Development of a small robot prototype for educational purposes

Emanuel de Jesus Soares Marta

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

Work oriented by:

Prof. Dr. José Lima

Prof. Dr. José Gonçalves

Prof. Dr. André Oliveira

Bragança

2021

Development of a small robot prototype for educational purposes

Emanuel de Jesus Soares Marta

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

Work oriented by:

Prof. Dr. José Lima

Prof. Dr. José Gonçalves

Prof. Dr. André Oliveira

Bragança

2021

Acknowledgement

First, I have to thank God.

I have to thank my teacher for the teaching quality of the highest order, all ways clear and straight, and for the knowledge provided from the curriculum until the real job market and beyond. Words can not express, and my gratitude is immense.

For my friends, for those that finished, for those that are ending, for those that never walk the phat, and for those that had fallen in the journey *"coração para Sempre"*, Forever.

For my family, from birth till here, the path was hard. Some have left us during the journey, a moment of silence for them, ..., for those that made it thank you for coming whit me to the starting line, all ways together never forget.

For Cantina of IPB and its staff, for "os combatentes do COVID" we work together we share food and drinks noting need to be said. I salute you.

For the city, for IPB, better is can always be found, but it was good I would've come again, thank you. My people, Thank you.

MAMA, it's finished.

Abstract

The project consisted of simulating an Automated Guided Vehicles (AGV) system and creating a model of an AGV that will serve as a teaching kit for future generations. The simulation was performed in SIMTWO, a realistic robotics simulator. Three simulations were built of increasing complexity starting from the simple line movements to AGV systems. It was plotted in each simulation the most important graphs to present the reader and explain what has happened. Moreover CAD files were created of the AGV using 3D printing and assembled. Then, a software that runs on the Arduino, was developed, and it uses a six-byte combination of symbols and numbers to control the AGV.

Keywords:Automated Guided Vehicles (AGV); SIMTWO; Teaching Kit;Arduino

Resumo

O projecto consistiu na simulação de um sistema de Automated Guided Vehicles (AGV) e na criação de um modelo de AGV que servirá como kit de ensino para as gerações futuras. A simulação foi feita em SIMTWO, um simulador robótico no mesmo, foram feitas três simulações de complexidade crescente a partir dos simples movimentos de linha para sistemas AGV. Foi traçado em cada simulação os gráficos mais importantes para apresentar o leitor e explicar o que aconteceu. Além disso, foram criados ficheiros CAD para o modelo AGV e criados utilizando a impressão e montagem 3D. Estes, um programa que corre no Arduino, foi concebido, e utiliza uma combinação de seis bytes de símbolos e números para controlar o AGV.

Palavras-chave:Veículos Guiados Automatizados (AGV); SIMTWO; kit de ensino; Arduino

Contents

Acknowledgement **v**

Abstract **vi**

Resumo **vii**

1 Introduction **1**

2 State of the Art **3**

2.1 Simulation 4

2.2 Teaching Kits 5

3 Approach **8**

3.1 Architecture 9

3.2 Equipment and software used 10

3.2.1 Software utilised 10

3.2.2 Components Used 10

4 Implementation **13**

4.1 Simtwo 13

4.1.1 Program Simtwo 16

4.2 Hardware 19

4.2.1 Electrical connection 21

4.3 Arduino 23

4.3.1 Architecture 23
4.3.2 Program 24

5 Results **28**

5.1 Simtwo 28
5.2 Hardware 36
5.2.1 Test conducted 37

6 Conclusion **41**

6.1 Future Works 42

List of Tables

4.1 Driver control tabel 21
4.2 Encoded signal 26

5.1 Table of positions 33

List of Figures

3.1 Block diagram overview 9
3.2 Arduino Nano 11
3.3 DC Motor 11
3.4 Driver 12
3.5 Battery 12

4.1 Road Map 14
4.2 Robot 15
4.3 Base plate1 19
4.4 Base plate zones 20
4.5 Base plate2 20
4.6 Graphic representation of the new wheel 21
4.7 Electrical diagram 22
4.8 Motor encoder 23
4.9 Program architecture 24
4.10 Code 25

5.1 Theta of the robot 29
5.2 X & Y of the robot 29
5.3 V of the axis 30
5.4 V of the axis corners 31
5.5 Theta around a corner 31
5.6 X&Y on a curve 32

5.7 X&Y of the LAP 33
5.8 Lap after Y change 34
5.9 The x Values of bots 1&2 34
5.10 The y Values of bots 1&2 35
5.11 Robot 36
5.12 Wheels 37
5.13 Basic control of velocity 38
5.14 Doing curves in arduino 39
5.15 Controlling the robot 39

Chapter 1

Introduction

This work aims to create a prototype of an AGV that is easy to fabricate and to see how it can be made as a robot to be used in teaching kits. The advance in robotics has allowed the big industry to substitute people in warehouses, other assembly line processes, and other places. However, the advantage of saving operating costs and increased plant-floor efficiency can be appreciated by companies of other sizes. Although it depends right now, the cost of a simple Automated Guided Chart (AGC) starts at 14000 USD, and that is just one without the system, maintenance, and the other things that are needed [1]. So we created a prototype of an AGV that costs a lot less than one in the market, and used the prototype for teaching kits that allows new students to develop their critical thinking and attention to detail. Moreover, because of the interactive nature of robotics, this technique is being used worldwide to incentivize students to join the STEM fields. And it's not only useful for STEM students, it can help everyone by letting them use their imagination and possibly find solutions for future problems.

The structure of this works flows a simple guide line of:

1. State of the Art= Which explains how we got here and what innovations happened along the way and where we are;

2. Approach= Which gives us the aim of the project, how we are planning to achieve them what will be used to accomplish them;

3. Implementation= This tells us what was done, how it was done, and explains the important parts and how we went through them;

4. Resuslts= This is what we were able to take from our experiments and what was learned;

5. Conclusion= Final thoughts and future works;

Chapter 2

State of the Art

Automated Guided Vehicles (AGV) are used to transport products or equipment mainly in inter logistics like warehouses, assembly lines ,or transport between different sectors of a building/company. One of the first to introduce this technology was Barret Electronics of Northbrook, Illinois, the USA in 1953, and it used wires implanted in the floor to guide itself. This technology entered the European market via the UK in 1956 and then in Germany in 1960 [2]. In the late 60s, the guidance circuits using vacuum tubes started to be replaced by systems using semiconductors technology. Subsequent data transfer follows via infrared or even radio signals. Ultimately, in the 70s, the demand for AGVs (Automated guided vehicles) would increase due to a combination of increased productivity and higher levels of automation. Now AGVs are used in many places doing multiple tasks and having a lot more contact with people. As an example look at Norway St. Olav's hospital in Trondheim where although task whit moving goods, waste and medical equipment. They have become such an intrinsic part of the hospital that the interaction between the AGVs and people have been studied to see how human actor and nonhuman actor relate to them[3].

At the moment, AGVs are divided according to there route guidance system, free or fixed. Fixed-route uses equipment's like magnetic or optical tapes, which makes it easy to install, but any changes have a considerable cost in time and money. On the other hand free route are the inverse, although there are tests to see if a merger of the two would bring

benefits to the 4.0 industry [4]. However, what is certain is that the technology itself is at a level that companies of all sizes should start looking in, to see if it is something that can be useful for them, like the case of a food company in Italy. Which performed a study to see if it was economically viable to apply it there, concluding that it is. The investigators arrived at three ways to achieve, the first a complete automation of the lines that connect to the kitchen, this requires a purchase of 8 vehicles in which two are spares, in case there are breakdown the system can continue to work. A second scenario in which only one line would receive the AGVs but would guarantee a slow entrance of the technology, and be easier for the operators and management. The third and last scenario only two or three lines would be automated. It was also conclude that the fist option was more expensive, but it had a faster return on investment [5].

2.1 Simulation

Process of simulation is generally used before the implementation of the project, since it provides statistical data to optimize the logistics and create other models for economics evaluation of the operation [6]. This is mainly do to the involvement of several areas such as mechanics, electronics and programming, among others. Nevertheless, it is also used for education, for example, in the training of nurses since it is designed to resemble reality. These simulations contribute to students learning in several ways enhancing clinical skills acquisition and confidence [7]. In robotics, this tool is often used as quick testing for prototypes of robotic hardware and controllers in different types of environments and situations [8]. In robotics education, simulators enable the user to engage virtually with the development and programming of robots through GUIs(graphical user interface), allowing for a decrease in cost since is not necessary to deal exclusively with real robots. However, the suitable simulator is necessary since there are a lot of them, which cover a wide range of needs, are programmed differently and directed at various ages of students like "*RoSoS*", "*Robot One*" and "*Simtwo*" [9].

Although creators/hobbyist often use online simulators, these do not represent with

good fidelity the real world [10] that's the reason programs like Simtwo are created.

2.2 Teaching Kits

Educational robotics is pedagogically grounded in a constructivist approach because it inherently incorporates traditional, individualized problem-solving with social activities. This method started with Seymour Papert, credited for inventing the utilization of computers and robotics in the classroom. He was a firm believer in learning by doing, in which pupils were encouraged to explore and expand their knowledge through hands-on activities [11]. Today we have teaching kits for multiple purposes, some of them like "Designing Engineering Teaching Kits (ETKs) for Middle School Students" created by the Virginia state university, which are focused on promoting awareness and stimulating interest among middle school students. In the process, the students will learn about essential engineering functions like design, building, analysis, test, and measure, all while dealing with budget constraints, reliability issues, safety, and much more [12].

It is possible to create an experimental curriculum for high school students using the LEGO basic robotics kit, supplemented with extra sensors (either from the standard catalog or self-made) to gain adequate freedom in creating and fabricating robots. Since LEGO is a well-known toy, students will already know how to use it and its construction methods. That will allow a trial and error teaching approach that is much better and engaging, typically leading to students showing a deep interest in concepts discussed in school, creating a starting point for more in-depth robotic lectures [13].

In 2016, Florence R. Sullivan & John Heffernan published an article synthesizing the finding of their ten-year study that looks at the influence of robots in early childhood and lower levels of education, and they found out that [14]:

- The impact of robots on the development of children's talents can be divided into four categories: cognitive, conceptual, language, and social (collaborative) skills.

- Apart from the primary users (children), parents and instructors must also be on

board in order for these initiatives to succeed. In cases where there was a lack of parent support, educational robots would only be used in the classroom.

- When it comes to bringing robots into an application, design is frequently the last thing on the list. However, as studies have shown, design can influence robot perception and, as a result, how youngsters engage with it.

There is also teaching kits for training works future or already working in there fields and these can be any fields like renewable energy manufacturing. Since the end of fossil fuels is getting near and near, countries are starting to invest a lot in producing renewable energy and the manufacturing of equipment, whit the understanding that a leadership position will give a competitive advantage in negotiations. This investment has created a demand qualified workers that hasn't been obtained for lack of personal. More and more academic programs are being created to study renewable energy to combat this phenomenon. The importance of lab exercise is paramount to comprehending complex concepts and theories better.

Teaching kits are one of the tools that these facilities implement, but it has known that most of these facilities build their equipment for labs and projects, and only a few use commercial lab equipment and experiments kits for their teaching. Nevertheless, there are several experiment kits to help the students gain professional skills in problem identification, engineering, team management, working together, and of course, hands-on experience. However, the majority of experiments focus on the study of solar, wind, biofuel, hydrogen, and smart grids. Options for others like wave and tide energy are pretty limited. Among these projects, the standard features gained are [15]:

- Sophisticated control systems
- Pre-designed lab assignments
- Made according to the industrial standards

Moreover, a few of these programs are:

- Solar
  - EDIBON (http://www.edibon.com)
  - Lab-Volt (http://www.labvolt.com/)
  - Hampden (http://www.hampden.com)
- Wind
  - Quanser (http://www.quanser.com)
  - Hampden (http://www.hampden.com/)
  - EDIBON (http://www.edibon.com)
- Hydrogen
  - Shatz Energy Research Center (http://www.schatzlab.org/)
  - Heliocentris (http://www.heliocentris.com/)
  - EDIBON (http://www.edibon.com)

Chapter 3

Approach

As mentioned before, the project aims to create a simple AGV system in terms of the robot. With that in mind, the first part was to simulate the robot in Simtwo. This will allow the extraction of the information necessary to program the robot in different conditions and learn more about the robot. One of these is safety since robots are made of hard material, and the focus is weight transportation. But the contribution goes beyond that how to stop when to stop, cornering and others.

At the same time, this thesis gives an overview of Simtwo, even if this is not the aim of the thesis. However, it is imperative to simulate how a system of AGV would work and how it can be done so that the user can simulate different processes. Nevertheless, this information depends on the user's capability, which is a significant part that will allow a student in groups to improve their social skills (working together).

