ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

STI Interdisciplinary Robot Competition

MA2 - Semester Project

Students - Team 3

Calabro Julien
Munier Louis
Perret Michael

Professor in charge

Ijspeert Auke

Supervisor

Crespi Alessandro

Coachs

Hauser Simon
Vivek Ramachandran

June 13, 2019

Abstract

This report presents the approach taken to meet the specifications of the Interdisciplinary Robotics Competition STI, which has been held every year at EPFL. The latter proposes the following problem:

- Move and locate itself on an arena where are obstacles and different types of ground (grass, pebbles and inclined path leading to a raised platform),
- Spot, then pick up scattered PET bottles,
- Return bottles to the recycling area.

To do this, we imagined a compact robot based on the martian rover idea. It would detect bottles through a camera and use rotating brooms to catch the bottles on the ground to throw them into the recycling area. For the mechanical part we mainly used MDF, aluminium and 3D printing. The locomotion, the obstacle avoidance and the throwing protocol is done by an Arduino ATMega2560 while all the image processing is done by a Raspberry Pi 3 model B with the help of a Raspberry Pi camera and a webcam. The environment is detect by infrared sensors and an ultrasonic sensor is used to know when a bottle is grabbed.

The robot is moving randomly by avoid the obstacles as long as it does not detect bottle. We do not have some localization as we only need the distance and the angle between our robot and the recycling area. To do this, a webcam mounted on a stepper is used to stay fixed on the LED beacons of the recycling area. Regarding object detection, we finally chose to use the property of the PET. In the arena it is the only material that reflect the light so well. To do so we added six power LEDs around our Raspberry Pi camera and used image processing to avoid false positive due to beacon LEDs and other situation when the light saturates. To pick up the bottles we were inspired by machines that clean the streets and we made our own rotating brooms. The solution to not waste time by going back and forth between the position of a bottle and the recycling zone is to launch them directly there. For that we were inspired by the spinshot tennis ball machines.

The rest of the report details our design process starting with the imagined solutions, a feasibility analysis and functionality study, then the development, tests and changes due to the problems encountered, the assembly, the programming and finally the results in the real situation.

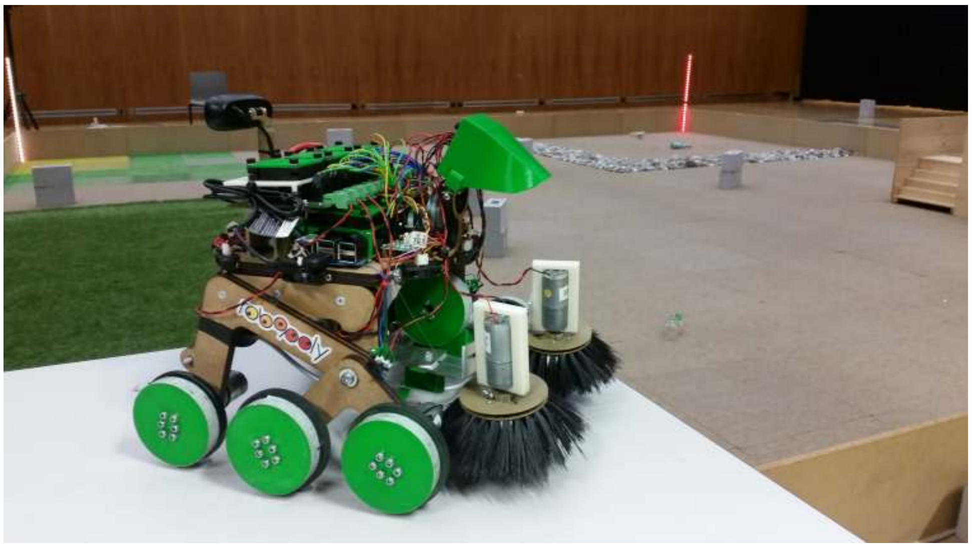


Figure 1: Final design of our robot just before competition

Contents

1 Project description

1.1 Rules

"The goal of the competition is to build a robot (or a team of robots) that will autonomously collect recyclable litter (PET bottles) in a challenging, semi-structured environment." [1]. The time limit to do this is ten minutes and no communication between competitors and robot is allowed. All computation must be done on-board and the robot have to use a battery to power up itself and his components. We can use multiple robot if necessary but no flying solution. The robot must not be dangerous for the public and must not damage the arena. There is no weight limit but the solution have to be contained in one cubic meter. The last rule is to use at most the real and virtual budget allowed to construct our robot by buying components in real shops or using a virtual catalog, no personal components are allowed.

1.2 Arena

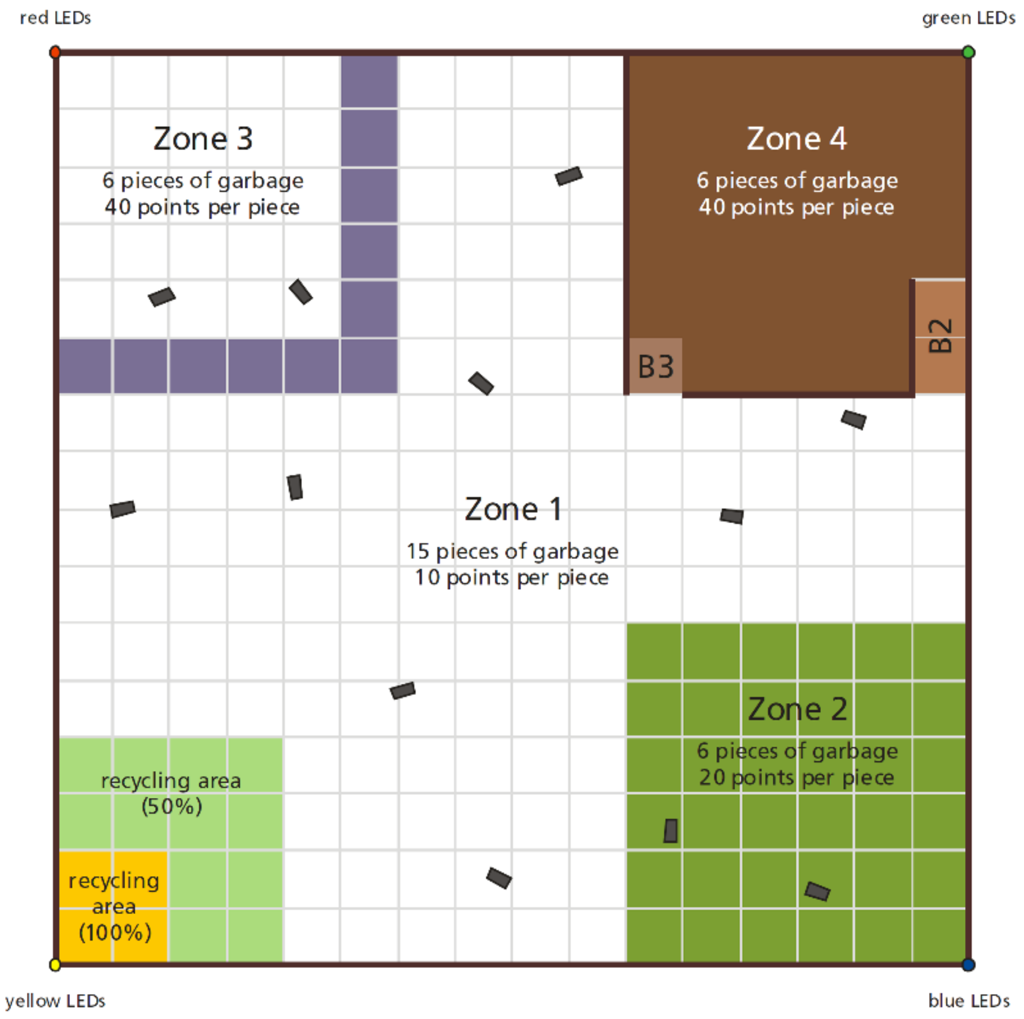


Figure 2: The arena

The map is a square of 8m of sides composed at each corner of different area: recycling zone, grass zone in *zone* 2, elevated zone in *zone* 4 accessible by stairs or via a gentle slope and a zone surrounded of stones in *zone* 3. All the zones excepted the grass one are composed with squared carpets of 50 cm of sides. Each corner has a colored beacon of LEDs to help in localization. Moreover all the map will have randomly distributed obstacles and bottles.

2 Project management

2.1 Task division

The distribution of tasks was done as the project progressed. Each of us performed tests for the tasks assigned to us.

2.1.1 Julien

At the beginning of the project I was in charge to make research about the mechanism we thought to catch the bottles. After we chose the design of our robot I made the full CAD of it. In parallel I designed the PCB for the power board. When we received it I solder all the components on it. Afterwards I made the cables and wired the electronics and made some 3D prints. I was in charge of the Arduino programming for the rest of the project.

2.1.2 Louis

Since we started this project my responsibility was to do all the image processing part and to be comfortable with our Raspberry Pi. I was also in charge to search on the processor and throwing part. When the design and components was chosen I went to cut the MDF parts at Made@UC and manage the orders. I also collaborate with Julien to finalize the PCB design and implement the electronic parts. I worked on the green 3D printed parts to cool our system and fixe the LEDs . We worked both of two on all the programming during last weeks.

2.1.3 Michael

I did some tasks like preparing the laser cutting file, then sticking the MDF parts to get the right thickness. I have prepared cables for electronics, as well as crimped connectors for infrared sensors. Having done CNC training, I machined our aluminum plates and realized the folding with Julien. I bought the broom that I used to make the rotating brooms. Then, I have designed and printed in 3D some pieces to solve some problems. I made the changes so that the propulsion wheels could accept any type of bottle. I made the video presentation for public competition and start writing the report as they finished the final adjustments between the Arduino and the Raspberry for the private and public competition.

2.2 Budget management

Regarding the management of the budget, we were very careful to not exceed the budget and took as many components as possible from the virtual shop. For the components we needed to buy, we have always attached importance to the price and the supplier to avoid unpleasant surprises such as delivery time and shipping costs. We also bought some spare pieces in case of breaking by ourselves. At the end we had almost enough components to build a second robot if the first one totally broke. Our biggest expense in the virtual budget is the ACI's PCB realization with the different motors and infrared sensors. In real budget, these are LiPo batteries. All our real and virtual purchases are listed in appendix D and finally we bought around $1'500$ CHF evenly divided between virtual and real budget.

3 Strategy and design

By hearing the rules and the arena on which our robot was going to evolve, we quickly agreed on an inspiring rover design [2] to pass on grass and stones. At the beginning of semester, we made some brainstorming to think about our strategy. We wanted to use simple solutions and implement them good instead making complicated things and having problems.

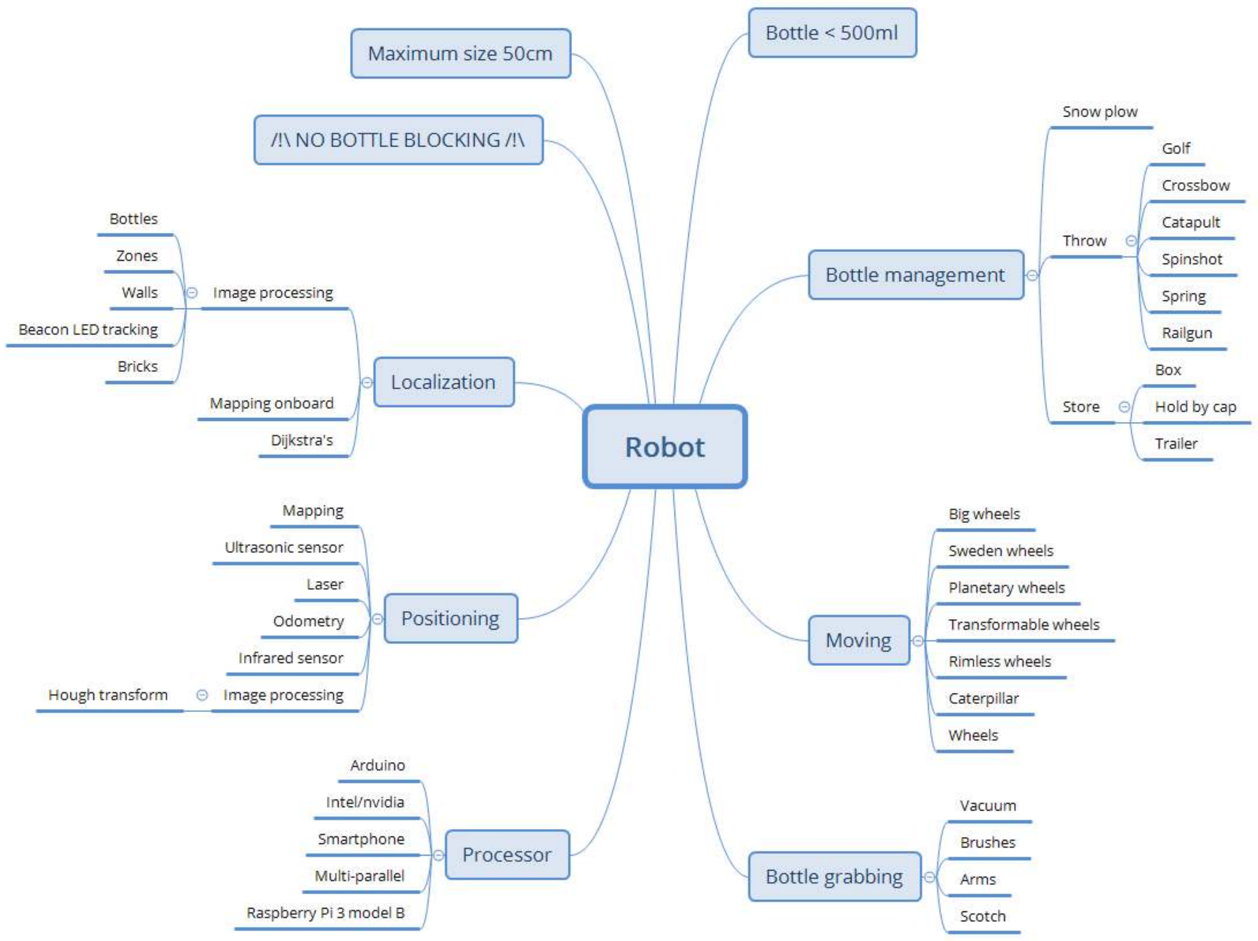


Figure 3: Our first brainstorming

As we can see on the figure above, we get plenty of ideas and solutions. We made a first selection to reduce the possible solutions but it was not enough. That is why we take all them separately to do more research and make a functional analysis before having our definitive choice.

