential use of automatic fruit harvesting systems, 2011
- 17- Jimenez A R; Ceres R; Pons J L, A Survey of Computer Vision Methods for Locating Fruit on Trees, 2000
- 18- DUDA, R.; HART, P. Use of the Hough transform to detect lines and curves in pictures, 1972
- 19- Alexandre Manuel Mota DETI-UA, 2017; Sistemas de Control 2
- 20- http://solarbotics.net/starting/200111_dcmotor/200111_dcmotor2.html
- 21- <https://electricalstudy.sarutech.com/working-or-operating-principle-of-dc-motor/index.html>

- 22- https://www.mouser.it/publicrelations_techarticle_rotaryencoderscriticalmotco_2015final/
- 23- https://en.wikipedia.org/wiki/H_bridge
- 24- R. Duda, P. Hart, D. Stork: Pattern Classification, Wiley
- 25- https://en.wikipedia.org/wiki/Hough_transform
- 26- José Luis Azevedo; Arnaldo Oliveira; Tomás Oliveira e Silva, Arquitetura de Computadores II, Guia das Aulas Práticas, Trabalho prático N.º 6
- 27- https://en.wikipedia.org/wiki/Standard_deviation
- 28- <https://www.citisystems.com.br/motor-cc/>
- 29- <http://www.circuitstoday.com/types-of-chopper-circuits>
- 30- MAIMON, Felipe. Projeto de um Sistema Eletrônico para o Controle de Motores de Alta Potência por PWM. PUC Rio. 2004.
- 31- <https://www.hitecnologia.com.br/blog/o-que-%C3%A9-encoder-para-que-serve-como-escolher-como-interfacear/>
- 32- <http://www.dpi.inpe.br/spring/teoria/filtrage/filtragem.htm>
- 33- http://www.geelmed.com/documents/products/producto_000000418DOC.pdf?d=1431189324