Another objective of the thesis is to create a robot model since is significant in any project, and it goes hand in hand whit simulation. The modeling process is abel to detect the technical difficulties in creating this robot to scale and for problems and difficulties that will need to be solved of course not all of them.

The modeling will allow the creation of a model that will serve as a teaching kit for additional students. Granting them a test bed for their projects.

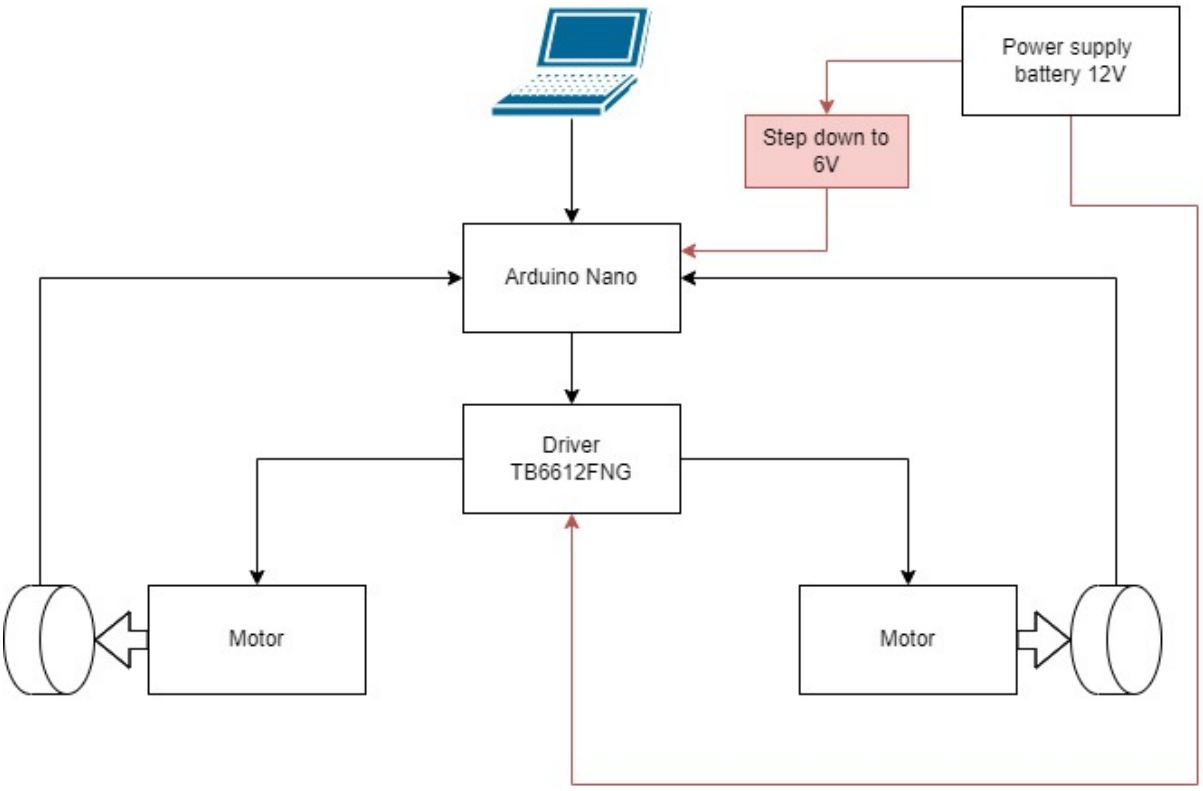


Figure 3.1: Block diagram overview

3.1 Architecture

The proposed architecture is going to be as simple as possible. It can be seen in the fig3.1 .

The Arduino nano will be the processor of the system receiving information from the computer that will work as an interface between the operator and the robot. Once's receiving the command from the PC, the Arduino will do the proper calculation 5. After the Arduino will send the signal to the driver to amplify it and send it to the motors. The Arduino will read the signals sent by the wheel encoder, calculate the error between the PC signal and the wheel signal, and correct the output, for feed back control purposes.

3.2 Equipment and software used

This project used multiple different parts and equipment's with the goal of obtaining our objectives

- SIMTWO
- SolidWORKS
- Arduino
- Servos
- Driver
- Battery

3.2.1 Software utilised

Simtwo

This is a realistic simulator with the main objective of simulating mobile robots that use wheels or legs, nevertheless it can simulate other types of robots from drones to industrial robots [16]. This tool can be used to design complete scenarios including complete environments and define objects characteristics indispensable for the realism of the simulation[17]. And this among other tings like the dynamic realism in the program makes it very good for the field of education considering that this field is inherent multi-disciplinary.

SolidWorks

One of the most used CAD programs with a little over 3 million cumulative user through the 4 quarter of 2015 it's excellent for designing parts for 3d printing [18].

3.2.2 Components Used

Arduino

Originally made for artists and designers Arduino have proven very good for DIY projects both for robotic enthusiast as well as for teacher and their students. In education, the introduction of arduino and similar devices can improve interest of building and designing things. Furthermore through continuous projects as well as challenges that can be found online the users can improve there critical thinking[19].

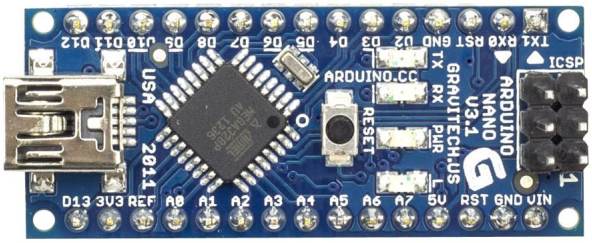


Figure 3.2: Arduino Nano

DC Motor

This project will require the use of two DC motors attached to a reduction gearbox, and it comes with an encoder for feed back control purposes.

Figure 3.3: DC Motor

Driver

The driver utilized in the project is toshiba TB6612FNG driver integrate circuit wich is a dual motor output that can chose between four modes Clock Wise, counter clock wise, short brake and stop mode. With a power supply max of 15 V and an current output average of 1.2 A and peak of 3.2 A.

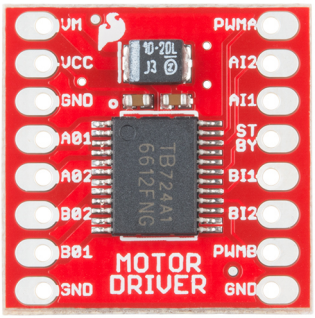


Figure 3.4: Driver

Batteries

In this case the battery used was Samsung INR18650 that is very small and light compared to other battery, since they are made of lithium polymer. this project will require at least 2 battery in series.

Figure 3.5: Battery

Chapter 4

Implementation

4.1 Simtwo

In Simtwo, we started by creating the track in HTML like in figure 4.1. This is constituted by eight blue lanes areas which are used to positioning the robot, the main avenue that's in pink, two roads that form a T that is in red used to load the robot and two parking spots in black that are used to store the robots after usage. Each of these has to be created through lines of code that will be inside a file that is named *track file*, and it's accomplished by following these steps:

1. create a new file
2. define the constants (note this can be skipped but for a better program organisation it is recommend)
3. declaration of the geometric that is going to be used, it can be "*<line>*" for lines, "*<circle>*" and others
4. stating the properties of the geometric shapes(*colour*, *position*, *size*, *tag value*/name , *repeat*-times to repeat and how)

The next step is the creation of a robot applying the same language before establishing the control code of the robot made in Pascal. The creation of the robot is done inside the

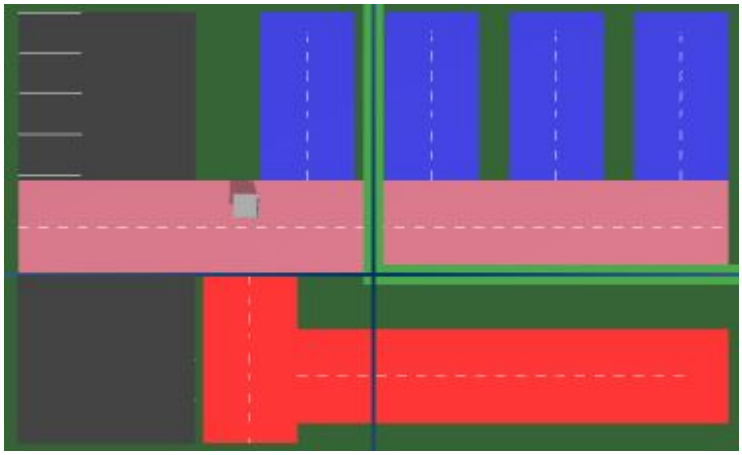


Figure 4.1: Road Map

program by using geometric solids *cuboid*, *cylinder*, *sphere*, some of this will be declared as servo motors or control center of the robots where the batteries and microcontrollers like Arduino would be located. For the motors, they don't show up visually, but they are inside the joints of the wheels, and the accomplishment of this is done by following the following steps:

1. The creation of a robot file
2. Establish the constants
3. Start inserting the solids= these will be the rigid parts of the robots that do not represent the logic parts and may not make a mechanical movement.
4. The shells= represent logical parts like Arduino, Raspberry pi . This is also where the programmer characterizes the slide pad, and in these cases, it will be put on the leg offering support to the robot.
5. Joints = the joints connect the solids and inside them resides the motors and here the characterization of how they are going to work, if they are active and are controlled and if they vary their speed or it constant.

In this scenario, the robot is constituted by a top part that will have a QR-code so it can be identified as a *shell* along with the backplate in green that supports the batteries

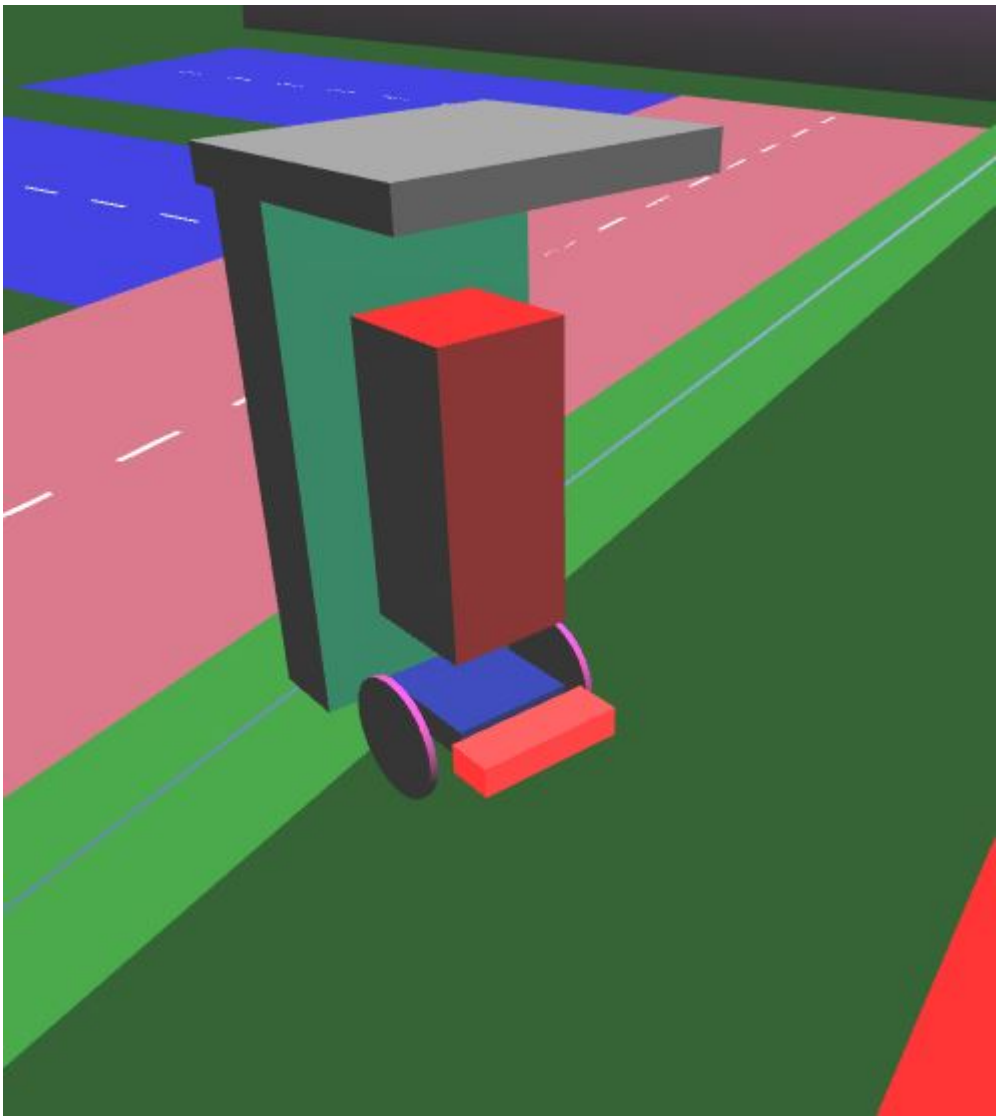


Figure 4.2: Robot

and microcontroller and as structural support. The motor position will be in the part of the red box, and its type is also a *shell*. The bottom plate will be a *solid*, and it would support the motors and gearbox in real life, where it is going to serve as the base that connects all the articulating thing pieces like the wheels. The wheels and the support will also be part of the *solids*, and these are connected with the base plate by *joints* in which only the support joint will not have an active motor.