4 Functional analysis

In the following tables, the points are the sum of the values estimated for some criteria of choice. The scale ranged from 1 to 6 with 1 meaning bad or difficult and 6 being excellent or easy. The criteria were the implementation, the space required, the feasibility, the repeatability, the efficiency or the accuracy. With these criterion the maximum possible is 36 for a solution. All the solutions with a score of 0 were excluded without any in-depth study because of these complexity.

Table 1: Function analysis of the positioning part

Positioning			
What	**Pros**	**Cons**	**Points**
Image processing	Objects localization Shape detection	Implementation (OpenCV) Computational cost	27.5
Infrared	Light intensity measurement Cheap Easy to use	Ambient intensity Short distance	34
Ultrasonic	Long distance Cheap	Shape sensitivity	34
Laser	Long distance	Expensive	33
Hough transform	Shape detection	Computational cost Hard to implement	20

Table 2: Function analysis of the grabbing part

Grabbing			
What	**Pros**	**Cons**	**Points**
Vaccum		Efficiency Space Power	0
Scotch	Easy Cheap	Dust Contact surface	35
Rotating brooms	Continuity Easy	Space	36
Electrostatics		Bottles should not touch the ground	0
Arms	Already makes its proof	Sequential Space Design	33

Table 3: Function analysis of the wheels

Wheels			
What	**Pros**	**Cons**	**Points**
Big one	Easy to implement Move "everywhere"	Stairs	29
Rimless	Swiping Move "everywhere"	Stairs Accuracy Design	0
Planetary	Good yield Move "everywhere"	Hard to implement Resources	0
Tank	Move "everywhere" Adhesion	Hard to implement	21
Transformable	Move "everywhere"	Hard to implement Design	20
Sweden wheels	Degree of mobility	Controlability Adhesion Stairs and rocks	0

Table 4: Function analysis of the throwing part

Throwing			
What	**Pros**	**Cons**	**Points**
Glof / Catapult	Efficiency	Machine Learning Impl. Space Loading	0
Spinshot	Efficiency Continuability	Calibration Design Implementation	31
Railgun	Efficiency	Loading Continuously	25
Spring/Crossbow	Efficiency	Loading	30
Linear motor	Accuracy Implementation	Speed	30.5
Pneumatic	Force Loading More bottle	Accuracy Space	24

Table 5: Function analysis of the processing part

Processor			
What	**Pros**	**Cons**	**Points**
Raspberry Pi 3B	Connection Community	Computational power	32
Arduino	Community	Computational power	31
Smartphone	Computational power	Connection Implementation Price	28
Intel	Computational power	Price Power consumption	0
Odroid-XU4	More computational power Documentation	Community Price	31
Nvidia jetson	Big computational power in I.P. Documentation	Implementation Price	0

Finally we do not chose only the functions that have the biggest score but we take care about our skills to implement each of them and our motivation to take some more difficult solutions. We would like to have globally simple solution to be able to concentrate all of our efforts in some particular features like spinshot and image processing.

Table 6: Definitive choice after the functional analysis

Definitive choices	
Positioning	Image processing Infrared and ultrasound sensors (obstacles)
Wheels	Normal one
Throwing	Spinshot
Grabbing	Rotating brooms
Processor	Raspberry Pi 3B and Arduino ATmega2056

5 Design conception

5.1 Locomotion

The idea is a 6-wheels rover to be able to pass on each zone. A part of MDF has 2 wheels that can rotate freely around another MDF part that features the last wheel. These two pieces of MDF are connected to a stable connecting plate that includes all the necessary electronics. As we will not need any odometry we decided to use 6 DC motors with reducer and without encoder, the *75:1 Metal Gearmotor 25Dx54L mm HP 6V*. For motor control, we simply need to send a PWM with the Arduino to the H bridge driver [3] to adjust the speed. With this, we can move the robot into the arena to retrieve the bottles. Our strategy of motion is to move straight until a bottle or an obstacle is detected. The behaviour of the robot in these cases will be explain below.

5.2 Localization

Since the robot will move randomly by avoiding obstacles we only need to find a way to quickly spot the recycling area to launch our bottles. To accomplish this, we imagined a system consisting of a camera mounted on a stepper motor that would aim to stay centered on the beacon of the recycling area. All the software procedure is detailed in section 11.2.2.

5.3 Bottle detection

In order to detect the bottles, we started on the idea of using a Raspberry Pi camera. During the first part of this project we try some methods to have learning in bottles detection.

5.3.1 YOLO

The first idea was to take a look at *YOLO* [4] and see if it could be implemented to our project. This method works pretty good and can be implemented on the Raspberry Pi with a good framerate but decreasing the accuracy by using the *tiny YOLO* version. Since this method is young and it is quite difficult to have our custom dataset we directly put it away.

5.3.2 Haar cascade

An other learning method that works well in this situation is the haar cascade. We tried a lot to make a well set of data because the more data we have to train the model the better it will be. So we took a lot of photos like in figure 4 in order to train our own haar cascade model.

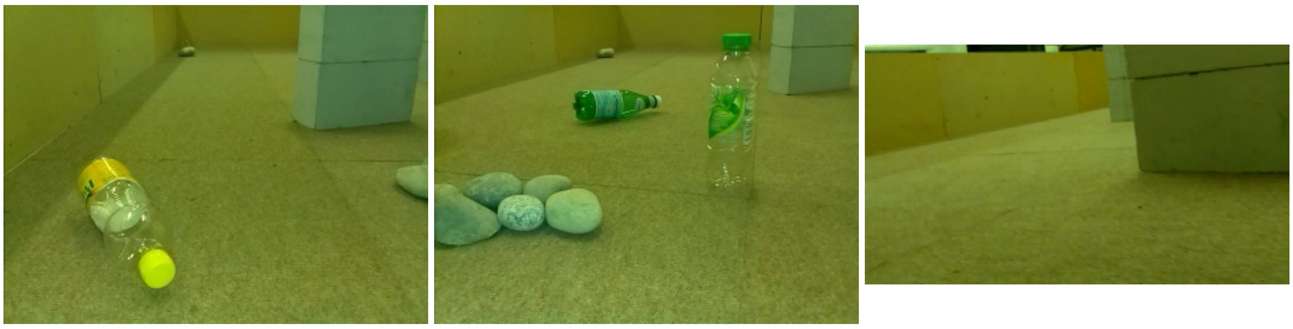


Figure 4: Two positive samples and one negative of our dataset of the 17th may 2018

Because of some mistakes during the creation of bounding boxes dataset the first results were not good enough. Since the project quickly progresses and we have to do some parts which was on the Michael's responsibility we will avoid this technique in order to use an other one which is simpler to implement but demand more tuning. It is the last method explored and the final one we used.

5.3.3 Filter

Since we realized during making the dataset that the bottles are the principal component in arena which is reflecting the light from our frontal LEDs so well we will use this particularity to detect them. In order to avoid beacon LEDs detection as a bottle we take two pictures, one with light turned on and an other one with light turned off. Since the LEDs are flashing, the scene is easier saturating so we setup the camera in very low luminosity to avoid saturation point on the image if it is not a bottle. At the end we apply some basic filters on our images in order to easy localize the bottle. The exact procedure is detailed in section 11.2.1.

5.4 Bottle collection

To collect PET bottles placed in all possible orientations we chose to use rotating brushes. In the first place we thought to buy them already ready made but after searching on the internet it was difficult to find one that suited us and whose bristle would work well on the PET. To make our own brushes we bought a broom and cut the bristles of it in order to fix them in a 3D printed base.

5.5 Bottle recycling

To avoid wasting time returning to the recycling area to drop our bottles we thought of a spinshot machine to eject them into this area directly after having picked up. So at most, we should be able to shoot at

$$\sqrt{s^2 + s^2} = \sqrt{2 \cdot 8^2} = 11.31m$$

where s is the side of the arena. This distance correspond to the diagonal of the arena, the maximum distance we have to reach is we were targeting to go everywhere. In our strategy we exclude to go to the ramp therefore we have to throw at a lower distance than this one. After testing our system we saw that we can send a bottle a bit more than the half diagonal which is sufficient for the zone we considered to explore.

5.6 Bottle management

Once we grabbed a bottle, we needed to orientate it at 45° and take it to the propelling wheels. For that we implemented two aluminum rods which will guide the bottle to the proper position. Moreover a guiding wheel placed above the entry of the robot ensure the motion of the bottle inside the robot. Only the backward extremity of the rods are fixed to the mechanical base of the robot. The forward extremity is linked to a stepper motor by two wires. Thanks to this design the rods can freely rotate. Two reasons motivate ourselves to do it like that.

1. Since these rods are the weakest part of the robot we lift up them until a bottle is detected to avoid any stones collisions and then we reduce the probability to break them.
2. When a bottle is grabbed the Arduino communicate it to the Raspberry Pi and the fire procedure is engaged. The moving rods help to adapt our system to any type of bottles.

5.7 Obstacle avoidance

In order to avoid obstacles and walls we used infrared sensors. We have chosen *sharp GP2Y0A21* from the virtual shop because of their measuring range is from 10 to 80cm which is perfect for our use. We have implemented a Braitenberg's vehicle following the concept that the speed of the motor is directly proportional to the signal being detected by the infrared sensor. In others words, the closer we are from the wall, the faster we turn to avoid it. However, if the turn produced does not allow us to dodge the obstacle and we reach the limit value allowed, the robot stops, does a maneuver corresponding to the situation and go forward again. This will be explain in details later.

6 Proof of concept

To be sure that our solutions can work, we started very early testing by building small prototypes that we will detail now.

6.1 Rover design

We built a basic rover structure to see realizable dimensions and angles. We found some wood and lego wheels at Robopoly [5] and this was the first try:

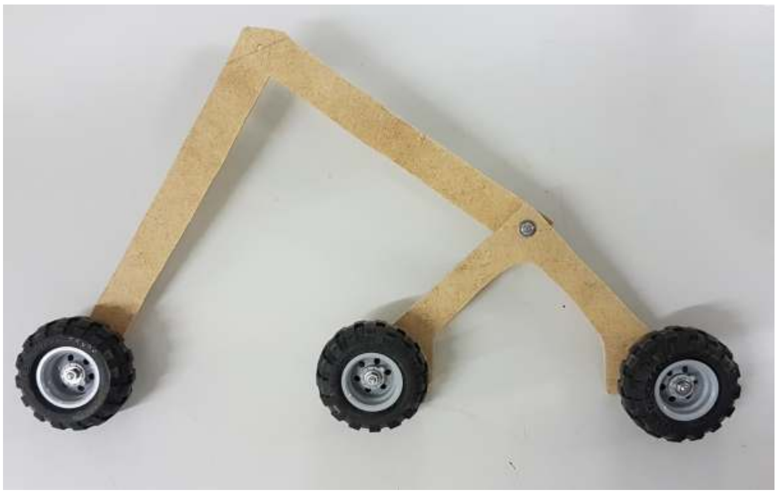


Figure 5: Prototype of the rover design

6.2 Rotating brooms

The first prototype built is the one on the left in figure 6 composed of a central piece of wood on which we glued electric cables. Subsequently, we added the heat-shrink sleeves at the end in order to have a better rubbing material. The video *First_rotating_brooms.mp4* shows how they work and we see that it works pretty well. Then, we wanted to buy ready made brushes but we did not find the right size so we made a first version of them as we can see below on the right picture. The latter allowed us to realize that we had to pay attention to the angle of attack of the bristle and increase the density. We corrected all this on version two which is on the final robot.

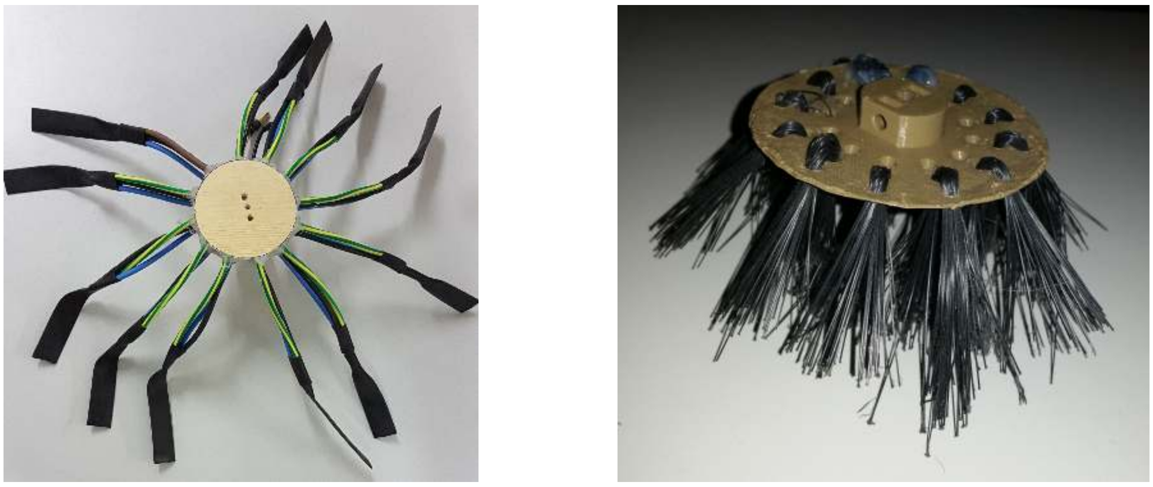


Figure 6: Evolution of the rotating brooms

6.3 Propelling wheels

One of the most important parts of the robot is the bottle ejection system. We started very early tests of the latter to solve all the problems that would have forced us to change the solution. We tinkered an installation with lego wheels and drone engines from a friend to see the result. Due to the high rotation speed, we immediately noticed that the lego wheel would not do the trick. We printed flexible filament wheels at Robopoly, then made new tests and these ones were more conclusive. We did a lot of video showing these prototypes in action and they are all in appendix.

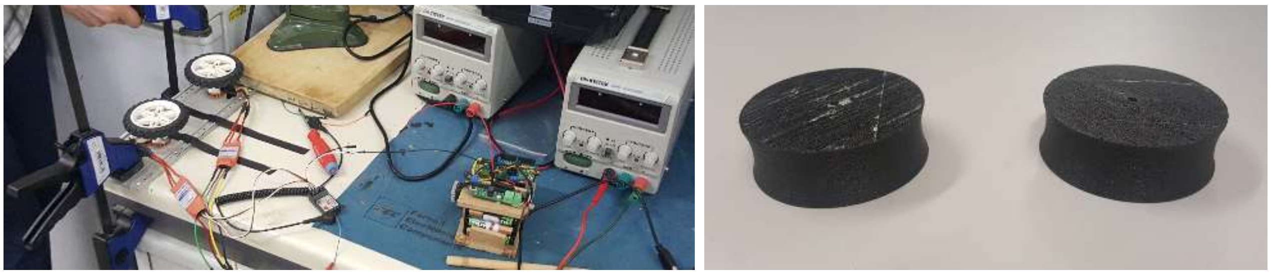


Figure 7: On the left the first setup and on the right the finale flexible wheels

6.4 Obstacle avoidance

In order to feel our environment, we use some *Sharp GP2Y0A21* infrared sensors to detect walls and obstacles in the arena. We did tests with them and ultrasonic sensors to detect the bottles. It is interesting to note that the infrared passes through the PET so it is hard to detect a bottle unless it falls on the label or cap. That is why we decided to do bottle detection by image processing and used infrared sensors to avoid obstacles. We use an ultrasound sensor inside the robot to check if a bottle is inside to start the alignment and ejection procedure.

6.5 Bottle detection

As a first test to prove that the bottle detection can be done with a Raspberry Pi and his camera we try to perform it with simple HSV threshold and a filter to avoid noise. The result, shown in figure 8 and 9 was satisfying at this step of the project but could not be keep because of its dependence to light environment and low robustness due to thresholds.

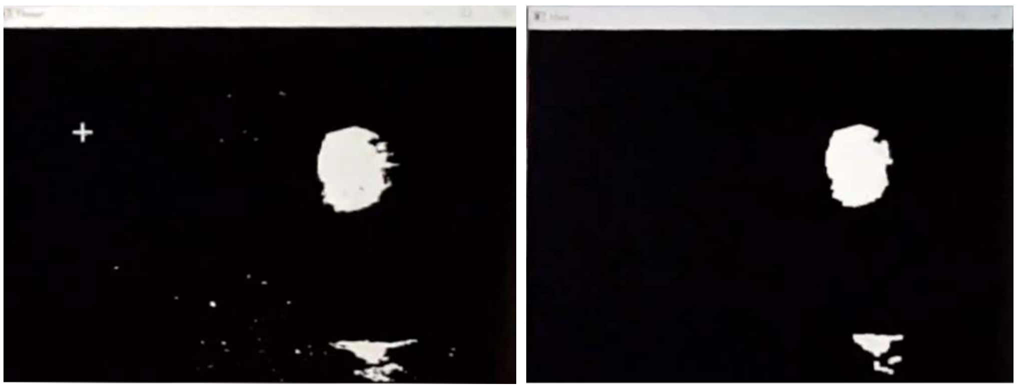


Figure 8: On the left the result after applying threshold and on the right the filtered one

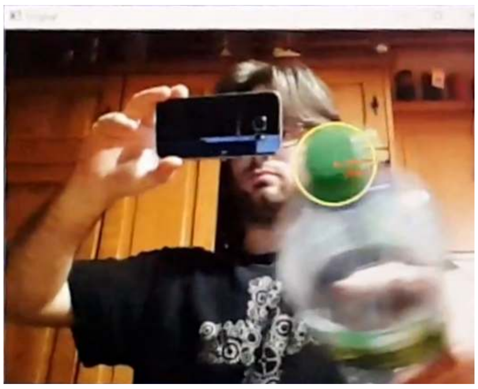


Figure 9: Original image with result of localization

7 Mechanical implementation

7.1 Frame

For the robot body, we assembled the MDF parts together by placing ball bearings in the necessary shafts and screwing the rest. To fix the top plate, we used simple squares. The motors to move are housed in their locations at the ends of the MDF sides and we simply tighten them with a screw and a nut. The remains of the elements are fixed above, below or on the sides of the upper plate.

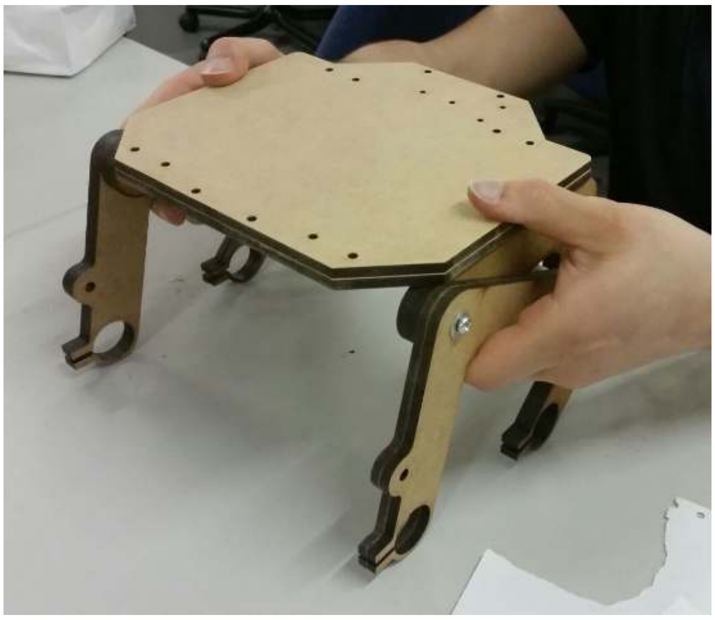


Figure 10: Only the MDF parts that composed our robot

7.2 Locomotion

The initial choice of simple wheels and the rover design was quickly realized and tested. The latter was intended to be simple and effective on the stones. We chose wheels of 80mm diameter and 10mm thickness of the brand Pololu and some *75: 1 Metal Gearmotor 25Dx54L mm HP 6V* motors from the virtual catalog. We knew by choosing them that the axis of the motor and the axis of rotation of the wheel would not have the same diameter. We could buy the connectors to remedy to this but we prefered to save money and do it ourselves by 3D printing.

To drive our motors we have to design our own motor power board. We chose the *TLE9201SG H-bridge* to provide the current and the necessary voltage by using a simple PWM command. This board will be treated in section 8.3. After some tests, an adaptation of the wheels has to be found because of the wheel's thickness the risk of getting stuck in the pebbles is too high. To try to overcome this we add an extension to the wheels thus when they are blocked and skate in the pebbles the extension compensate the contact zone. This extension is wider and with a smaller radius to enlarge our wheels without adding friction. We add some double-sided tape to have a good friction when this extension is used on stones. Finally it does not help so much but as we were already two days before competition we did not do an other test to improve this part and decide to avoid stones.

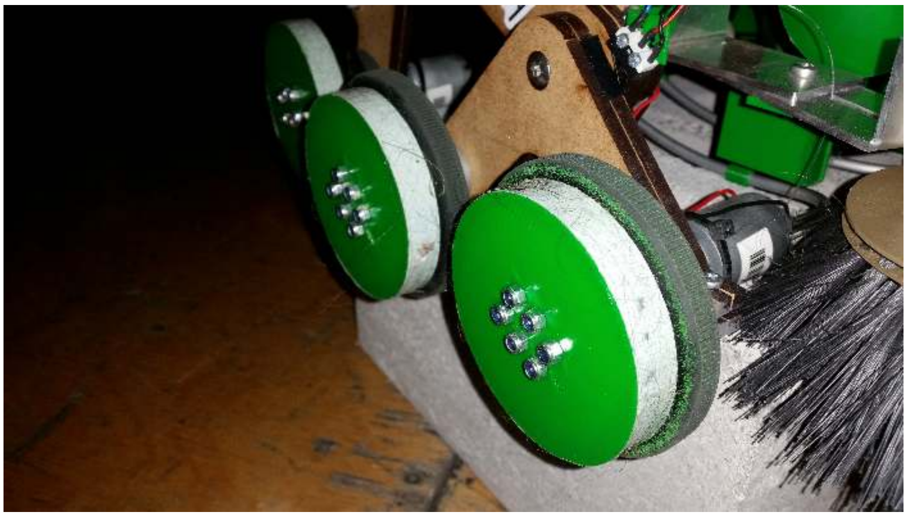


Figure 11: Wheels with the extension for the stones

7.2.1 Stone or slope detection

Because of the risk to be blocked we added a mechanical sensor to detect the elevation of the front wheels indicating a slope or passing over pebbles. The use of this sensor allowed us to implement a motion to avoid going into the rocky zone and elevated platform during the first seven minutes. After this time, we could turn off this sensor and go over the rocks or climb the slope.

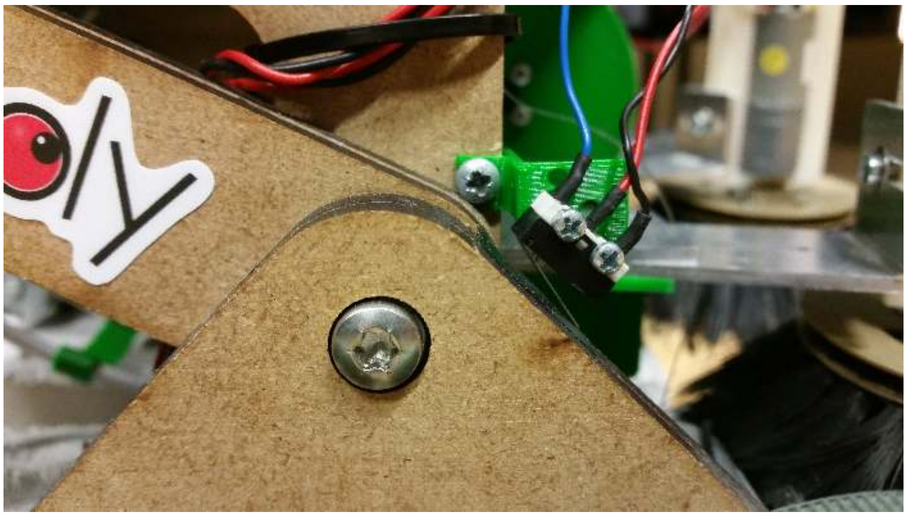


Figure 12: Mechanical sensor to detect stones or slope

7.3 Distance to recycling area

As said before, we used a stepper motor *Debo Moto1 from JOY-IT* and a 3D printed support to elevate the webcam high enough so that it can always keep the beacon of the recycling zone in its field of view. The camera that we chose is a normal webcam for computer. The *Logitech HD Webcam C270* has a resolution of 3 megapixels but we used a resolution of 320×240 to take pictures because the image processing takes more time on bigger images and we do not need more precision. We connected the webcam to the Raspberry using the normal USB cable. The code used to perform this operation is detailed in section 11.2.2.

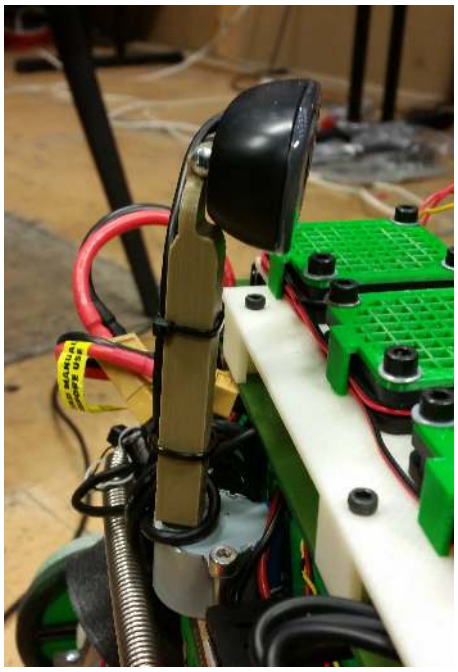


Figure 13: Installation of the webcam on its stepper

7.4 Bottle collection

The first tests of the rotating brooms worked very well. To design the version fixed on the robot, we have printed parts in 3D and fixed on the bristles of a broom. This piece is attached to the motor which is itself in another part in 3D to maintain it and adjust it. The motors used here are *34:1 Metal Gearmotor 25Dx52L mm HP 6V* from the virtual shop. Indeed, for the adjustment of our brooms, we used the CNC to cut two aluminum plates in order to be able to lengthen or shorten, to change the height and the orientation of the brushes. After new test on the robot, we realize that sometimes the bottle does not get inside with the right orientation so we decided to add two 3D parts more to force the path.

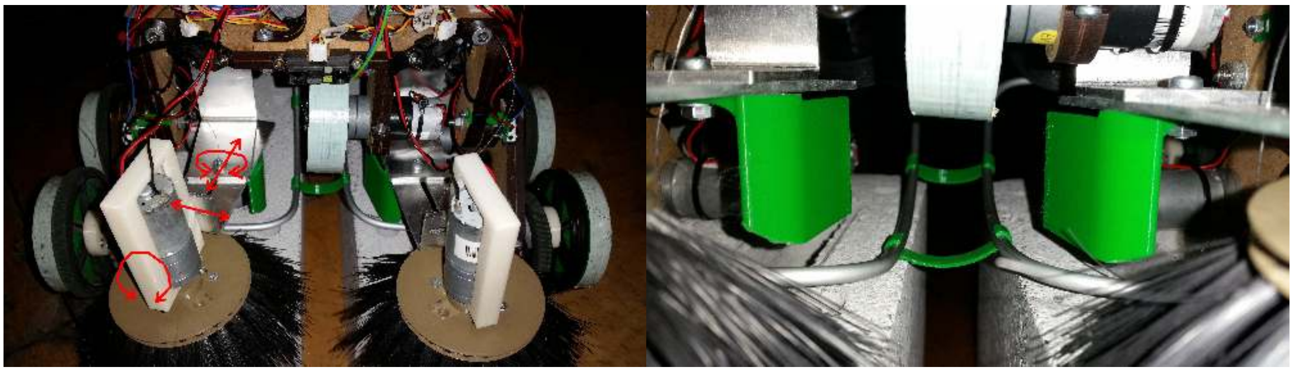


Figure 14: On the left presentation of the adjustable holder of the rotating brooms. On the right the green plastic parts force bottle guidance

7.5 Bottle management

The guides are aluminum rods of 5 mm diameter that we cut and folded to obtain the desired shape to guide the bottles. The placement of these guides on the sides part of the robot is made with holes slightly wider than 5mm in diameter and lock rings prevent the bars from moving. These rods are connected to a pulley that turns to wind or unwind the wire to go up or down.