The next step is creating solid objects, classified as *objects* that can act as walls or objects that are not affected by the robot as such, and they are used to replicate the location as much as possible these objects do not get affected by the physics engine. In other cases, a movable object is needed like chairs and other stuff it should be classified as *things* since they respond to the stimulus of the world, whether it is by the robots or other things, of course in this classifications it is essential to give them there properties like mass and surface(attrition and softness).

The creation of a scene is done by calling the files created for the robot where the programmer has to give it a name(*ID name*), a position(*pos*), an orientation(*rotdeg*) and the full name of the file that contains the robot, note a robot file can be used by multiple robots, as for the other three they only need their classification and file name as shown below:

$$< thingsfile =' things.xml'/ >$$

4.1.1 Program Simtwo

The first step is always to declare the variables (number of variables), although not all of them are necessary, it is good coding practice to use them, for example, the cases of the command "RCButtonpressed" , it makes it easier and faster to correct mistakes. The second part of the code developed was to retrieve the precise position of the robot for the operator and the central computer so one of them could plan routes of the AGV.

The first task was giving the robot (see fig4.2) a point and for it to arrive there by

following a straight line. For this, the robot must know the difference between where it is facing and where is the final point, so it can orientate itself accordingly to follow a straight line. To accomplish this, the computer retrieves the robot's current position and the point position, and it calculates the *theta*, using the equation 4.1.

$$\theta = \arctan \frac{Yp - Yr}{Xp - Xr} \tag{4.1}$$

This is the angle of the point, and the robot subsequently needs to calculate the difference between the angle of its orientation and the angle that was previously calculated. After that, it is capable of using the difference between angles that is stored in the variable named ThetaNC to multiply with the gain KT and then send it to the wheels. In the case of velocity, it needs to first calculate the distance between it and the point by using Pythagoras's theorem that, in this case, is demonstrated in the equation 4.2.

$$H = \sqrt{(Xp - Xr)^2 + (Yp - Yr)^2} \tag{4.2}$$

Moreover, the value that it gains from the operation is them multiplied again to get the speed(equation 4.3), and the parametres will be a gain of value 10 and the variable C that is being used an electronic switch (0 or 1) that is commanded by a button that simulates an emergency stop button.

$$V = H * 10 * C; \tag{4.3}$$

In order to preserve the lifespan of the motor, it's decided that if the distance is big (over 0.3m), the value of the variable V is going to be 5.

As for the two speed components (linear speed V and the rotational speed W) are combined it's possible to see them at equations 4.4 (velocity for left wheel) and 4.5 (velocity for right wheel).

$$Ve = V + W \tag{4.4}$$

$$Vd = V - W \tag{4.5}$$

Two methods were created so the user can define the location of the point, the first one is to use *sheets* window to insert the points manually and changes them when it reaches there, and the second is to write the points in the program in states and having the program call them one by one, these would consist of the X and Y components of the points like demonstrated below.

$$Xp = 0.716$$

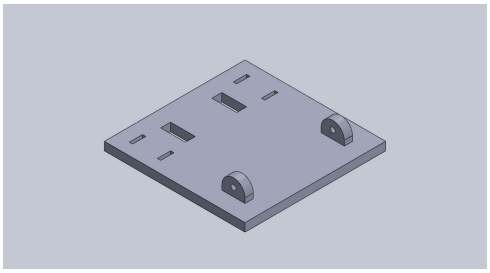
$$Yp = 0.1425$$

Furthermore, using the command *SetRCValue* that is used to display a variable in the *sheets* window it is possible for the user to see every variable and parameter in *sheet* window allowing them a greater understanding of what the program is and how to improve it, and bellow are some of the variables that are presented to the user are.

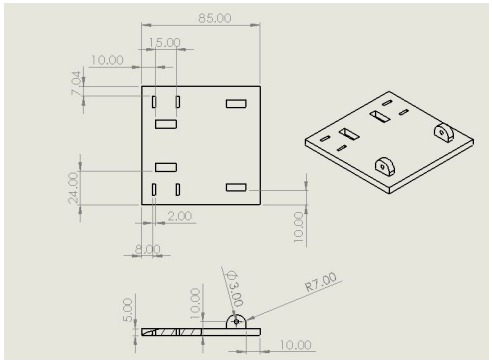
- H
- Theta
- ThetaNC
- Xp
- Yp
- Xr
- Yr
- Tr (Theta of the robot)
- V
- W

4.2 Hardware

The prototype of the robot was build using 3D printing techniques for the structure and an arduino for control. They started by designing the bases of the robot as pictured in Fig4.3.



(a) 3D Model



(b) Measures

Figure 4.3: Base plate1

The fig4.3b presents the model sizes, and these are 85x85x5 mm, allowing the placement of the microcontroller and the batteries in places 1 & 2 (see fig4.4). This would facilitate any workaround in case something went wrong. The CAD files also present holes that will allow anchoring the motors and the battery system with zip ties.

After some consideration, it was decided to scrap the zip ties for the batteries and use a battery holder of another project that was scrapped and changed the wheels' placement. The change in the wheel axis location was to facilitate the programming of the robot, more precisely speaking, the angular velocity of the robot and because the center axis location would make the robot topple over we add poles with holes for sicked pads (see fig4.5a) to stabilise. The sicked pads can be made of various materials, but in the end, we used polystyrene foam commonly known as "*styrofoam*".

The wheels were made with a slot down the middle so it would a accommodate a rubber strip but due to problems in printing it was necessary to add a small chamfer so the printer could print without problems (see figure4.6).

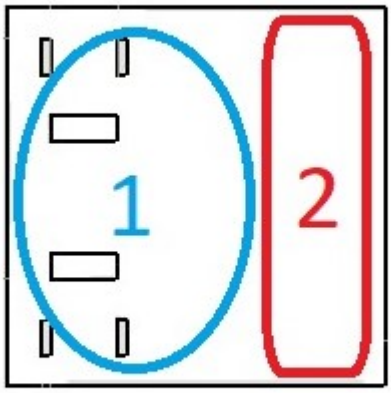
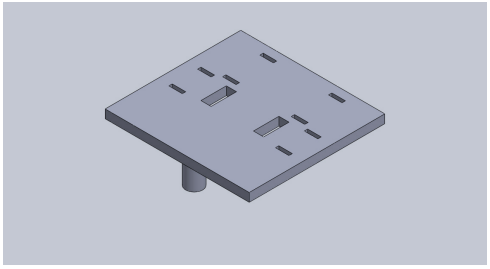
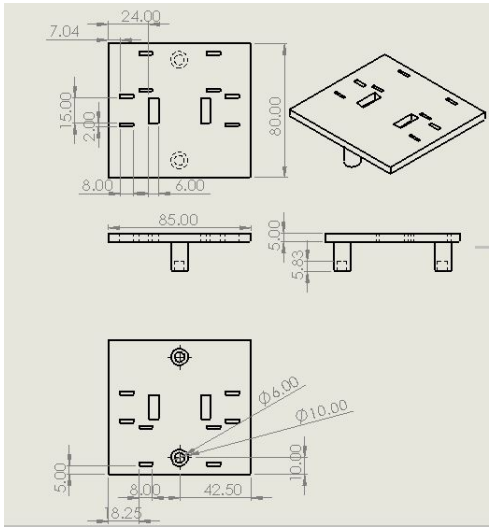


Figure 4.4: Base plate zones



(a) 3D model



(b) Measures

Figure 4.5: Base plate2

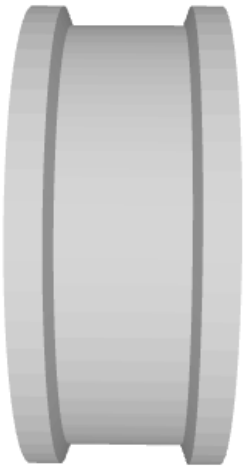


Figure 4.6: Graphic representation of the new wheel

Input		Output
0	1	R CCW
1	0	R CW
0	0	Stop

Table 4.1: Driver control tabel

4.2.1 Electrical connection

Figure 4.7 presents the electrical connections that were made and are the main ones are: the four wires that control the motor directions and that are in white departing from the Arduino to the driver, the two in yellow that transmit the PWM signals from the Arduino to the driver and four more wires from the motor to the Arduino.

This set of wires is for transmitting signals from the components to the Arduino. The white is used for the direction of the rotation requires two per motor resulting in the four wires telling the TB6612FNG what the direction of rotation should be *Clock Wise(CW)* or *Counter Clock Wise(CCW)*, and due to this circumstance, it allows us to stop the motor as seen in the table4.1.

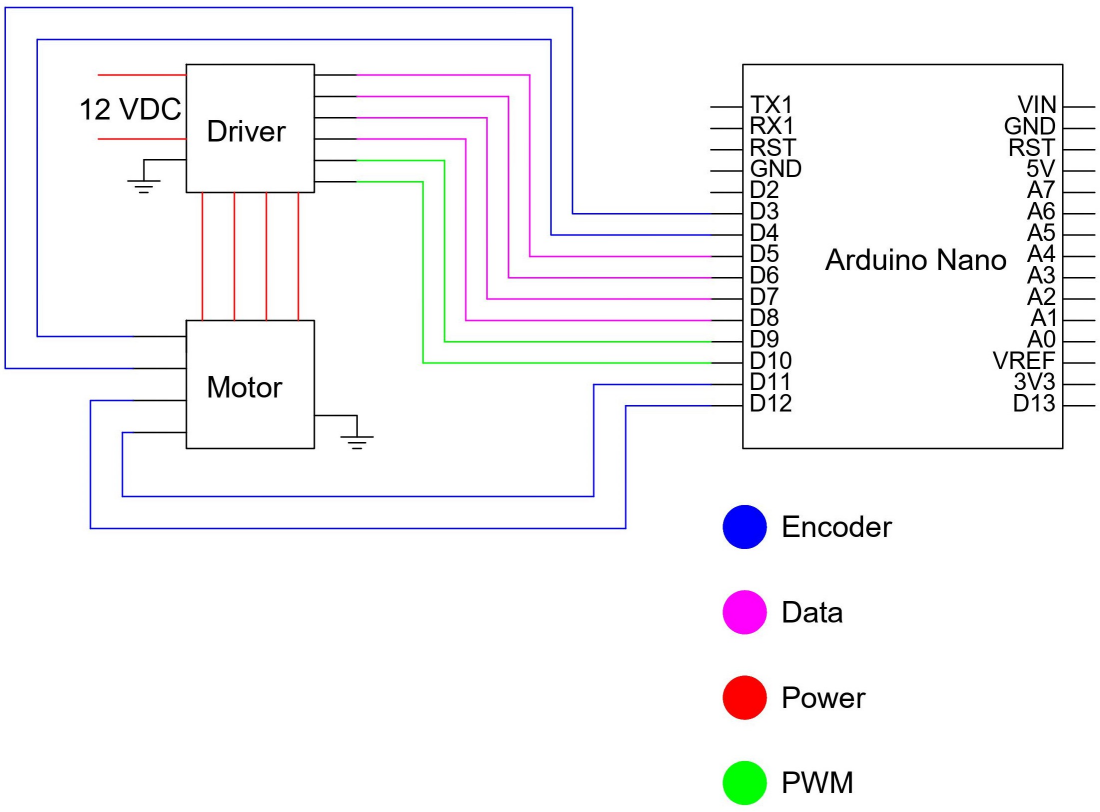


Figure 4.7: Electrical diagram

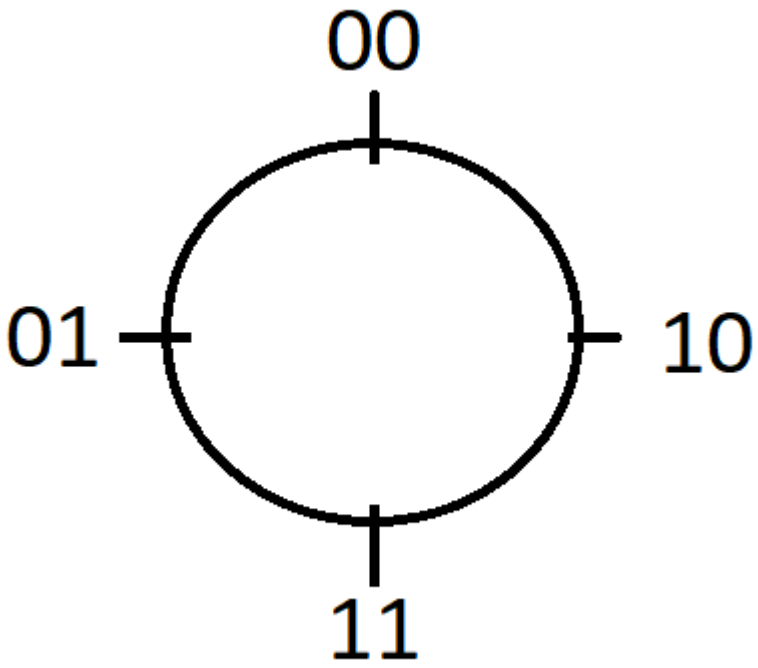


Figure 4.8: Motor encoder

The TB6612FNG is supplied by the Arduino using one of its 5V pins along with the encoder of the motors. It also use a 12 V power supply to amplify the PWM signal. The encoder will change the signal at every quadrant(see fig4.8), and because of that, it is possible to determine the velocity and direction of rotation with the Arduino.