The guiding wheel is fixed on a *34: 1 Metal Gearmotor 25Dx52L mm HP 6V* motor of the virtual shop which is held by a piece of MDF by clamping. The MDF piece is attached to the top plate by a square. The first version of the guide wheel was in flexible filament but we replace it by a 3D printed part with double-sided tape all along the perimeters to have a better adhesion and contact with the bottle.

7.6 Bottle throwing

To maintain the brushless motors at the right height and angle, we also machined at the CNC some aluminum plates. These pieces will serve as fixings to be rigid against the vibrations produced by the rotation of our high speed motors. When the engines being fixed we add the 3D printed flexible wheels. The first tests were conclusive however we had to redo these aluminum parts to adjust the height to grab the bottle in the middle. Despite these changes, we still were not convinced of real change in terms of catch. We passed the fixation from below to above in order to raise more the both wheels.

After some tests in the real arena we realized that some bottles were much wider than our tested ones and so they were stuck between the propelling wheels. This is why we slightly unclamped the screws that hold the aluminum plates to add some play in the distance between our wheels. In order to keep a good force on the bottle who will thrown we added a spring between the two aluminium fixations. In this way the bottles can have different diameter but they will all be tight when they pass between the two propelling wheels. Thanks to this solution, we also fill the difference in ejection distance that appeared if the bottle entered through the cap or from the bottom.

In addition, we noticed that sometimes the bottle was not correctly grabbed by our propelling will and try to pass above them. To remedy this problem we added a piece to prevent the bottle from jumping and guide it to the exit. We put a ball bearing to avoid frictions.

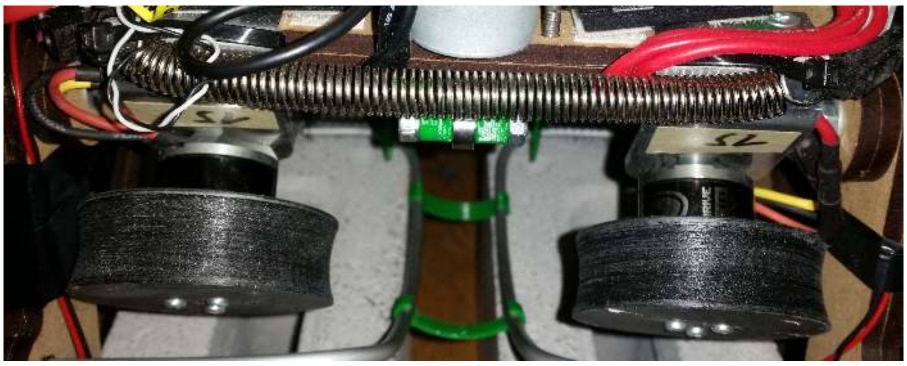


Figure 15: Throwing system with brushless, aluminum and 3D printed parts

7.7 Obstacle avoidance

Seven infrared sensors placed on the side of the top plate are used to avoid obstacles. These are distributed as follows to allow detection from all sides and to keep a good distance:

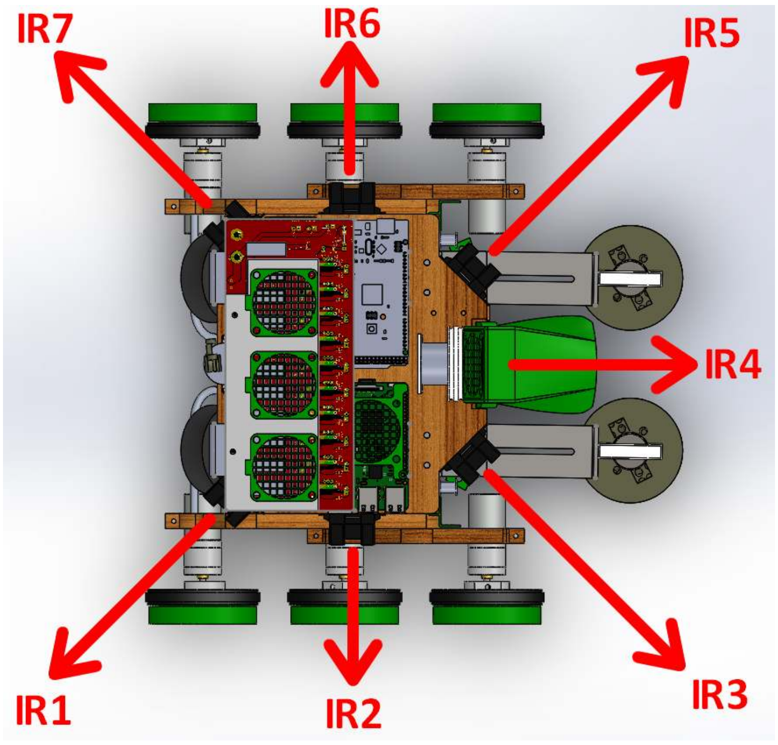


Figure 16: Infrared sensor's position on the robot. One is hidden by the cap

8 Electronic implementation

Regarding electronics, we used a Raspberry Pi 3 model B to do all the image processing which has good computational capabilities. Since we have only one Raspberry Pi and we would like to have a camera to acquire images from the front of the robot and another one which always track the recycling area we chose the Raspberry Pi camera and a webcam to do the job. This webcam mounted on a stepper is able to determine the distance and the angle between to the recycling area.

To manage the environment and the motors driving we chose an Arduino which is designed to develop these used. A custom power board was done by Julien to be able to control all our nine DC motors with PWM and to have connections to power the Arduino, ESC and LEDs directly to our battery.

The communication between the Arduino and Raspberry Pi is implemented by using UART protocol. It is detailed in section 11.2.3. To power all these components, we chose a LiPo battery with 3 cells and a high capacity of $5'200$ mAh that gives us enough margins to turn for ten minutes during the competition. All these parts are detailed in this section.

8.1 Bottle detection

To fix the Raspberry Pi camera to the front of the robot we cut a small support in MDF and we fixed it a 3D part that allowed us to orient it. After many tests of image and treatment on the advice of our coach we added LEDs to have a uniform lighting thus we fixed six powerful LEDs on a veroboard because we did not have time to make an other PCB. All the driving part of these LEDs was added to our custom power board in order to have a cleaner solution.

Thanks to this brightness the bottles detection by light reflection on them is a good way. The assembly is shown in pictures 17 and we had to add the green cover to avoid being dazzled when we look at the robot for the comfort of the public and ours. We also took advantage of the fan orders to add one to this assembly. Since these LEDs are not supposed to be welded like that to dissipate their heat it was better to add some cooling.

8.2 Frontal LEDs

Bottles detection is performed by reflecting light on them thus we present here the electrical diagram of this small installation and its realization:

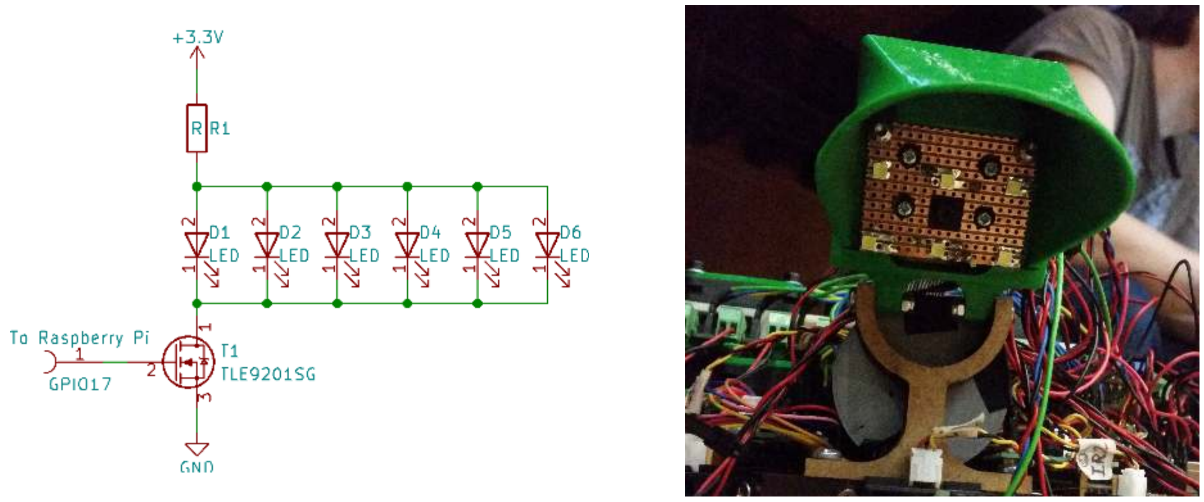


Figure 17: On the left the circuit diagram and on the right the realization

To do this we chose the *LEDs MXA7-PW65-H001* from Lumileds [6] and decide to reduce there current consumption to avoid their heating and breaking so we had to dimension a resistor. We neglect the voltage drop by the transistor when it saturate because it is not significant so we have:

$$R1 = \frac{V_{R1}}{I_{R1}} = \frac{Vcc - V_{max\_led}}{6 \cdot I_{max\_led}}$$

Where all the values are taken from the datasheet and take into account that we powered our LEDs with maximum $3.3V$ thus we have

$$R1 = \frac{3.3 - 3.2}{6 \cdot 240 \cdot 10^{-3}} = 70m\Omega$$

It was our theoretical result but after some tuning after receiving our components we found that a resistor of $200m\Omega$ was more appropriate to reduce the consumption to $200mA$ and be sure to avoid breaking problem because our veroboard is not able to dissipate heating as well as an appropriate PCB.

In order to drive these LEDs with the Raspberry Pi we decide to connect them in parallel and used a MOSFET transistor [7] in saturation mode. By using a MOSFET transistor we do not have to put a resistor to reduce its consumption from the Raspberry Pi which is insignificant.

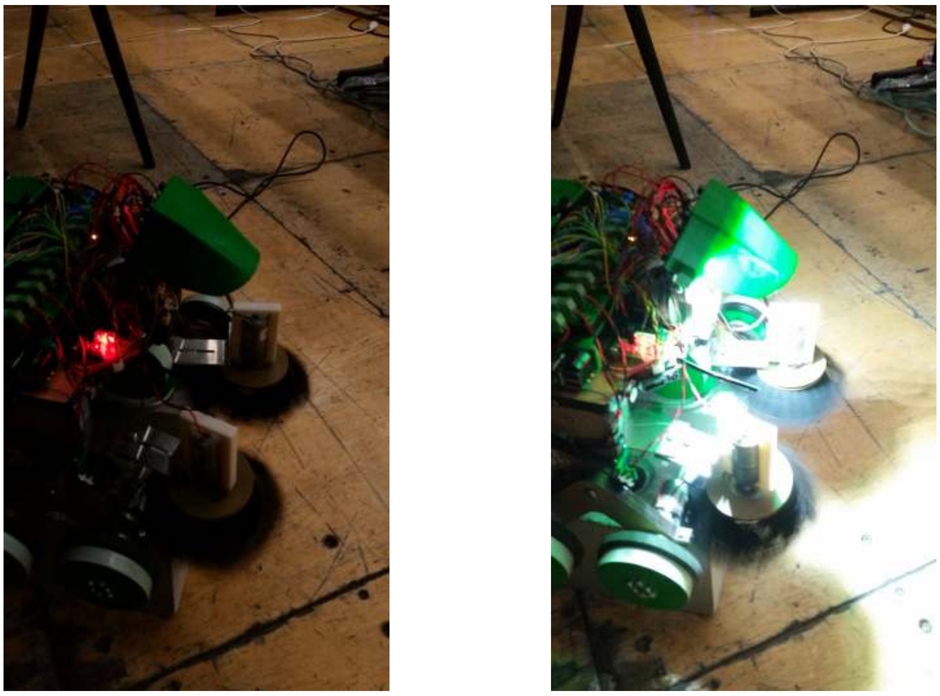


Figure 18: Demonstration of the LEDs illumination

8.3 Motor power board

Our power board mainly consist of ten h bridges, TLE920SG [3]. Moreover we added the power supply output for the ESC and for the LEDs. Below is the schematics of the power board.

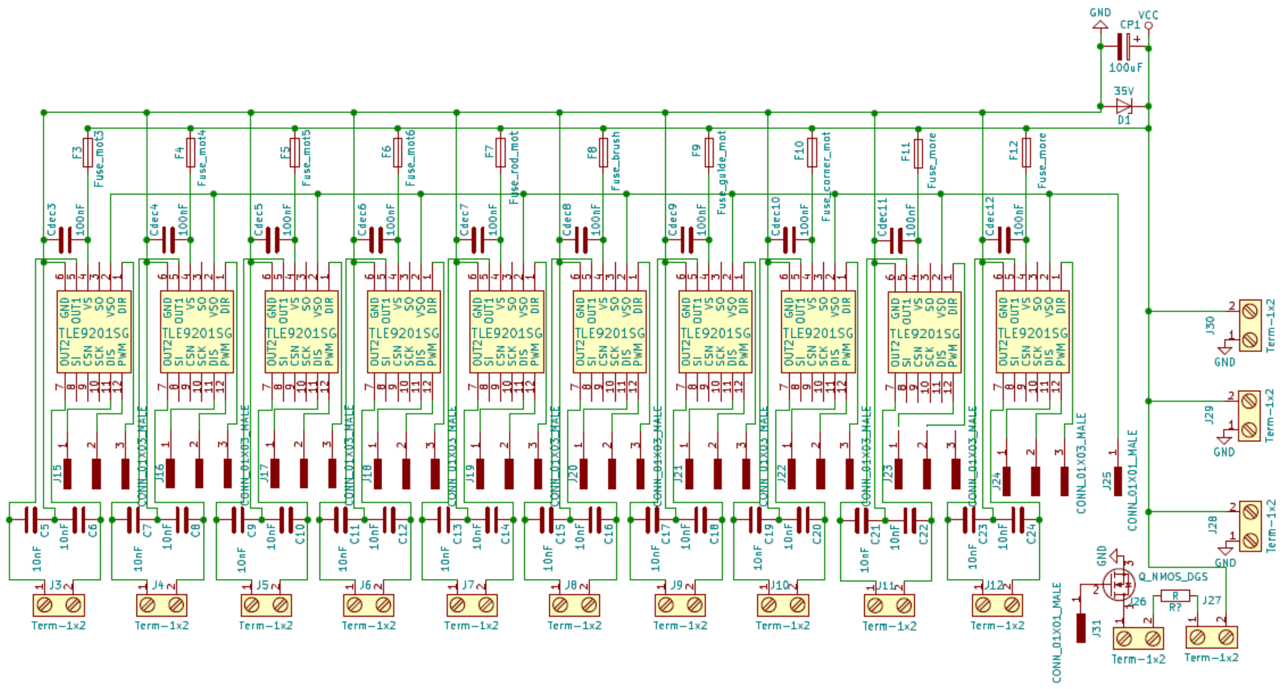


Figure 19: Schematic of our motor power board

8.4 Encountered issues

In the first version of our motor power board shows below, we had omitted a connection that did not seem necessary to us except that it was. In order to start the programming we added this connection afterwards with the red wires. However to have a safer power board and avoid short circuits we decided to make another one without forgetting this connection. In addition, we took the opportunity to reduce the size of our fuse holders which saved us a little space.

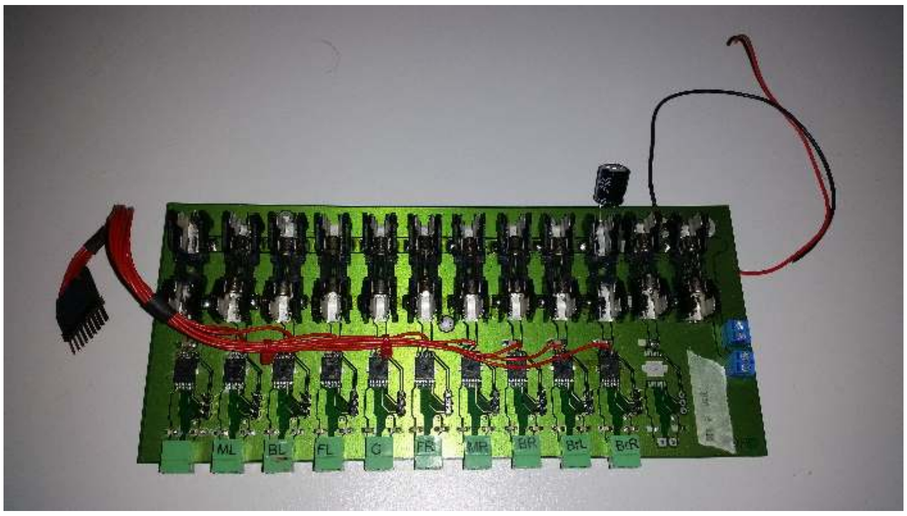


Figure 20: First version of our motor power board

Finally we have done a second version of this power board and we added the LEDs drivers on it instead of putting them on the veroboard. This version is shown in figure 21

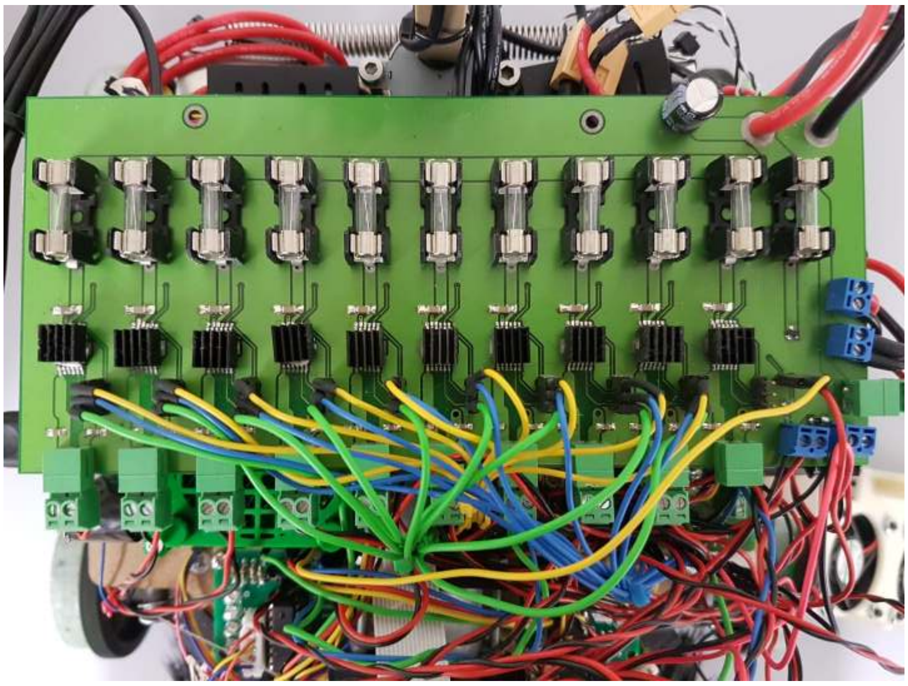


Figure 21: Final version of our motor power board

The only problem we had with the newest version was that our H bridges were giving off too much heat and their thermal protection was locking in before instead of our fuse in case of troubles. To avoid that we placed heatsinks above each H bridges with thermal paste. Moreover we added three fans above the whole board to increase the air flow.

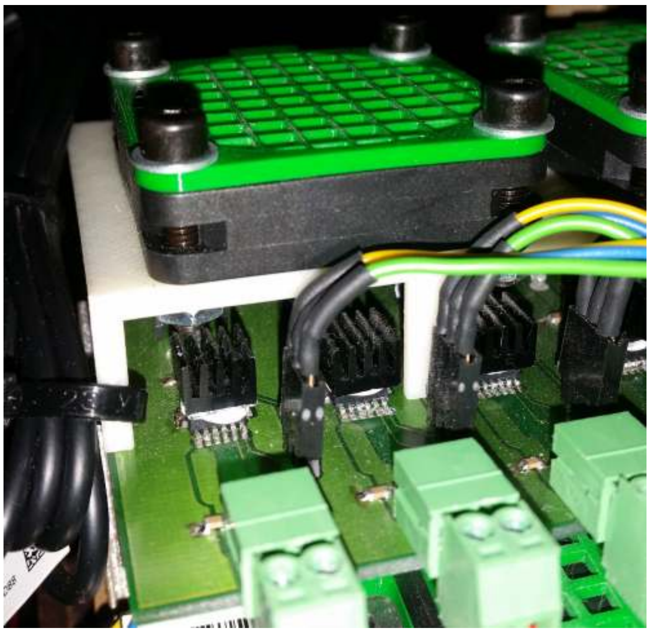


Figure 22: Zoom on the H bridges to show all the cooling solutions implemented

8.5 Raspberry Pi

While the installation of an optimized version of openCV on our Raspberry Pi we compiled the software from the sources and used all the four cores of the Raspberry Pi. This step overheats it so we decided to add heatsinks and a fan on it to avoid under-clocking when we used it to do the image processing. Even if we used C++ language to have a more optimized code we used at least 80 percent of the processor so we had to cool it well. Some 3D printed part was added on top of the screw to avoid short circuits with our custom power board because they are closed enough.

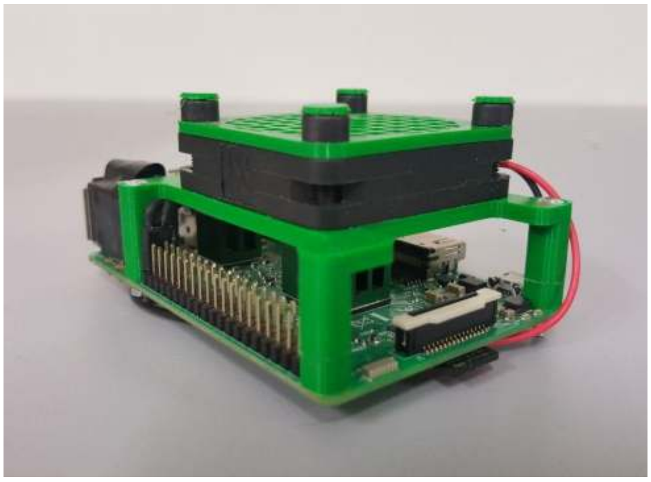


Figure 23: Zoom on the Raspberry Pi cooling solution implemented

8.6 Consumption

Since our robot has to be able to be autonomous during ten minutes we have done some estimates about is consumption. We take care about the worst case and think about the result to dimension our battery.

Table 7: Consumption analysis

What	**Consumption**	**Number**	**Total**
DC motors *move, grab bottles*	550 mA *free run* and 6.5 A *stall* [8]	9	58.5 A
Brushless	20 A max [9]	2	40 A
Stepper	0.5 A by driver (max 2 enable currently) [10]	4	2 A
Fans	121 mA max [11]	5	605 mA
Raspberry Pi 3 model B	2.5 A max [12]	1	2.5 A
Arduino	50 mA per 3.3 V pin used [13]	44	2.2 A
LEDs	200 mA per LED [6]	6	1.2 A
Total consumption			**107 A**

In the worst case when all the components are enabled and consume their maximum current we have around 110 A of consumption. Since the brushless motors will not be enabled at the same times of the moving motors and all the moving motors should not be blocked together we can do the assumption that the maximum current consumed is around 50 A. We have also to notice that this maximum consumption is on a peak and not on a steady state so we can divide by at least two the consumption of all the motors to have the maximum consumption because of we consider the enable time of all these motors the real mean current consumed decrease a lot.

In order to have a big autonomy and be comfortable during the last tests where we used a lot our robot in real conditions we took two batteries oversized to be able to charge one while the second is used. So we did not have time out while the two batteries were discharged. Our choice of battery was two LiPo with 3 cells and 5′200 mAh of capacity [14] because of the accuracy of this technology and its contained size. This battery can provide a constant current of 52 A and 104 A during a peak so it is well enough.

9 Manufacturing

To produce the structure of our robot, we chose to use MDF and cut it with a laser machine. To do this, we asked an acquaintance of makerspace Renens, Made@UC, to use their machines. Since we would like to have a robust robot we cut all our pieces twice in a piece of 5 mm of thickness and glued them together in order to have a 10 mm of thickness for all our wood parts. We cut to be able to construct two robot just in case of breaking a part.

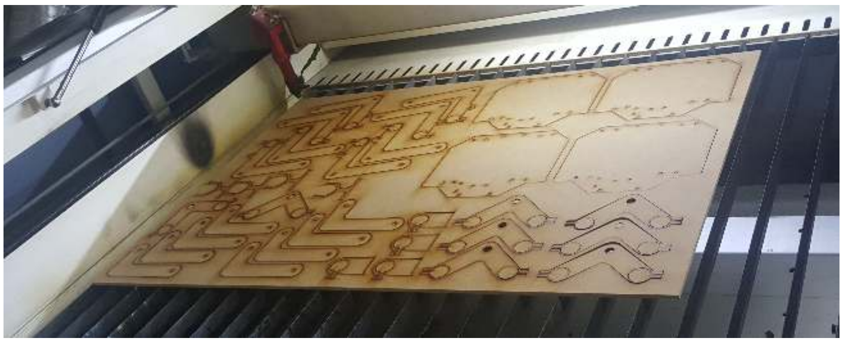


Figure 24: Our MDF frame cut by laser cutting machine

To hold our propulsive motors, we modeled the supports that we cut with the CNC of the ATPR workshop and made some bends to obtain the desired shape.

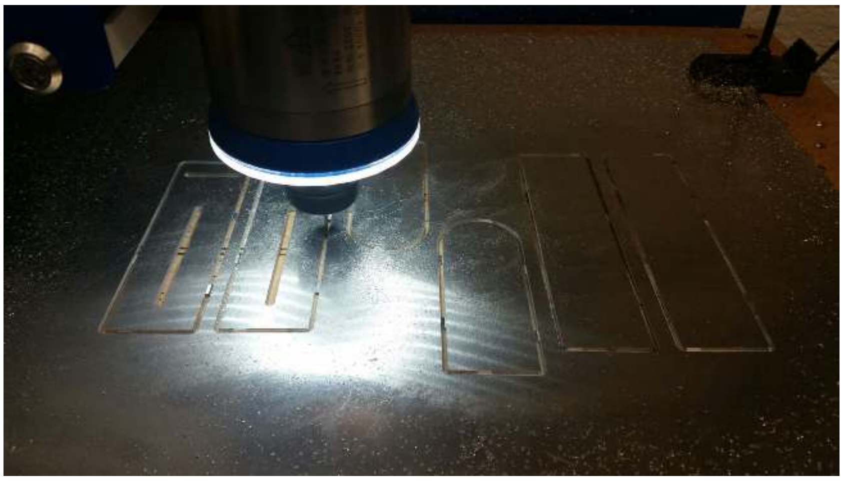


Figure 25: Our aluminum plate cut by the CNC machine at the ATPR

After the cut, we bend them to give them their final shape:

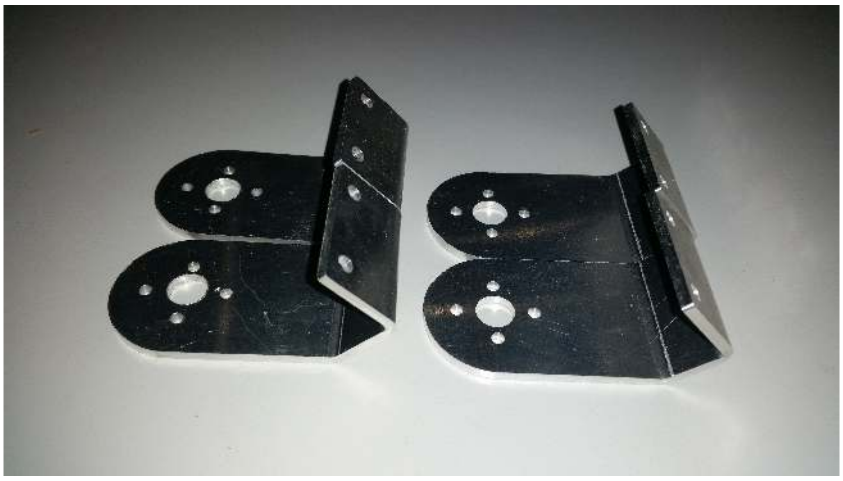


Figure 26: Some of aluminum part when machining is complete

We have used a lot of 3D printing because it is a really fast and reliable way of production if we design and print our parts correctly. The printers in MEA were very impressive but the lack of possibility in the settings for the media was sometimes annoying. We had to clean all the holes or hollows that were automatically filled and sometimes by doing this task we broke our part. The Robopoly's printers that allowed us to print when all those of MEA were busy, allowed us to make pieces just as accurate, without support and colorful.

10 Robot assembly

The assembly of the robot was made as soon as the components arrived. We quickly made our first purchases and cut our MDF with the laser cut machine to assemble the basic frame of the robot after gluing them together two by two to get the right thickness.