4.3 Arduino

This section will present the architecture of the program and then in detail the main parts of the program and how they work.

4.3.1 Architecture

The result was the following architecture of the program as seen in fig4.9.

It is possible to see that the program will intake a V and a W that are the linear and angular velocity, after it will receive the velocity of the wheels from the sensor then using

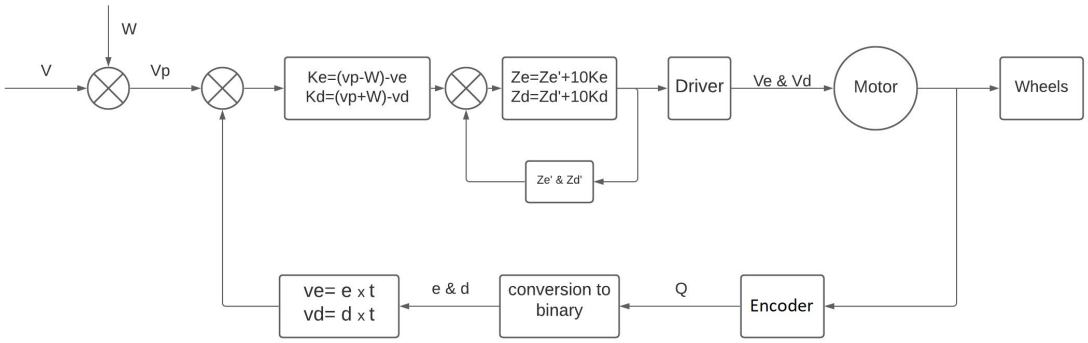


Figure 4.9: Program architecture

the value of the past Z, the program will correct the value of Z before sending it to the driver. The signal will be amplified to the desired peak value and the correct duty cycle given by the Arduino. The motor has coupled sensors that will produce a digital signal sent to the Arduino to decode and calculate the velocity and distance made by the motor.

4.3.2 Program

Because the available Arduino did not have a wireless connection, it forced the robot to be controlled via the USB. So to accomplish this endeavor, a plan was formulated: first, the creation of 6-byte code to send the signal as seen in 4.10, second creating a program to receive and decipher the contents of the message, and last but not least change/ modify the operation of the robot according to the message. In this code, the first byte would signal the type of movement F(forward), B(reversing), and S(stop) the second indicates which wheel is to move A(all), E(left), and D(right) the next3 are used to determine the velocity of the wheel controlling the motors and the last byte is used to send an end of line character to the Arduino. For the Arduino to receive and execute the code sent, a piece

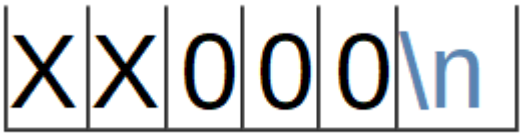


Figure 4.10: Code

of code started by *while loop* design to investigate if the computer has sent any data over the serial bus. After that, a 6 bytes array is created and start copying the contents of the serial receive buffer to that array in sequential order until the next byte is of the line end character or it passes the maximum value of 6 bytes of our message. After transferring the last character, the Arduino uses *memcopy* to copy the contents of the array to the respective variables or arrays and closes the *while loop* and use contents of the message to control the wheels so the robot can move in the desired way. The arrays that are created are:

- Direction = array that stores the type of movement
- Wheel = array that stores the wheel(s) chosen
- Speed/Velocity = array that stores the velocity in PWM format

For the first two arrays, their content is checked by *if* command to interpret the command, and the pair have three options that can be taken by the user or program as explained in the beginning. For the information stored in *"velocity"* to be used, it needs to be converted from an array to integer, and the Arduino accomplishes this by using the command *atoi*, so the command *analogWrite* can use it once it can not accept an array.

After some testing, a better way of cornering was suggested that required receiving an angular velocity and adding to a wheel, and subtracting to another. This would simplify the program operation for the operator.

encodeposition	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ERRO	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0
Passo	0	1	-1	0	-1	0	0	1	1	0	0	-1	0	-1	1	0

Table 4.2: Encoded signal

Feedback loop

The feedback loop starts like any other by having to read the signals sent by the motor encoder (these will be Variables QE 2 & 3 and QD 2&3) and compares them with the previous signals(QE 0 & 1 and QD 0 &1) and with the help of a look-up table this was found to be the best way since comparing them using *if* statements would not be fast enough. To be compared, the signals are transformed into a decimal number using the equation below, and this will generate an "unique" number for every combination of the signals and using the table 4.2.

$$e = 8 * QE[0] + 4 * QE[1] + 2 * QE[2] + QE[3] \tag{4.6}$$

This table shows every possible result that the encoder can output have an encode comparison that will take the variable e for left and d for right. With this number, the program will take the value P at that position and add or subtract to the variable *Qd(e)*, and it will serve as a record of how many quadrants have the axis passed and in which direction (4.7). Also, the error will be recorded as well in the variable E this will happen when the code generates a value that signifies that the axis jumped a quadrant(4.8). Before this piece of code runs again, the variables representing the past states will take the variables of the current state. After that, the code will repeat this piece of code is not in the main *loop* function that represents the main body of the Arduino codes, to repeat at every 100 microseconds using *Timer1.initialize*.

$$Qd = Qd + P[d] * -1 \tag{4.7}$$

$$E = E + Erro[e] + Erro[d] \tag{4.8}$$

The velocity is obtained in 5 seconds increment to achieve that the command *delay* is used, and it sets the time for the velocity equation. The distance is obtained by subtracting the current number of states passed by the previous and multiplying by the ratio R as seen in the equation 4.9

$$De = (qe1 - qe) * R \tag{4.9}$$

With the distance and time (5s), the Arduino can calculate the velocity using $vd = Dd/5$. The velocity will allow it to determine the error between the user's velocity (vp, intended speed) and the wheels, and the result will be multiplied by the gain G to increase the velocity until the desired level (see equation4.11).

$$kd = (vp + W) - vd \tag{4.10}$$

$$Zd = Zd + G * Kd \tag{4.11}$$

Chapter 5

Results

5.1 Simtwo

What was accomplished in Simtwo was the simulation of multiple robots three at max. The first test that is made is for the robot to move in a straight line after giving it that represents its destination, and this act is being done in *sheet window* end position is monitored by using the *chart window*. The test point selected was (-1;-1) chosen randomly since the only criteria was not to be directly in front.

The graph displayed in the Figure5.1 presents the angle of the robot in radians. What was understood from this graph was that the robot does a huge course correction in the beginning, and after about two seconds, it only does a small course correction because it has to face the point first them starts to move. The confirmation that the robot does the course correction before it moves can be seen in the figure5.2 in which it shows a solid start to the incline, meaning that it when straight after the start in the correct direction. The particular reason for the gradual changes in incline that occur at the end is due to what was implemented in the code. After the robot gets close to the point, its velocity decreases, this is done to increase the safety of the robot and to prevent it from overshoot and hit something. Another point that demonstrates the movement is the graph in figure5.3 in the beginning, the axis rotates in the opposite direction. This

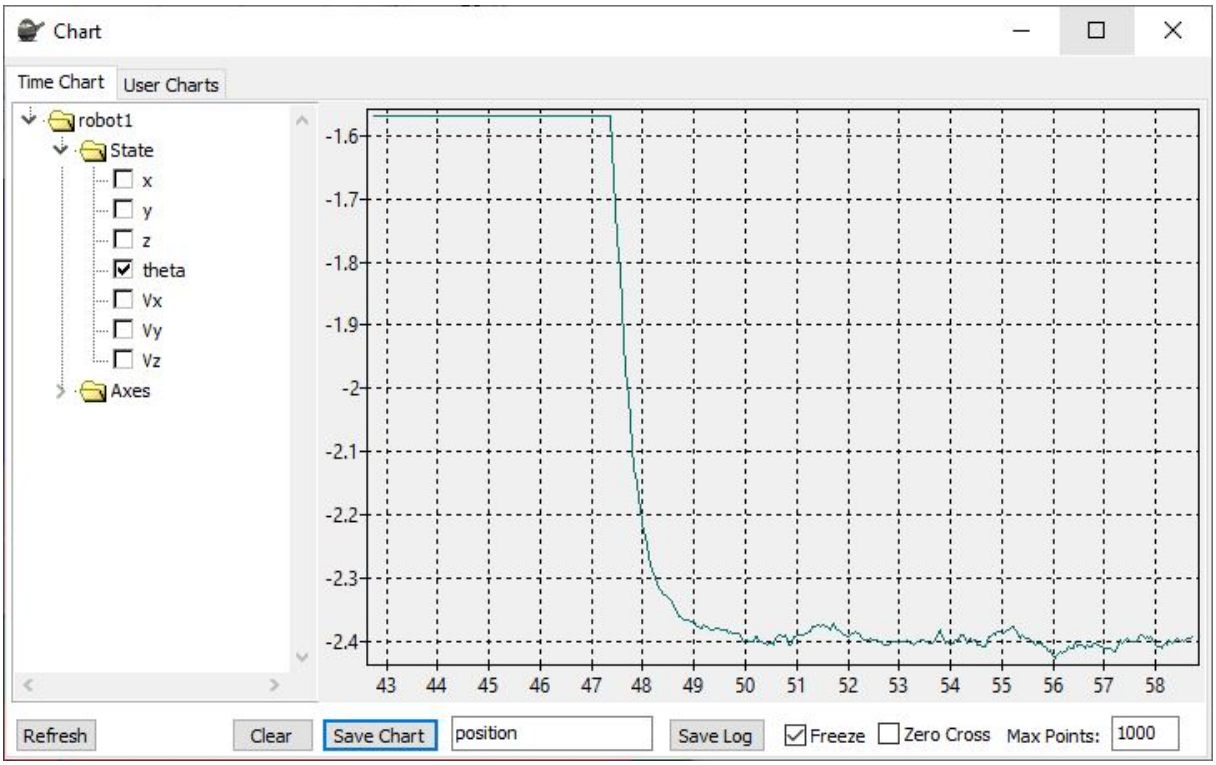


Figure 5.1: Theta of the robot

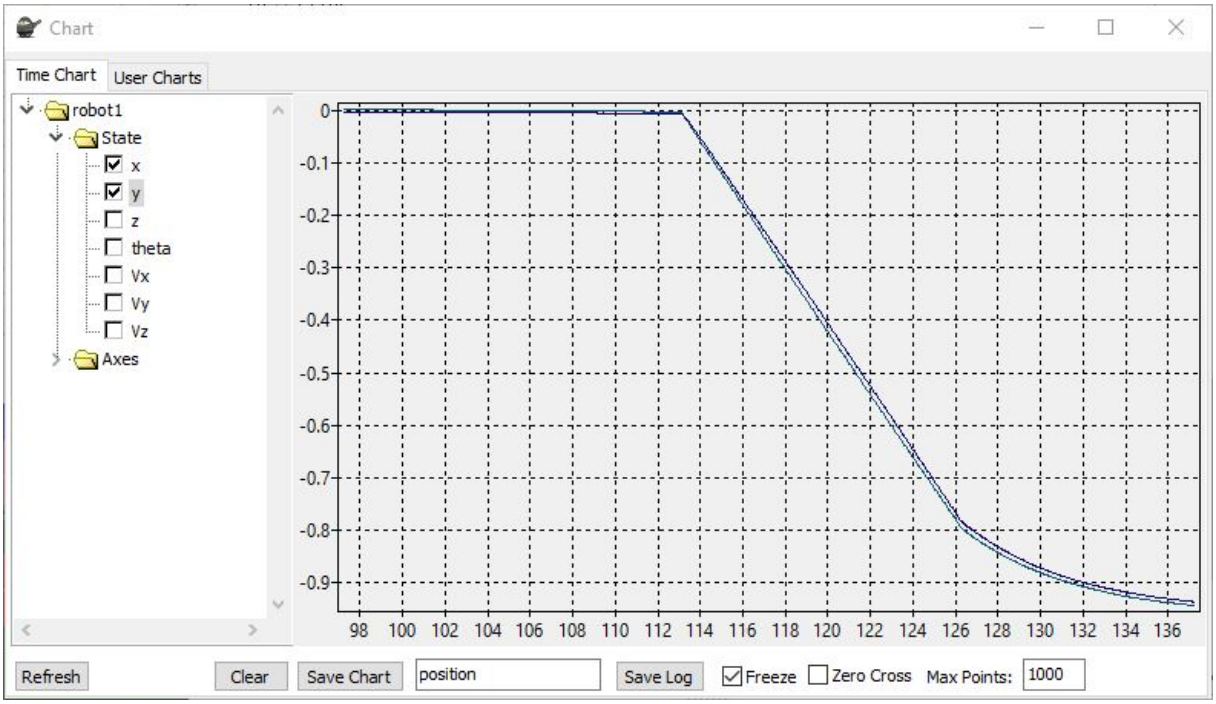


Figure 5.2: X & Y of the robot

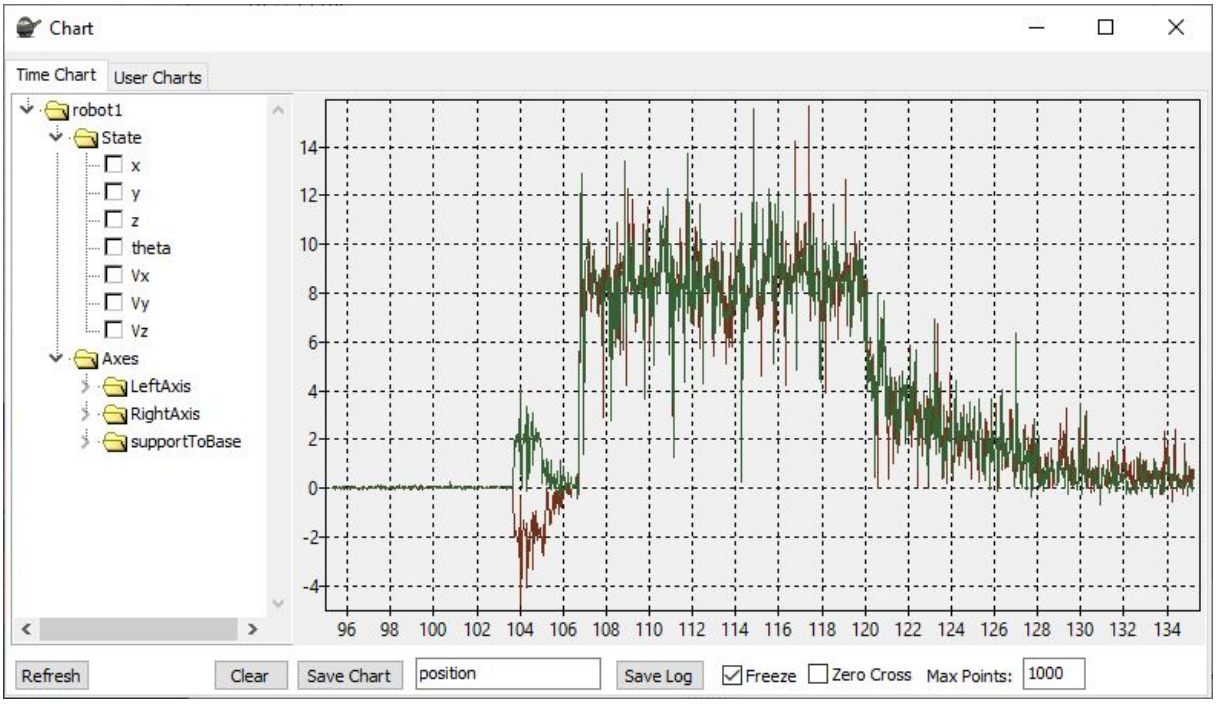


Figure 5.3: V of the axis

is to rotate the robot above its center point while doing the robot does not move in the X and Y direction. After that occurrence, there is a spike in speed in both motors, but this varies a lot due the constant course correction. Subsequent to this, the circumstance that makes the average velocity decreases it's simply when it gets close to the point due to the program.

For get thing around a corner the robot have two options: pivot on top of the center point or go around a point. The graph in the figure5.4 displays that the difference in the axis velocity that lasts for quite some time, and that's because the robot was going in circle's and due to this circumstance, the velocity of both axes are positive and only differ in value so that the robot turns. In the latter part of the graph on figure5.4 allow us to see the time that it takes to make a 180^{o} around a corner. In the same scenario, the graph in figure5.5 displays the angle variation, and it shows a sawtooth wave and the particular reason for this circumstance is that the angle changes signal when it reaches 180^{o} or π in radians proving that sawtooth wave wich means that the robot is going in circles. At the end sharp 180^{o} turn is displayed, meaning that the incline of the wave has to do with