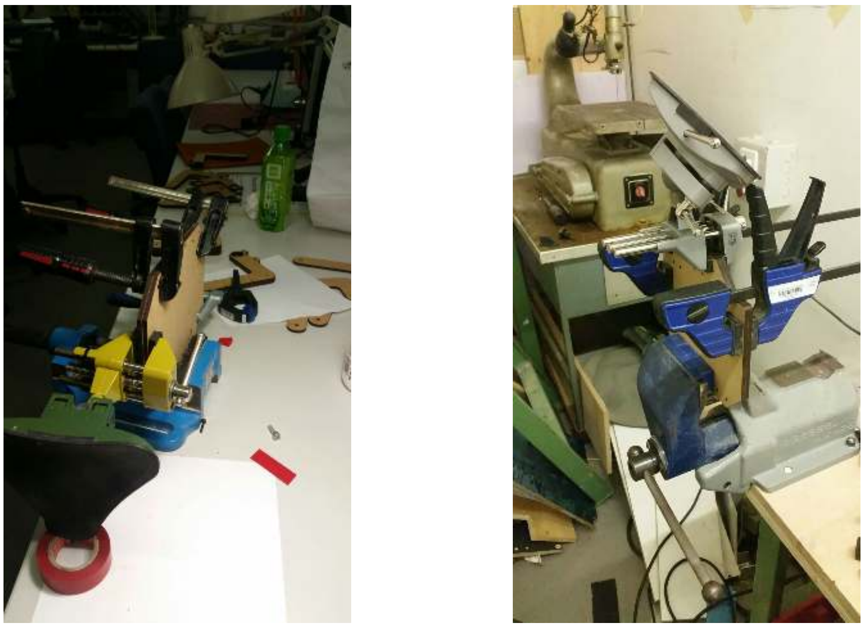


Figure 27: Bonding under stress to prevent the plate from being curved

Then we attached the DC motors and wheels using the necessary connector. After, we were able to integrate the first electronic elements such as the Arduino board, the Raspberry Pi, infrared sensors as well as our motor power board for the motors after having it produced by the ACI and soldering the components on it.

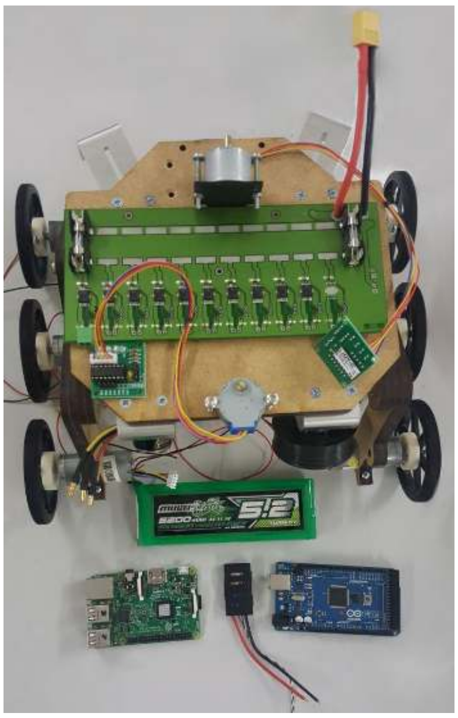


Figure 28: All the electronic components have to take in place on the upper plate

The rods were put in place as well as the wire that connects them to the reel itself mounted on the stepper motor. At the back, the installation of the webcam on its 3D piece that came to fit on the second stepper motor was also set up. The two controllers for the stepper motors were placed on the side of the reel and the Arduino and Raspberry boards were put right behind. Our robot being very small, we had to create levels of electronics so that everything fits on the top plate. We printed 3D support to fit the battery while being able to add over it our PCB to gain space.

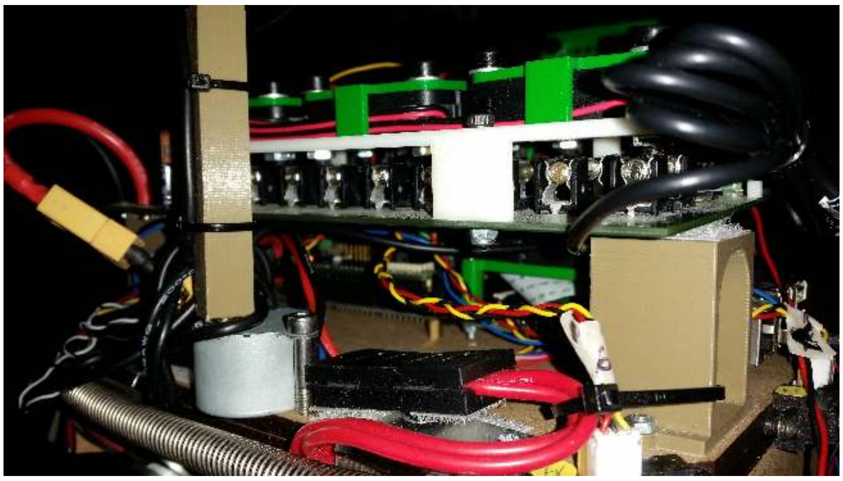


Figure 29: The location of the battery (removed on this picture) is under the motor power board raised by the brown supports

Then, the introduction of the brushless motors was easy and the same for the rotating brooms at the front because we could put everything together before coming to fix it to the top plate. During the progression we made some improvements like we bought fans to improve the cooling. We put three above the motor board control, one above the Raspberry and the last one on the LEDs panel.

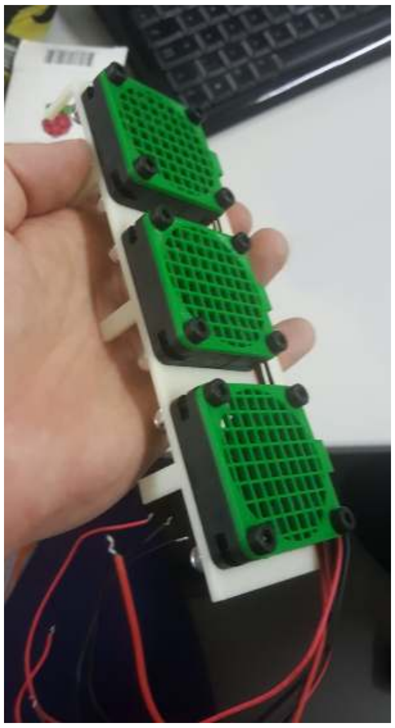


Figure 30: Realization of the cooling support for the H bridge

11 Software part

11.1 Arduino programmation

As mentioned before, the Arduino takes care of all the interactions between environment and the robot.

11.1.1 Obstacle avoidance

Before using the infrared sensors, we tested each individually by placing an obstacle at a distance and recording its intensity to define their distance-intensity curve using Matlab. Here is an example of the curves obtained and we have chosen to make an rational approximation of the type $y = p(x + q)$ where x is the intensity of the sensors and y is the distance in centimeters between the sensor and the obstacles.

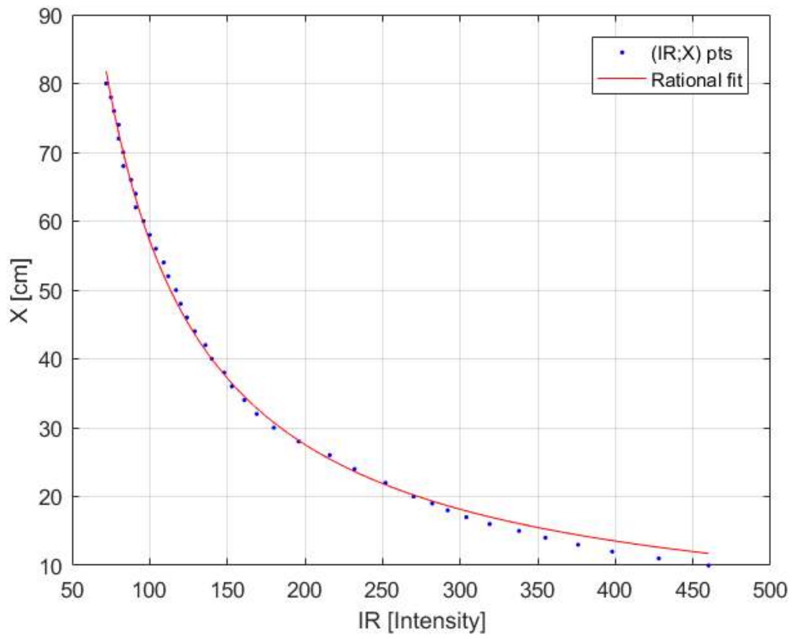


Figure 31: Example of the rational curve fitting between distance and the intensity value

Using these distance-intensity curves, we implemented a Braitenberg vehicle in order to have a smooth obstacle avoidance without stopping the whole program. To do this we used the forward lateral infra reds to know IR2, IR3, IR5 and IR6 from picture 16. The global idea is to directly link the infrared values to the driving motor speed. We followed several steps to achieve this:

Getting the infrared values

To avoid any hit with obstacles we have to check the distance at a quite high frequency. We then use a timer set at $8Hz$ which will read the distance from any objects around the robot. The values are converted thanks to the formula found with Matlab as explain previously.

The normalization of the values

The infrared values are bounded, the maximum value is set to $60cm$ and the minimum is about $25cm$ for the front sensor and $10cm$ for the others. These minimums correspond to the distances below which there is a contact with the obstacles. Now to normalize it we get the difference between the read value and the minimum one all divided by the maximum value. This gives a result between 0 and 1.

Setting the speed function

The motor speed have two component one is constant and correspond to the speed when the robot go straight and a second is the added value to make a rotation. The speed function change this second term. The idea is to have a function that will add a percentage of the differential term which is bounded to not exceed a defined maximal rotation. This mean that our function has to be bounded between -1 and 1 (maximum reduce or increase of the constant speed). The resulting function is:

$$f(x, y) = x - y$$

where x is the product of the normalized values of IR2 and IR3 and y is the product of IR5 and IR6. We can see that if IR2 or IR3 is close to 0, x is close to 0 and then $f(x, y)$ tends to $-y$ which is equal to -1 if there is no obstacles at the left side of the robot. $f(x, y)$ tends to x is IR5 or IR6 tends to 0.

Application of the speed function

Once we have computed the speed function we applied it directly on the speed command of the motors thanks to the setMotorSpeed(leftSpeed, rightSpeed) function:

$$\begin{aligned} \text{leftSpeed} &= \text{constantSpeed} + f(x, y) \cdot \text{maxDeltaSpeed} \\ \text{rightSpeed} &= \text{constantSpeed} - f(x, y) \cdot \text{maxDeltaSpeed} \end{aligned}$$

Thanks to this implementation, the closer to an obstacle our robot is the faster it turns.

Now if any of the infrared get a value equals to its minimum, the state of the robot change to OBSTACLE. In this state all the motors are stops and the obstacle is manage following different case recognize by looking which sensor reach its minimum value:

The obstacle is right in front of the robot

In this case we differentiate two situations. If the IR3 and IR5 are below a defined threshold this mean that we are hitting a corner and the corresponding behaviour is to go straight backward and then turn forward. We turn left or right depending on the result of

$$\text{time} \mod 2$$

where the time is get with the millis function on Arduino language. If only one of the IR3 or IR5 is below this threshold, the robot have the same behaviour but turning left if it is IR3 and right if it is IR5.

The obstacle at the front right or left position

Depending on which of the IR3 or IR5 has reached the minimum distance, the robot will turn backward left if it is IR3 and right if it is IR5.

The obstacle is at the side left or right, or back left or right

The robot will turn forward left or right depending on which infrared reach it minimum.

After avoidance action

The state goes back to the previous state.

11.1.2 Stones avoidance

The first seven minutes the stones are considered as obstacles. We detect them with two switch button placed as shown in picture 12. When they are detected, the state of the robot change to STONE. In this state the robot will go straight backward then turn forward left or right depending on which switch button detected the stones. Afterward the state go back to the previous state.

11.1.3 Reaching the bottle

When the Arduino receive from the Raspberry the coordinate of a bottle, the state of the robot change to BOTTLE. In this state the Arduino stop the Braitenberg vehicle behaviour and manage three cases to reach the bottle before getting the coordinate of it:

Getting the coordinate

The Raspberry send the coordinates of the bottle as a string therefor the first step is to convert it in an integer using the ASCII table.

The bottle is at the left side

In this case the robot will turn to the left until the bottle is right at the front of the robot.

The bottle is at the right side

In this case the robot will turn to the right until the bottle is right at the front of the robot.

The bottle is in the front

When the bottle's x coordinate is at a certain distance from the middle of the picture, 50 pixels around the middle, the robot go straight forward.

11.1.4 Throwing position

When the bottle is grabbed, an ultrasonic sensor inside the robot detect it. At this point the Arduino send to the Raspberry that a bottle is handled and the state of the robot change to ALIGN. In this state the Arduino receive a rotation command from the Raspberry, left or right. The robot will turn in the respective direction.

11.1.5 Fire procedure

Once the robot is align with the recycling area beacon, the Raspberry send the information to the Arduino with the distance from the beacon and the state change to FIRING. In this state the fire procedure is followed:

- All the motors are stopped.
- Rotation speed of the propelling wheels is computed thanks to a function defined by testing different ones.

  $$f(d) = 10d + 80$$

  where d is the distance from the beacon.
- Propelling wheels are activate.
- Guiding wheel is activate.
- Rods are raised which will push the bottle against the guiding wheel and then to the propelling ones.
- All the motors are stopped.

- Rods go back to their initial position.
- A straight forward motion is engage.
- State return to RUN.

11.2 Raspberry Pi

11.2.1 Bottle detection

Setup

In order to control the most as we can the light saturation of the scene we set a lot of parameters like contrast, brightness and exposure. Some other parameters like maximum frame per second and size of the image are also set to avoid too much computation costs.

First filter

To increase the luminescent points on our image we apply a sharp 2D filter of size 3×3. The results are on the figure 32.