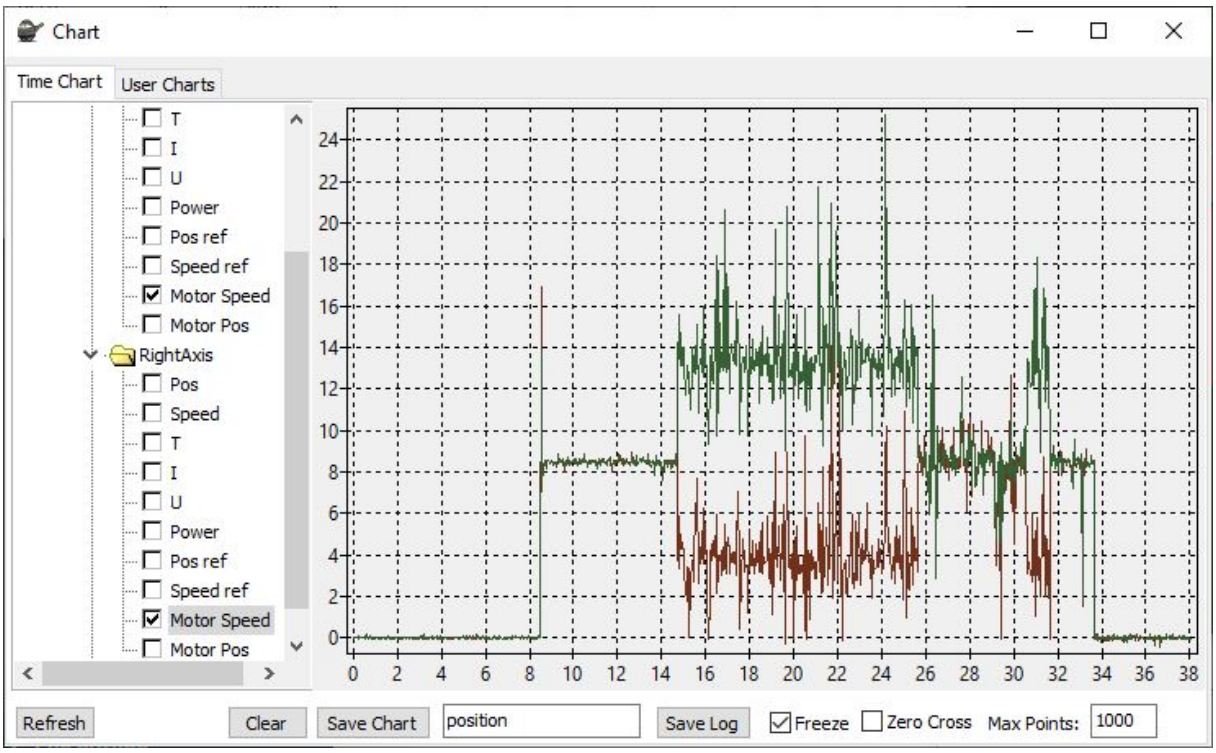


Figure 5.4: V of the axis corners

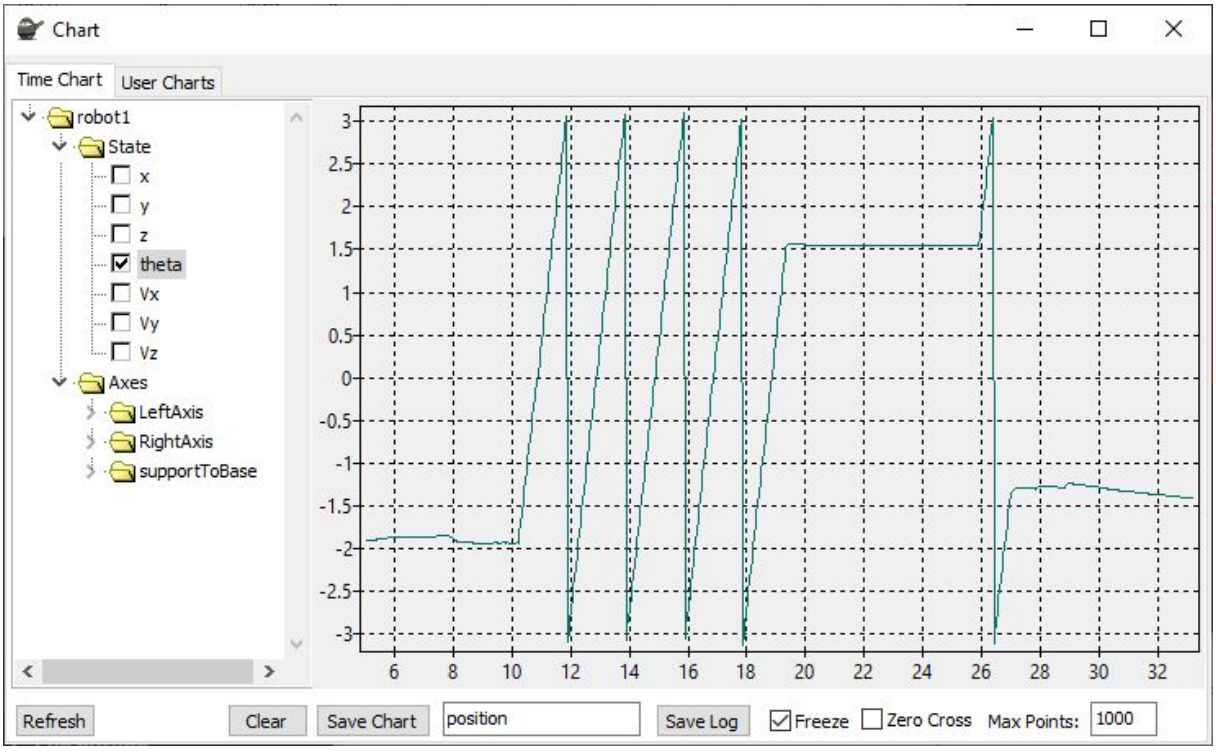


Figure 5.5: Theta around a corner

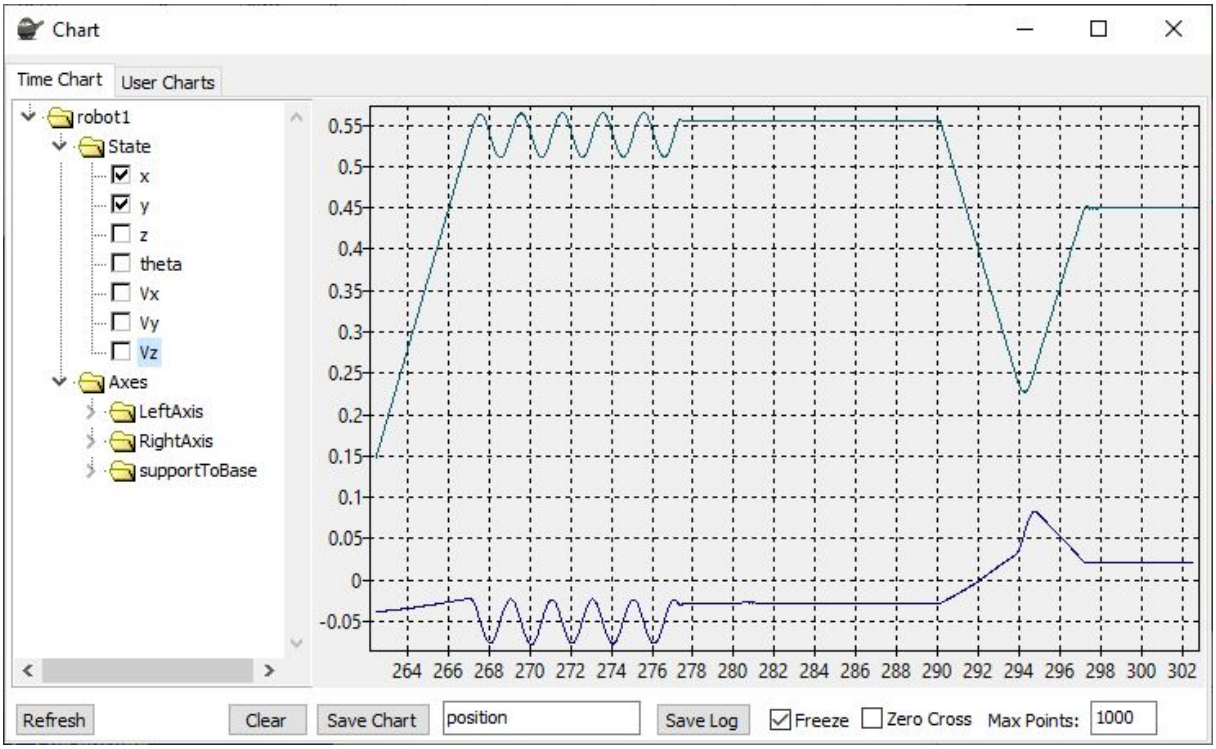


Figure 5.6: X&Y on a curve

how fast the robot does the corner. The graph in figure5.6 shows the variation that was created by the robot, and it took the form of a sinusoidal wave, and the particular reason for this was that it was going circles, and further along with the graph, the user will be encountering a sharp wave that signifies a sharp turn.

The final scenario had the robot make a lap on the main avenue to see what will be learnt and the results that will be gather in the graph in the figure5.7. What the user gathered from this is that the robot can do the laps without any significant problems and that had been shown by the sinusoidal wave (see Table5.1), now the fact remains that the peaks of the wave are a bit deformed, and the particular reason of this facts is two-fold:

- the change in speed when it gets close to the point
- the change in the code that had allowed the robot to move to the next point before reaching this one

The reason for the first one is safety because if the robot just slams on the brakes, it will

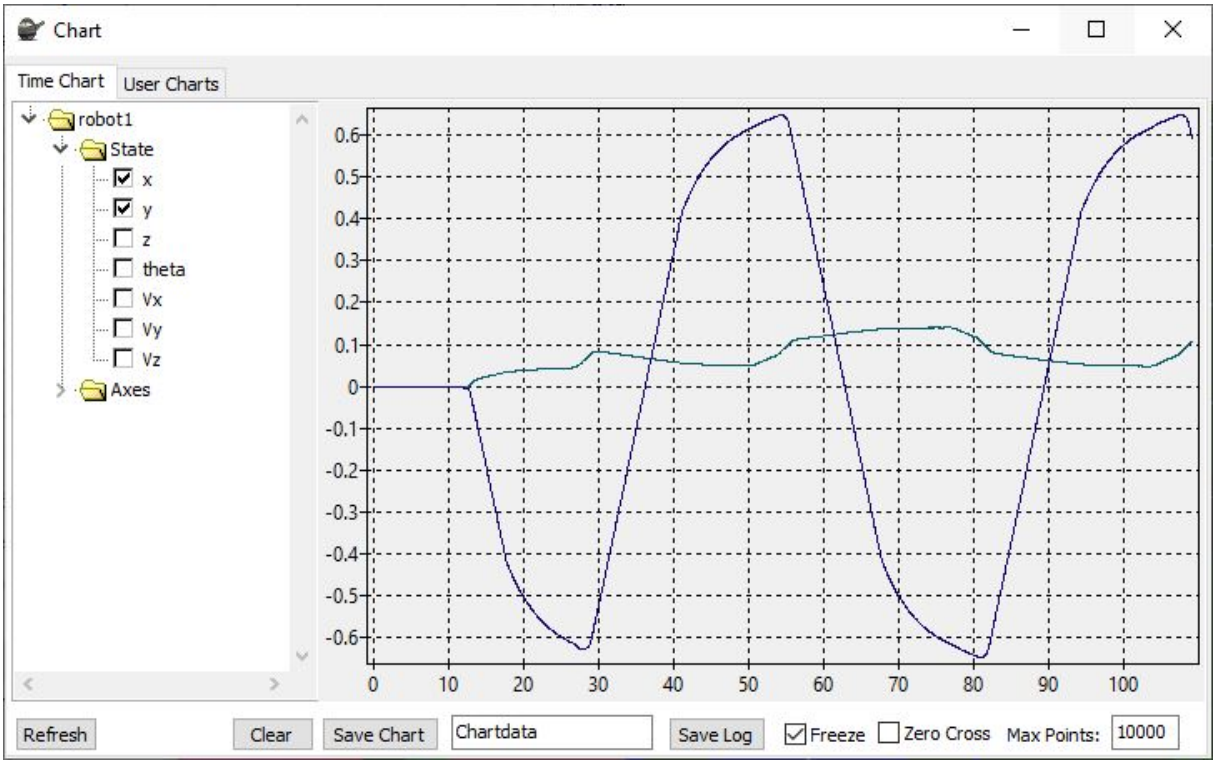


Figure 5.7: X&Y of the LAP

	P1	P2	P3	P4
x	-0,716	0,716	0,716	-0,716
Y	0,0475	0,0475	0,1425	0,1425

Table 5.1: Table of positions

create a lot of stress to the system, so they had chosen to preserve the system by coming to a slow stop. The second one allows the robot to move more freely before reaching the final destination.

The reason for these positions is because they represent the four corners of the main avenue, and after changing the Y value for *P1* and *P2* to -0.5475 the median of the Y value stabilized as can be confirmed in the figure5.8. The solution to this is to decrease the distance at which the robot starts to move to the next point.

In the last consists in having two robots, do a lap and see if they can do this without any failures. The results were the following graph in figures 5.9 and 5.10.

The graph in figure5.9 shows that the robots can move freely, and they were able to

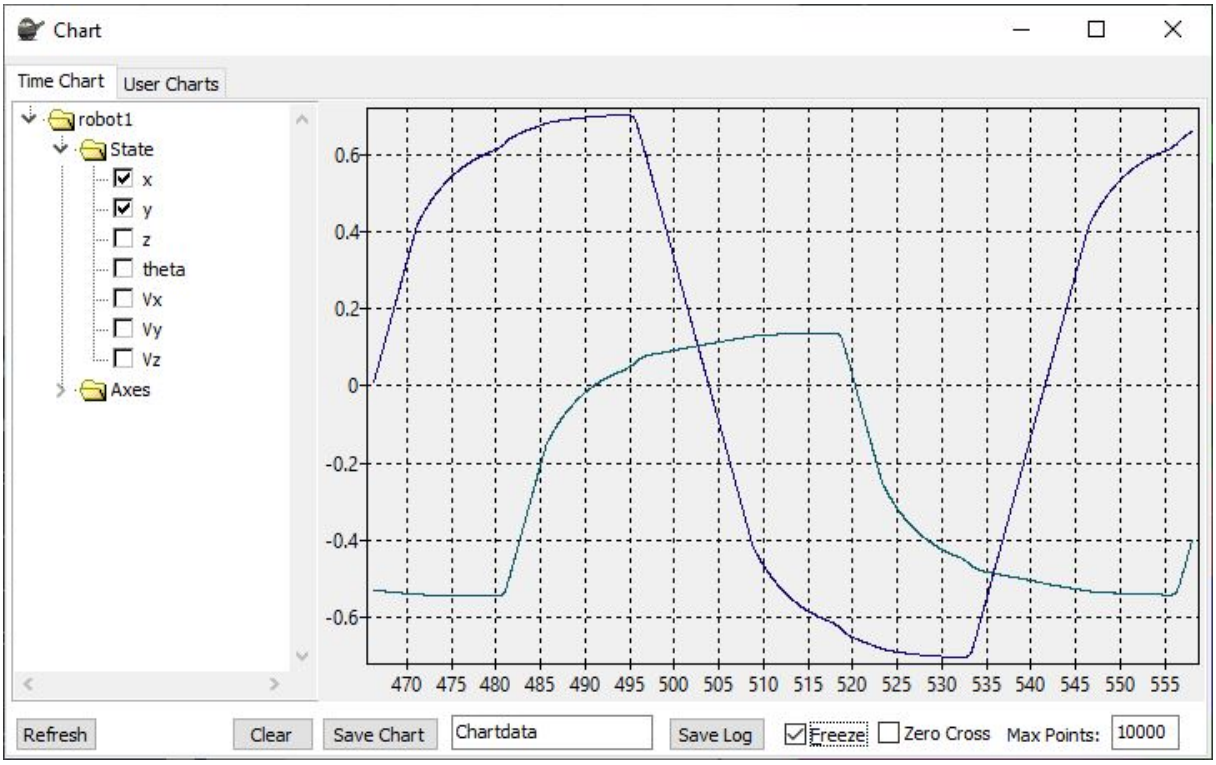


Figure 5.8: Lap after Y change

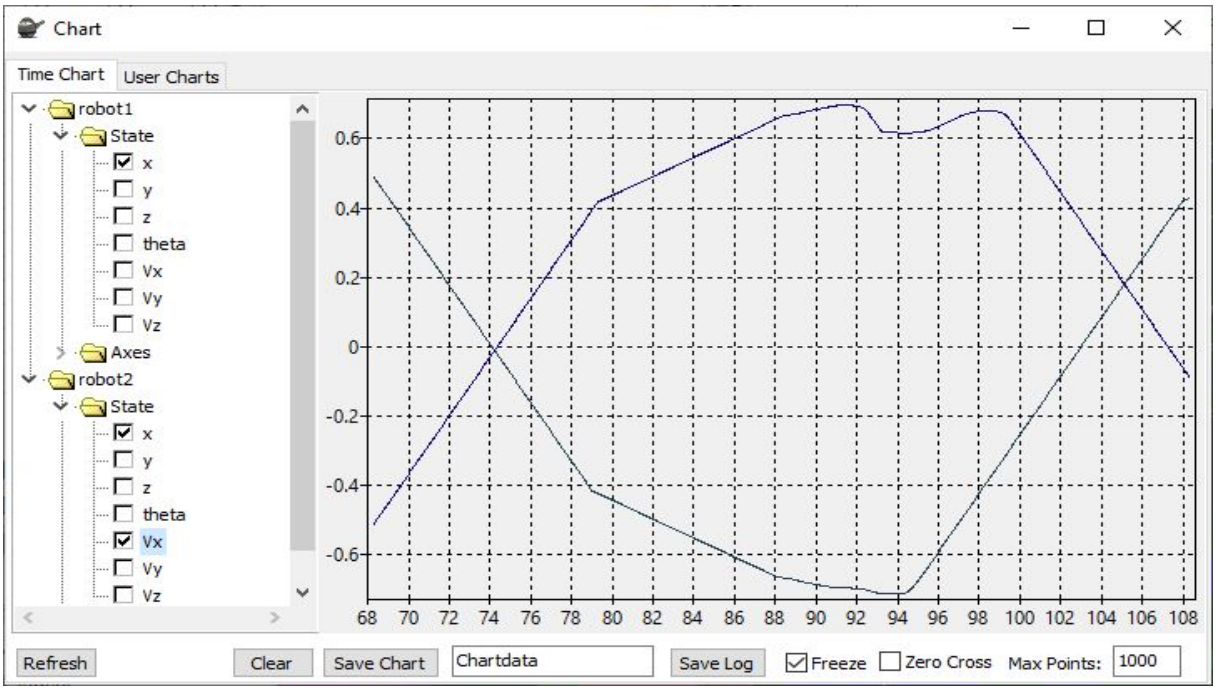


Figure 5.9: The x Values of bots 1&2

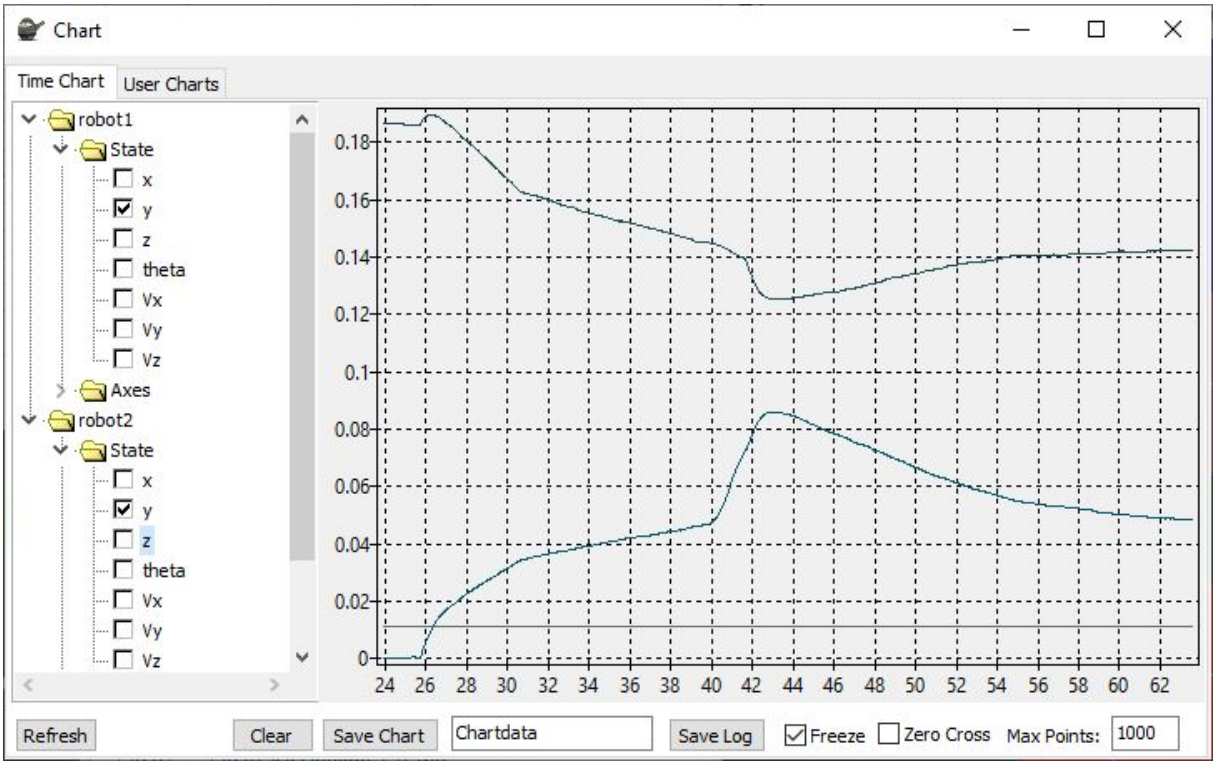


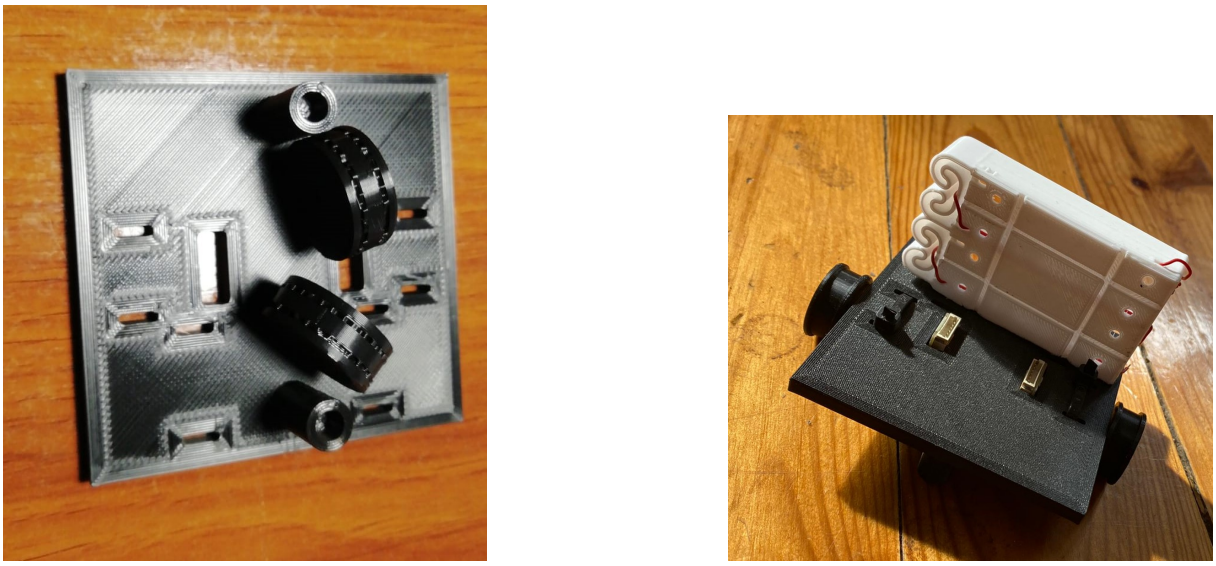
Figure 5.10: The y Values of bots 1&2

do the whole lap without any problems, since they paced each other two times to do a complete lap and had no collisions. Figure5.10 demonstrates that the robots starting point is not in line with the first points. After reaching the respective points, the robots redirect themselves resulting in the respective lines in the graph getting close, and after that, they continue in a straight line until the next point. This type of simulation is excellent in teaching because a good program for this end should handle two independent vehicles, and this is great for simulating a lot more systems enabling other collaborative robots and swarm technology. In the case of teaching kits, the scenario that we present is excellent due to the particular challenges that are predicting what the robot will do when reaching that point and if it has a chance of colliding whit the other robot. As we shown earlier is that even smaller details like the distance between points and when the robot should start the next task can cause deviations in its pattern.

5.2 Hardware

The hardware was started by printing the CAD files that were designed along with the wheels as seen in the figure5.11a it was all 3D printed after being designed in SolidWorks. Although it did not require to create support in base plate it was still necessary for the first wheel (figure5.12b) draft this was the main reason to redo the wheels and create a chamfer. The 3d printer did not leave support in the wheels since this proves extremely difficult without damaging the wheels. The new wheels (figure5.12a) required a reduction in the depth of the valley in the middle, and the figure shows the results of the changes were great for the wheels since it wasn't needed support during printing.

The assembling process required the use of a hot glue gun to attach the part that will hold the battery, and the motor was fixed using zip ties, and the final product can be seen in Figure5.11b.



(a) disassembled (b) assembled

Figure 5.11: Robot

(a) new

(b) old

Figure 5.12: Wheels

5.2.1 Test conducted

The test was done with the plot mechanic of the arduino as it allow the user to see the variation in the variables along the time. The figure5.13 presents a very straight graph. This is due to the 5-second delay used in the Arduino code. In this scenario, the Arduino is given a certain speed to match, and it goes to that speed, and then it is given a speed to slow down a bit. This is just a check to see the codified part is working correctly or not.

The test for cornering was started again by giving a set speed to the Arduino, then giving an angular velocity W and seeing how the exit would behave. The result of that is shown in the graph of Figure5.14, demonstrates that when given a W the signal sent to each will diverge, making that the rpm of the wheels be different that in turn turns the robot. But unlike the simulation, the robot can not turn on its center of mass quickly mainly because the Arduino does not produce a negative voltage without an external circuit and due to the driver selected not being compatible whit this since the pin that

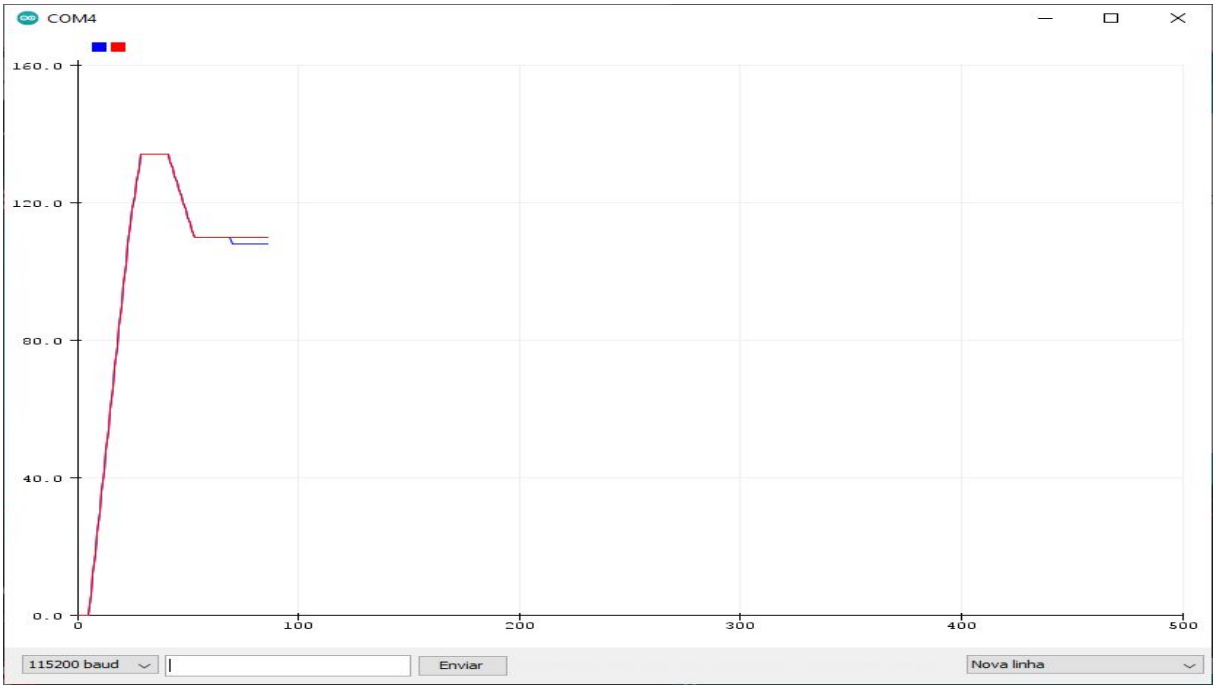


Figure 5.13: Basic control of velocity

receives the PWM signal is rated from 0 to 6 Volts, and to be able to spin on the spot the wheels need to be turning in different directions.

So in our case for the for our robot to change the heading it has to be done by being already in motion or by telling the arduino to change the output of certain pin so the driver can rotate the wheels in a different direction.

Another test that was made was to give the robot W bigger than it can use, see graph in figure5.15. This graph shows that the positive signal was limited by us to 170 out of the 255 of the Arduino but not the negative part. This produced a deficient negative value, but as said before, the Arduino does not produce a negative voltage, and the PWM accepts only values from 0 to 255. So whats happens when you *Write* in the PWM more than it can receive it goes around. For example, if the user writes 300, it would be analyzed and would output a PWM of like if the user wrote 45. The same thing happens if the user writes -10, it will look like it was 245.

Still, after getting this event, it was decided to limit W to a quarter of the velocity so that this would not happen due to the rapid cycle of accelerating and decelerating that