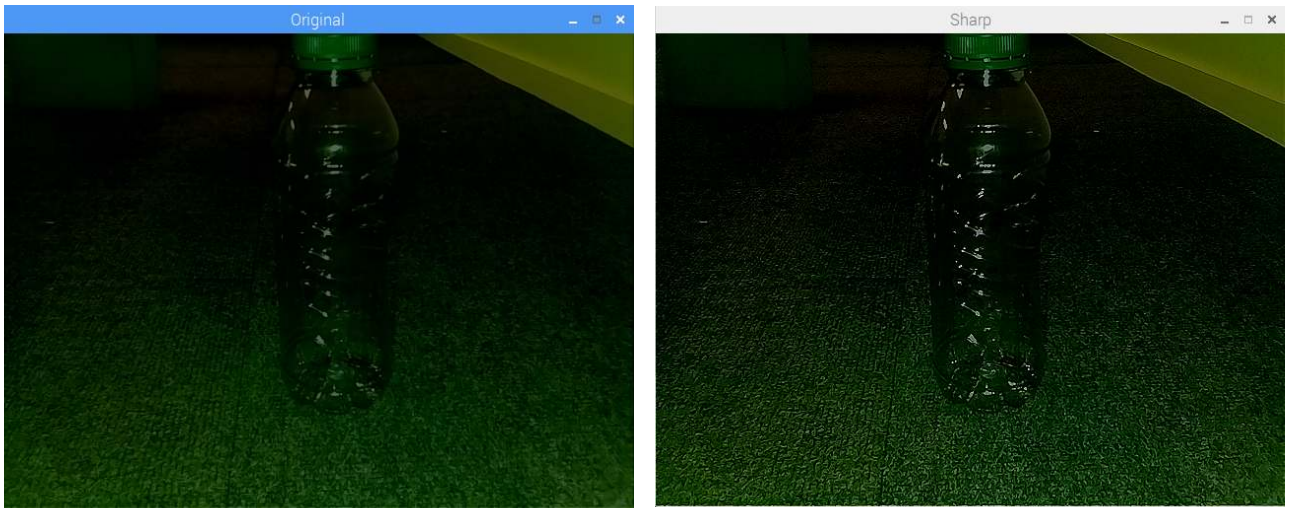


Figure 32: Difference between before and after applying sharp filter

Beacon avoidance

The third step is to be sure that we will not detect beacon LEDs as a bottle thus we take a picture without light, apply an HSV filter to it and extract the mask to apply it on our image taken with light. Since the beacon LEDs are the only thing who emit light in the arena it is quite easy to isolate and detect them on a picture. The result is shown below on figure 33.

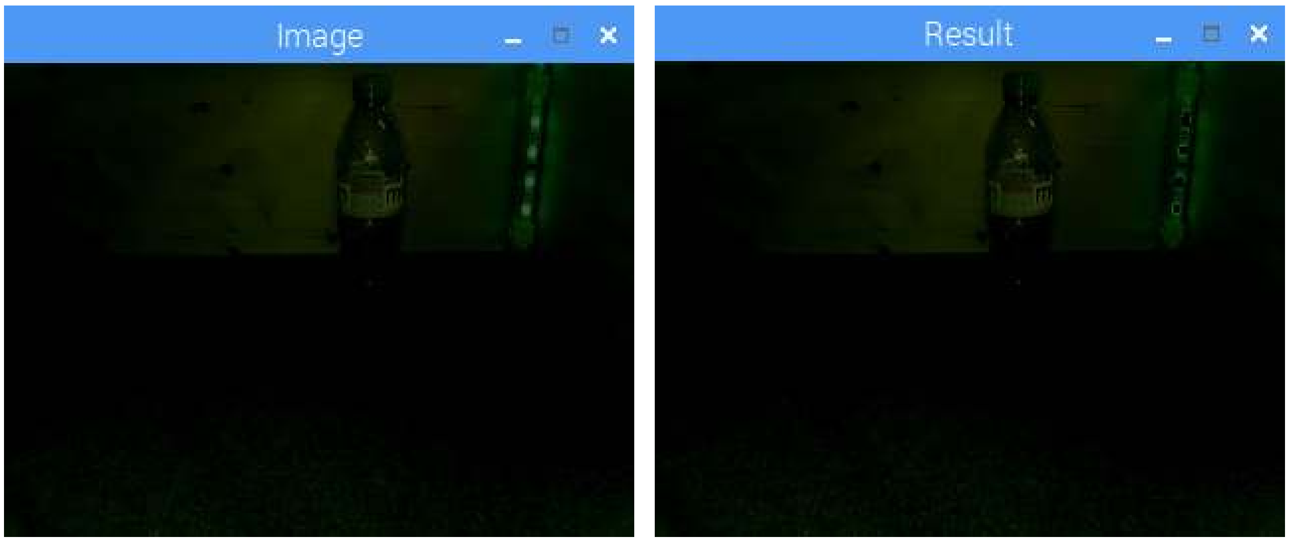


Figure 33: Difference between before and after applying mask to avoid beacon detection as a bottle

First blurring

Since the main part of this bottle detector is taking care of the most luminescent point in the image we have to blur the part which is just in front the robot. Because of the hardware implementation we have a camera that have a certain angle so the scene just in front of the robot is overexposed to the light from our LEDs panel. It is obvious that is not a bottle so we blur the lower half of the picture before searching a bottle on it.

Last steps

The last steps are the main parts of this detector. We just have to find the more saturate point on our picture and we are sure with a high probability that there is a bottle. We improve a little bit this part to avoid some false detection like when we are too near from a wall or rocks. These improvements are made by correctly adjust the parameters in the first step of this section and by comparing two measurements.

1. The mean of the image is not so high when we have a bottle so we keep an eye on this measure to avoid the rocks and the near wall.

2. The maximum luminescent point must be higher than a certain threshold to avoid to detect a bottle when there is not one in front of our robot.

All these steps result on a good bottles detector but too dependent from the light environment. The result can be see in images 34. We detect a bottle on the image only when there is not too much saturated pixels and a certain threshold on the maximum luminescent pixels is passed.

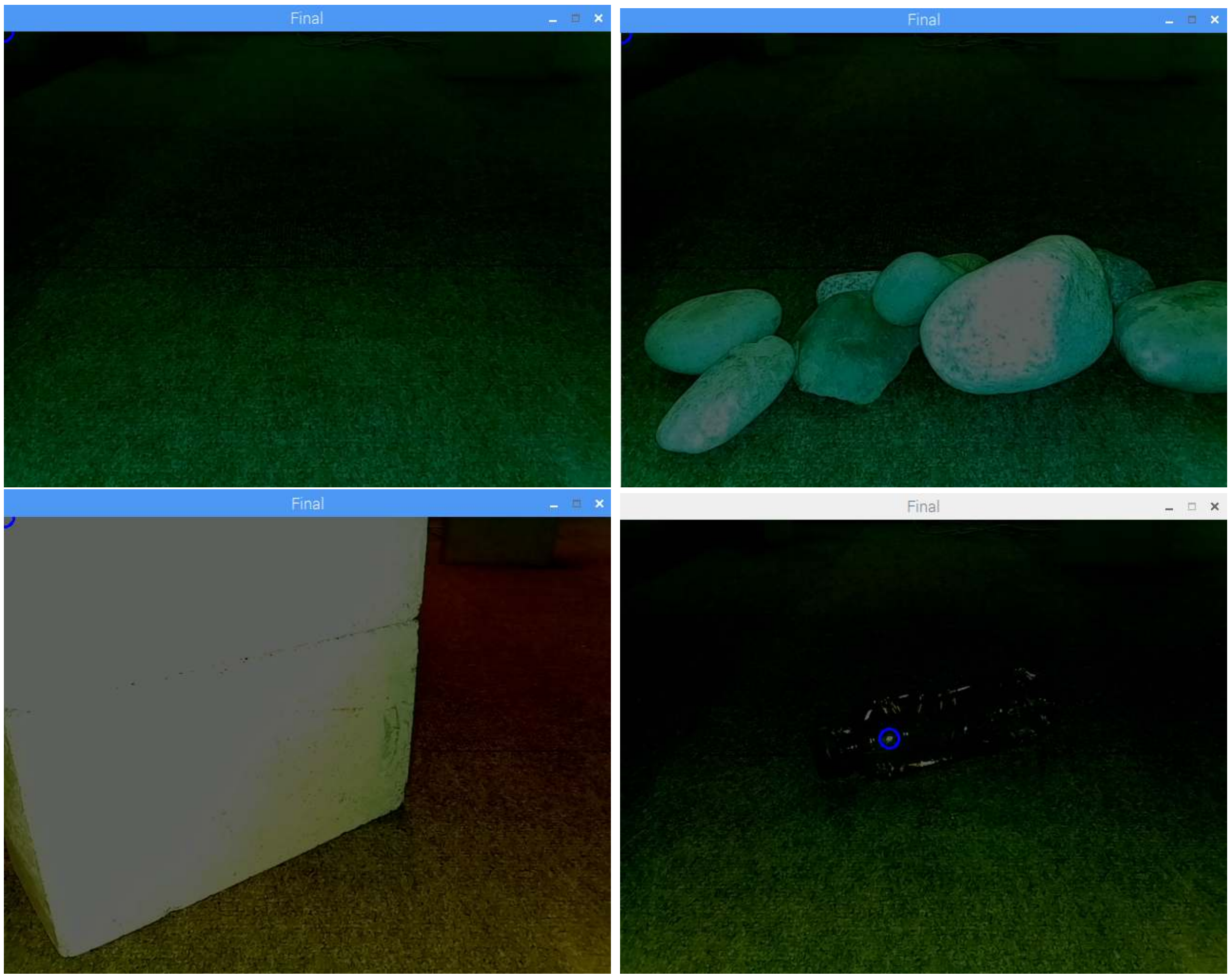


Figure 34: Some results of our bottles detector. Top left: no bottle, top right: rocks, bottom left: bricks and bottom right: bottle

All the code used for the competition is in provided files and the whole testing code implemented during this semester are in a github repository [15].

11.2.2 Beacon tracking

This method is principally done by setting a very low luminosity in the webcam setting and apply a threshold in the HSV domain. After these steps we search the most luminescent point which is the simplest method to obtain the position of the beacon. To know the distance we search the highest point of the beacon LED so we scan our image in the whole height in a certain range in the x direction. With this technique we know the highest point of the beacon LEDs and his height. The result is shown on figure 35.

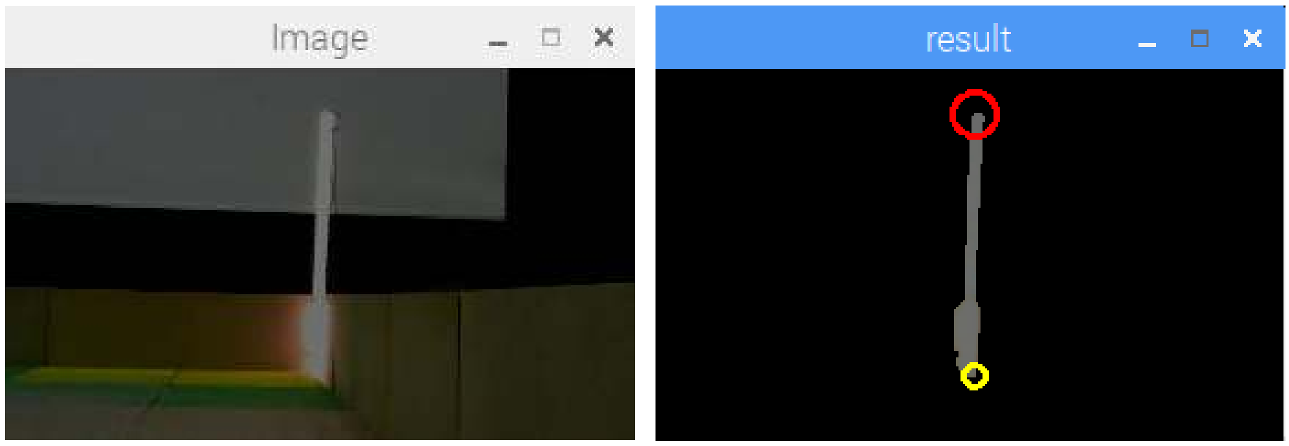


Figure 35: Results of algorithm to find beacon LEDs

In order to know the distance between our robot and the recycling area we do some measurements by taking some pictures and compute the highest point and the height of the beacon. We plot the results on Matlab and fit a curve to find an approximated function thus we are able to know the distance at all the point on the map just by taking a picture. Theses two different functions, one with the highest point and the other one with the whole height, are shown on the figure 36.

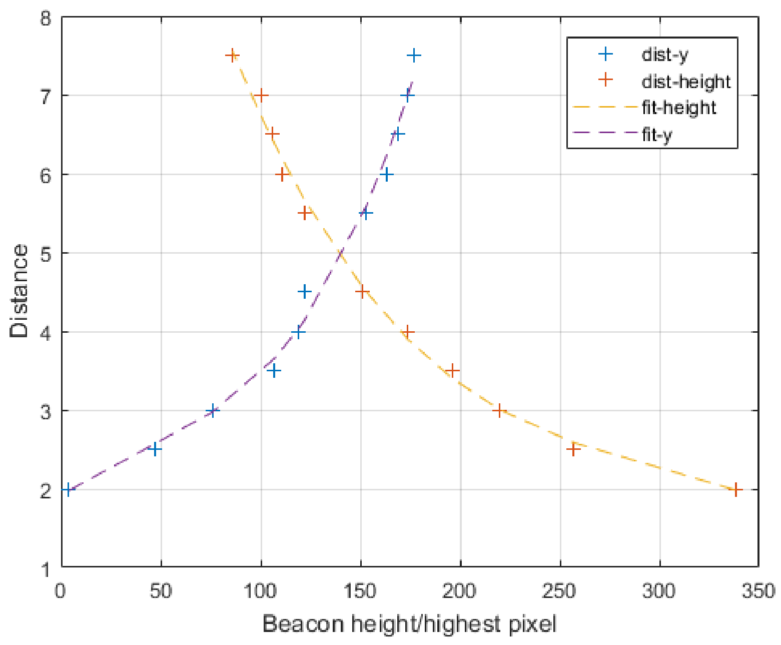


Figure 36: Matlab plot to find a function between LEDs beacon and distance

As we see on this graph the difference of accuracy between only the highest point of the beacon and the whole height is not significant. So we compute the distance by finding the highest point of the beacon and compute the whole height only to know if an obstacle is too near of the robot position. This second computation is needed to avoid throwing a bottle in an obstacle.

11.2.3 UART communication

To synchronize all the electronic components and the two boards we used an UART communication in both direction between the Arduino and Raspberry Pi. It is implemented with a class and an object is created at the beginning of the main and copy to all the function who need it. The Arduino is mainly consider like a slave who follows instructions from the Raspberry Pi to track bottles. It just ignore these orders when it has to avoid an obstacle.