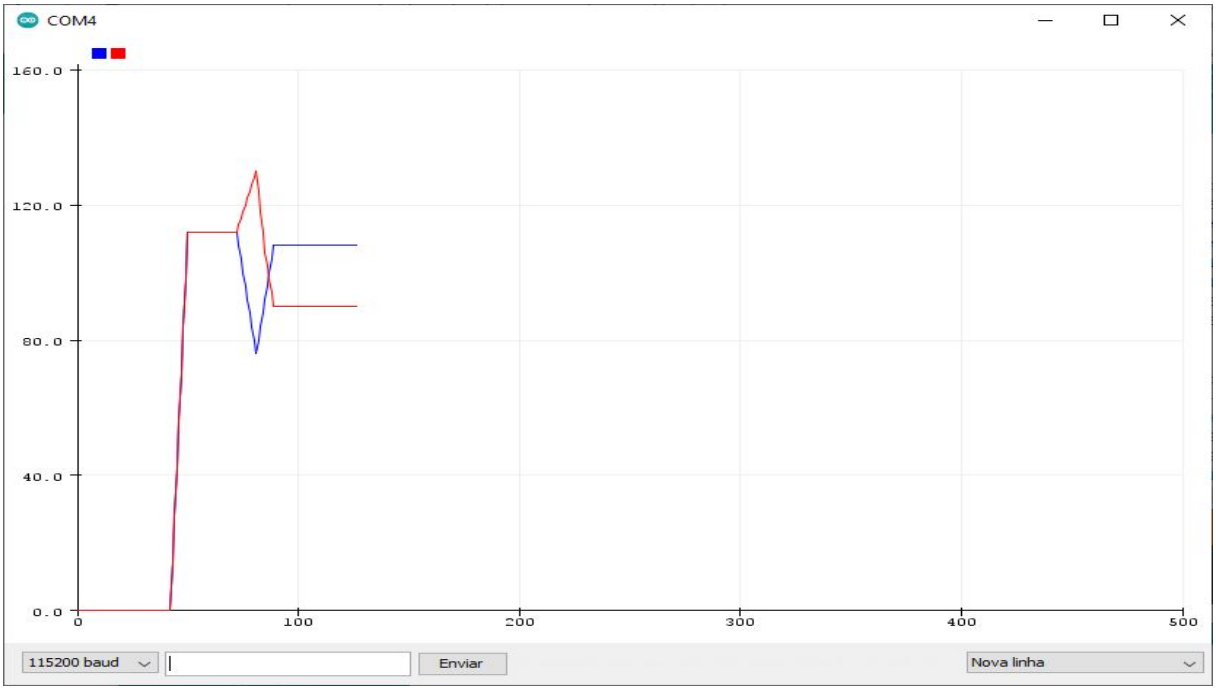


Figure 5.14: Doing curves in arduino

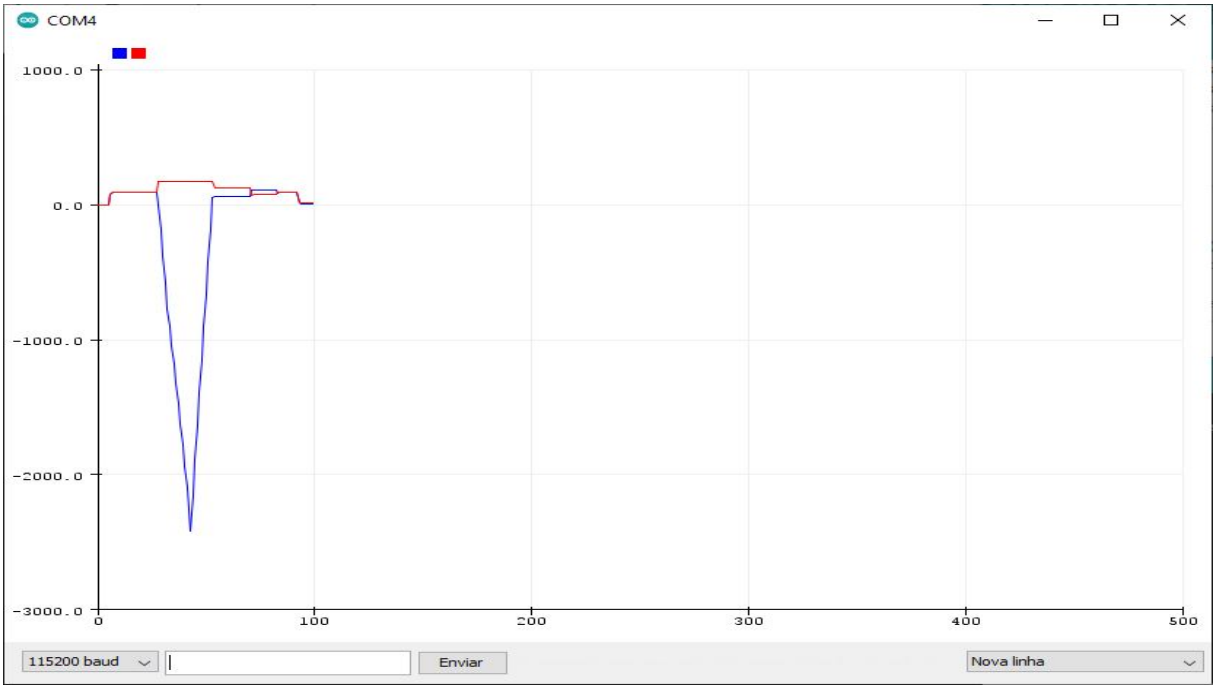


Figure 5.15: Controlling the robot

the motor was going through. After the output got to an acceptable value, the robot received the signal to turn in the other direction, and after that, the robot was stoped.

Chapter 6

Conclusion

The project was aimed at creating a model of an AGV that is cheap to produce and has a few sensors. This was achieved, and it also works as a testbed for the incoming students.

What was determined from Simtwo was that this type of AGV will depend a lot on external server for computing power, whether for orientation or to move around, and for these cases, it will have to have safety features for it to stop immediately to protect itself, the payload, and what is around them. As a result of unforeseen situations like power outages, breakdowns, and other occurrences.

This thesis shows that 3D printing in conjunction with Simtwo can contribute valuable knowledge to the user and create excellent teaching kits. Simtwo is an excellent platform that simulates an abundant number of systems that vary a lot between themselves. Moreover, combining Simtwo whit 3D printing allows the students to test their ideas, perfect them, and see if they can be manufactured or assembled. Although 3D printing can help in construction, it still has its problems, but it is becoming more and more widespread, and this means that the students should start practicing whit it all when they have the chance.

A project idea to prepare students of mechanical engineering and electronic engineering, is to from groups of two or three to create a two-year project in the university. They would learn a lot. The mechanical student, for example, would focus on a new manufacturing method by determining the material resistance of the parts made and how to

improve it. In comparison, the electronics student will focus on gaining experience in an expanding field both in terms of programming and designing a single robot or integrating it with a broader system.

6.1 Future Works

Future work on the AGV system should focus on the two most important objectives first:

1. the camera system

2. the path finding

The first one will be how to set up a multiple camera system from the ceiling and that looks down at the hole space to interpret what is happening. This will have two main problems that are: integrating the images of the multiple cameras into one and making a software that can take away the distortion that occurs because no camera will be exactly squared with the robot most of the time, and that is not taking in consideration the other stuff that belongs to the environment. This all will have to be done via artificial vision, and also the detection of the robot and other objects/people, the latter is more difficult since the robots can be painted in a way to be easily detectable will they can be any color. Nevertheless there is still the problem of converting a 3D image into a 2D image, so the height of the objects will have to be extrapolated via known markers or objects and the position of the unknown object, and some algebra.

The second one has to do with the system's artificial intelligence (AI) since all of the robots will take all the information from the servers. It will have to plan a path, select the necessary actions for every robot, and respond to a changing environment. The AI is a crucial system since it connects the camera system whit the robot, involving a lot of algebraic equations.

There are still, of course, smaller systems that are still essential, like the communication between all of these modules, the firewall, how can an of site technicians communicate

with the systems, or how an operator take control of all the robots or just one in case he has the need to.

The model has immense potential and can be used to train kids and college students in robotics to learn the difference between programming in a real robot. The model will be of great use to the new students that come to IPB, and it also can be used whit the high schools in the area, creating courses for students to learn about STEM. Moreover, it will help increase the student's critical thinking, which provides them with the needed skills to solve problems and issues in the environment regardless of their nature. Exciting projects to make are balancing the robot on two wheels and having it move around, needing only an electronic gyroscope. This piece of material makes an excellent robot for a teaching kit.

Moreover, there still is the CAD files that can be made into other robots whit different microprocessors like *esp32* and make the model wireless there also the possibility of controlling the robot using radio car commands or installing optical sensors and transforming it into a prototype of a vacuum bot. An excellent future project would be to make a list of materials needed to make a commercial kit.

Bibliography

[1] agvnetworkcom. "Agv cost estimation. how much does an automated guided vehicle cost?" (2015), [Online]. Available: https://www.agvnetwork.com/agv-cost-estimation-how-much-does-an-automated-guided-vehicle-cost (visited on 05/31/2021).

[2] G. Ullrich, "The history of automated guided vehicle systems," in *Automated Guided Vehicle Systems*, Springer, 2015, pp. 1–14.

[3] R. A. Søraa and M. E. Fostervold, "Social domestication of service robots: The secret lives of automated guided vehicles (agvs) at a norwegian hospital," *International Journal of Human-Computer Studies*, vol. 152, p. 102 627, 2021, ISSN: 1071-5819. DOI: https://doi.org/10.1016/j.ijhcs.2021.102627. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1071581921000458.

[4] J. Mehami, M. Nawi, and R. Y. Zhong, "Smart automated guided vehicles for manufacturing in the context of industry 4.0," *Procedia Manufacturing*, vol. 26, pp. 1077–1086, 2018, 46th SME North American Manufacturing Research Conference, NAMRC 46, Texas, USA, ISSN: 2351-9789. DOI: https://doi.org/10.1016/j.promfg.2018.07.144. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2351978918308205.

[5] L. Tebaldi, G. D. Maria, A. Volpi, R. Montanari, and E. Bottani, "Economic evaluation of automated guided vehicles usage in a food company," *Procedia Computer Science*, vol. 180, pp. 1034–1041, 2021, Proceedings of the 2nd International Conference on Industry 4.0 and Smart Manufacturing (ISM 2020), ISSN: 1877-0509.

DOI: https://doi.org/10.1016/j.procs.2021.01.352. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050921004063.

[6] H. Neradilova and G. Fedorko, "Simulation of the supply of workplaces by the agv in the digital factory," *Procedia Engineering*, vol. 192, pp. 638–643, 2017, 12th international scientific conference of young scientists on sustainable, modern and safe transport, ISSN: 1877-7058. DOI: https://doi.org/10.1016/j.proeng.2017.06.110. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705817326565.

[7] R. P. Cant and S. J. Cooper, "Use of simulation-based learning in undergraduate nurse education: An umbrella systematic review," *Nurse Education Today*, vol. 49, pp. 63–71, 2017, ISSN: 0260-6917. DOI: https://doi.org/10.1016/j.nedt.2016.11.015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0260691716302751.

[8] P. Thalamy, B. Piranda, A. Naz, and J. Bourgeois, "Visiblesim: A behavioral simulation framework for lattice modular robots," *Robotics and Autonomous Systems*, p. 103 913, 2021, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2021.103913. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889021001986.

[9] S. Tselegkaridis and T. Sapounidis, "Simulators in educational robotics: A review," *Education Sciences*, vol. 11, no. 1, 2021, ISSN: 2227-7102. DOI: 10.3390/educsci11010011. [Online]. Available: https://www.mdpi.com/2227-7102/11/1/11.

[10] L. Eckert, L. Piardi, J. Lima, P. Costa, A. Valente, and A. Nakano, "3d simulator based on simtwo to evaluate algorithms in micromouse competition," in *New Knowledge in Information Systems and Technologies*, Á. Rocha, H. Adeli, L. P. Reis, and S. Costanzo, Eds., Cham: Springer International Publishing, 2019, pp. 896–903.

[11] S. Papert, *Constructionism: A new opportunity for elementary science education.* Massachusetts Institute of Technology, Media Laboratory, Epistemology and ..., 1986.

[12] L. Richards, "Designing engineering teaching kits (etks) for middle school students," in *2003 Annual Conference*, https://peer.asee.org/12509, Nashville, Tennessee: ASEE Conferences, Jun. 2003.

[13] I. M. L. Souza, W. L. Andrade, L. M. R. Sampaio, and A. L. S. O. Araujo, "A systematic review on the use of lego® robotics in education," in *2018 IEEE Frontiers in Education Conference (FIE)*, 2018, pp. 1–9. DOI: 10.1109/FIE.2018.8658751.

[14] F. R. Sullivan and J. Heffernan, "Robotic construction kits as computational manipulatives for learning in the stem disciplines," *Journal of Research on Technology in Education*, vol. 48, no. 2, pp. 105–128, 2016. DOI: 10.1080/15391523.2016.1146563. eprint: https://doi.org/10.1080/15391523.2016.1146563. [Online]. Available: https://doi.org/10.1080/15391523.2016.1146563.

[15] W. Sun, B. Kramer, Z. Li, and J. Stuart, "A review of the commercial trainers and experiment kits for teaching renewable energy manufacturing," in *Proceedings of The 2014 IAJC/ISAM Joint International Conference*, 2014.

[16] C. Paulo, G. José, L. José, and M. Paulo, "Simtwo realistic simulator: A tool for the development and validation of robot software," *Theory and Applications of Mathematics & Computer Science*, vol. 1, Pages: 17–20, Apr. 2011. [Online]. Available: https://uav.ro/applications/se/journal/index.php/TAMCS/article/view/3.

[17] G. Amaral and P. Costa, *U.Porto Journal of Engineering*, pp. 80–88, 2015, ISSN: 2183-6493. DOI: https://doi.org/10.24840/2183-6493_001.001_0008. [Online]. Available: https://journalengineering.fe.up.pt/index.php/upjeng/article/view/2183-6493_001.001_0008/24.

[18] SolidWorks. "Dassault systemes." (), [Online]. Available: https://www.solidworks.com/sw/docs/3DS_2016_SWK_CorpFactSheet_2015_2H.pdf. (accessed: 01.09.2021).

[19] A. A. Galadima, "Arduino as a learning tool," in *2014 11th International Conference on Electronics, Computer and Computation (ICECCO)*, 2014, pp. 1–4. DOI: 10.1109/ICECCO.2014.6997577.