When a bottle is grabbed the Arduino communicate it to the Raspberry Pi. The instructions send from the master switch to the alignment mode where the commands are only turn left or right until the robot is well aligned. Once it does, the Raspberry Pi check if there is an obstacle to near of the robot and sends a forward command if it is the case otherwise it sends the fire order.

Once the bottle shoot to the recycling area the robot takes back is state to the searching bottle and obstacle avoidance.

11.2.4 Multithreading

In order to have all these computation in parallel on the Raspberry Pi we had to implement some different threads. We have three of them which are:

1. Bottle detection
2. Beacon tracking
3. UART RX listening

We have already well described the first two functions and it is obvious that we needed to do one thread for both of them in regard of the computational cost and the importance of these process. The third one is needed to always have the UART RX port open to have a feedback as soon as the Arduino send it to the Raspberry Pi.

With the thread implementation we have to be careful with the speed and the synchronization of each thread because if one crashes the other two will also crash. Both bottle detection and beacon tracking threads need to use the same function to communicate their results to the Arduino. In order to avoid collisions between these two we use a mutex variable to lock and unlock access to the communication function when a thread is already using it.

12 Achievements

12.1 Strategy in the arena

The robot starts in the yellow zone facing the diagonal. After the "Go", he rushes along a random trajectory consisting of straight segments and changes of direction when obstacles are seen. During the first 7 minutes, we decided that our robot would have the right to go on the grass but not on the pebbles thanks to our mechanical sensors which will detect if the ground becomes hunchbacked and it is not allowed to go on the ramp either thanks to the same sensors. During this time, he will cross the arena to avoid any obstacle. When he detects a bottle in his field of vision, he will go to it to pick it up. The pickup procedure has 4 steps:

1. The collection with rotating brooms that will be activated at the detection of the bottle.
2. When the bottle has passed the first step, it will be guided by the guiding wheel until the ultrasonic sensor confirms its presence inside.
3. The rods will rise a little to allow the robot to align with the LED beacon of the recycling zone.
4. The propulsion wheels will activate at the desired speed to travel the right distance and the guiding wheel will activate again for the bottle to be ejected from the robot.

12.2 Final competition

12.2.1 Debriefing

During the public competition, the first five minutes were dedicated to the presentation of our robot, we showed videos made throughout the semester that showed all the features. These last taken independently worked very well but when everything is put together problems arise although we tried to minimize them as much as possible.

During our 10 minutes of competition, the robot managed to detect the presence of bottle but failed to position correctly and he missed the bottle. He continued on his way, changing direction as agreed, but he got stuck in a wall that he should have avoided. After an intervention to direct him into the arena, he continued to move but the bottle detection also had a problem. Unfortunately it was only at the end of the competition that we realized the light were not in presentation mode and since our bottle detection is dependent from the light it causes these troubles. We had permission to restart from the same position after giving him the same code. He started again and detected a bottle. However, during the pickup procedure, he went against a wall and missed the bottle again. We had the permission to help him catch the bottle by putting it inside so that he could finish his ejection algorithm. Unfortunately, this did not happen so we asked to go into demonstration mode. However the Braitenberg vehicle works well. The webcam position also works well and was perfectly aligned with the beacon of the recycling zone.

For the demonstration mode, we wanted to show that the sequence to pick up and shoot worked fine so we loaded this code on the robot. Three pickups and shots were fired from different locations. Two bottles ended in the green zone and the last one in the yellow zone.

12.3 Design review

Considering the design we are globally proud of it as everything works pretty well. However we still had to make some adjustment. We modified the wheels and made them wider as we realized the they were too thin to go in the stone area. The support of the propelling wheels have been modified to because of the variation of the bottles radius. We unscrew a bit the aluminum support in order to permit a bit lateral translation. We added a spring to ensure a compression of the bottles when throwing them.

12.4 Encountered issues

Considering the hardware we did not have any major problems. All the electronics worked well despite the death of the Raspberry Pi camera two days before the public presentation.

For the software the main problem was ourselves. Althought that we planned well the programming, the tiredness made us make mistakes. A final reading of the code made us realize that our robot did not react when there were an obstacle in front because of a missing condition in the manageCorner function.

The team management was also an encountered problem as we did not had all the same skills in robotics neither the same involvement in this project.

13 Conclusion

This semester project was very exciting, interesting, instructive and as expected very exhaustive too. We learned how to manage a project from the conception to the realization which is very satisfying. Thanks to this opportunity to create our own robot we get a knowledge of the common mistakes that could be made, mistakes that we are now able to avoid.

14 Acknowledgements

We would like to thanks all the people who participated in some way to help us. Some particular thanks to Mr. Hauser Simon and Mr. Ramachandran Vivek who followed us all the semester and give us a lot of helpful advice. They gave us the opportunity to asks questions to certain people, were here for our both presentations and were ready to help us whenever we needed.

This competition could not be done without Mr. Crespi Alessandro, all the experts and coaches for their availability, expert's eyes and wise counsels during all the semester.

We can not do these acknowledgements without mentioning all the workshop and makerspace who helped us to do our design. We think about ATPR workshop where we learned how to use the CNC under the supervision of Mr. Schneider Kewin as well as the valuable advice of Mr. Thomas Alfred for his expertise. The ACI workshop helped us a lot by doing our PCB in short time. We can not forgot Made@UC, the makerspace of Renens where we did our lasercutting early in this project and Robopoly, the makerspace of EPFL who gave us some well advices, a lot of space and materials. A lot of 3D printed parts have been done in this space.

This project will not be the same without all the great team who participate this year. Special thanks to all of them for the fair play during this competition, all the well advices and discussions around our robot to solve our problems. We all learn a lot by speaking between us and it is rewarding. Some thanks to Mr. Di Tria Julien who was here during the last two weeks and has been very interested by our projects and always ready to help us if needed.

We also would like to thanks all the people who support us in some way. The people who are coming see us during competition and who are interested during the semester to give us some advice and expertise.

References

[1] Competition committee. *Robot competition rule book.* 2018. URL: https://robot-competition.epfl.ch/files/content/sites/robot-competition/files/documents/rulebook.pdf (visited on 02/26/2018).

[2] Repbaza. *Martian rover.* URL: https://www.thingiverse.com/thing:1318414 (visited on 03/18/2018).

[3] Infineon. *tle9201sg.* URL: https://www.mouser.com/pdfdocs/InfineonTLE9201SGDSv01_00en.PDF (visited on 05/15/2018).

[4] Joseph Chet Redmon. *YOLO learning technique.* URL: https://pjreddie.com/darknet/yolo/ (visited on 03/20/2018).

[5] Robopoly. *Robopoly.* URL: https://robopoly.epfl.ch/.

[6] Lumileds. *MXA7-PW65-H001.* URL: https://docs-emea.rs-online.com/webdocs/146a/0900766b8146a7f2.pdf (visited on 05/20/2018).

[7] DiodesZetex. *DMN2056U-7.* URL: https://docs-emea.rs-online.com/webdocs/1567/0900766b81567f25.pdf (visited on 05/20/2018).

[8] Pololu. *Metal Gearmotor 25Dx54L mm HP 6V.* URL: https://www.pololu.com/product/1575 (visited on 04/02/2018).

[9] MFO. *Prop Drive Series 28-30A 750KV / 140w.* URL: https://hobbyking.com/fr_fr/ntm-prop-drive-series-28-30a-750kv-140w.html?___store=fr_fr (visited on 04/02/2018).

[10] ST. *ULN2003.* URL: http://www.seeedstudio.com/document/pdf/ULN2003%20Datasheet.pdf (visited on 04/02/2018).

[11] Maglev. *MB45101V2-000U-A99.* URL: https://docs-emea.rs-online.com/webdocs/10f0/0900766b810f09b0.pdf (visited on 04/02/2018).

[12] creative commons. *RPi Hardware.* URL: https://elinux.org/RPi_Hardware (visited on 04/02/2018).

[13] Arduino. *ARDUINO MEGA 2560 REV3.* URL: https://store.arduino.cc/usa/arduino-mega-2560-rev3 (visited on 04/02/2018).

[14] Turnigy. *Multistar High Capacity 3S 5200mAh LiPo Battery (XT60).* URL: https://fpvracing.ch/en/batteries/1239-multistar-high-capacity-3s-5200mah-lipo-battery-xt60.html (visited on 04/02/2018).

[15] Munier Louis. *Github repository of all the raspberry pi code.* URL: https://github.com/lmunier/compet_sti (visited on 06/12/2018).

Appendices

Appendix A Primary mathematical analysis

We made different calculations to have an idea of the rotating speed for the motors that throw the bottle. Let's assume the side of the arena is 8 meters, an angle $\alpha = 42°$ to the ground to shoot, an initial position $(x_0, y_0) = (0, 0)$, a initial speed $(v_{x0}, v_{y0}) = (vcos(\alpha), vsin(\alpha))$ with a speed at the exit $v = 25\ m/s$, a gravity $g = 9.81m/s^2$, we can define the position of the bottle after a given time without taking into account the friction.

$$x(t) = vcos(\alpha)t \ and \ y(t) = -0.5gt^2 + vsin(\alpha)t$$

When y(t) = 0 we touch the ground so we need to find t in the formula:

$$0 = -0.5gt^2 + vsin(\alpha)t \Leftrightarrow 0 = (-0.5gt + vsin(\alpha))t$$

It gives only two possibilities: $t = 0$ and $t = 2vsin(\alpha)/g$. The first solution corresponds to the initial position and the second one is to determined the range of shot. So if we introduce this result in $x(t) \Rightarrow d$ we get :

$$d = 2v^2cos(\alpha)sin(\alpha)/g$$

With some trigonometry, we can find the speed needed to travel the desired horizontal distance thanks to the formula: $v = \sqrt{dg/sin(2\alpha)}$ and to throw at d = 11.31m, a speed of 10.56 m/s is required.
To found the right motor that we need, we have to calculate the rotation speed according to the radius of the propulsive wheels .

$$v[m/s] = \omega 2\pi r \Leftrightarrow \omega[rev/s] = \frac{v}{2\pi r} = \frac{10.56}{2\pi * 0.03} = 79[rev/s]$$

We need a motor which can rotate at 4740 rev/s at least.

Appendix B Videos provided

In the following table we listed all the videos that are attached to the report by given the name file and a description.

Table 8: All videos that we made during the whole semester

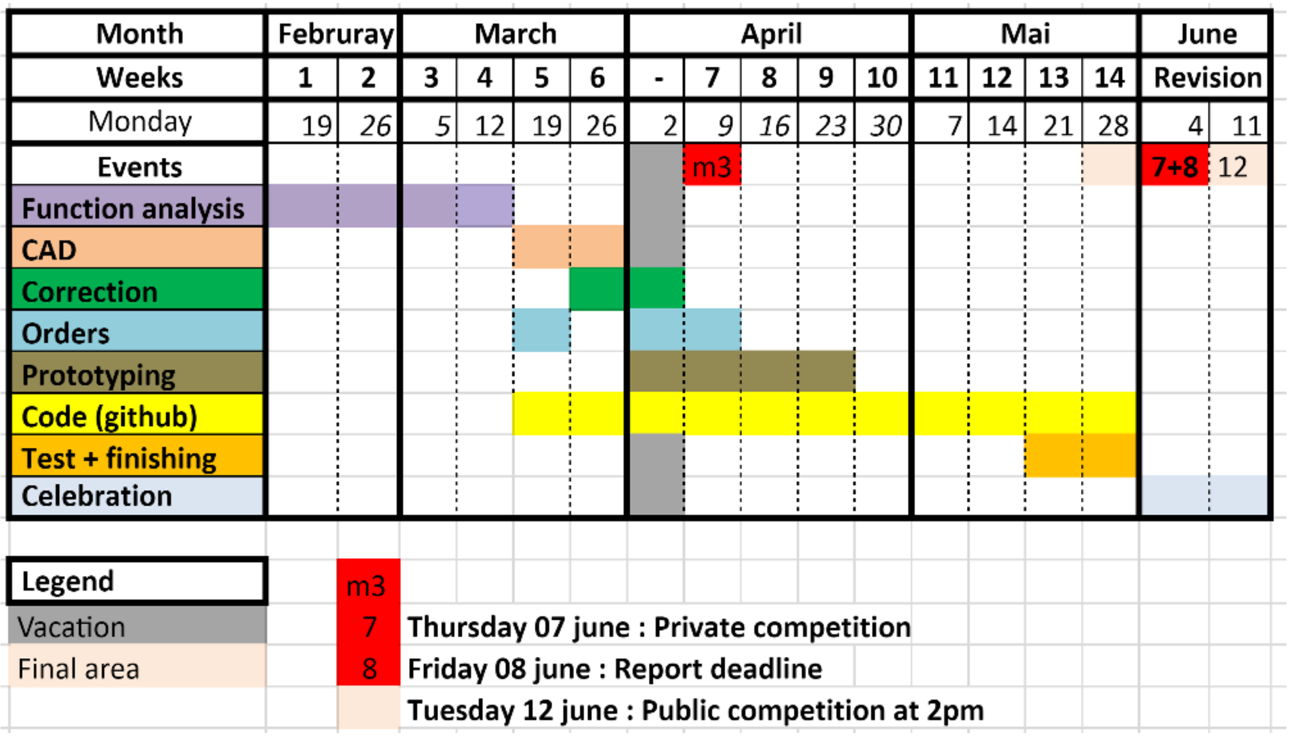
Name file	Description
laser_cut	Small video during the MDF cutting with the laser machine at Made@UC, the makerspace in Renens
infrared_with_PET	Small installation to observe the intensity received by an infrared according to the color of the surface (black or white) and with a PET bottle. We noticed that the infrared can not detect the bottles. The infrared in the test is not the model used on our robot because it did not have the correct range of use.
ultrasound_with_PET	Same idea as above but this time using an ultrasound sensor. It is possible to detect bottles with good accuracy over distance.
ultrasound_as_reference	Short video showing that we can not use the ultrasound sensors to have a constant measure to follow a wall in parallel. The reason why is that the ultrasound wave is conic and it detects any object between the robot and the wall even if the sensor is directed directly at the target.
rover_inspiration	Simple wood setup to show the efficiency of the rover structure.
guiding_rods	Simple setup with the guiding rods to show that a bottle can be guide by them.
haarcascade_face_detection	Before trying to make our dataset of PET bottle, we verify with a face dataset the results of the Haar Cascade method. It is good and powerful.
bottle_detection	Using colors filters and adjustment of parameters we can detect the label or the cap of the bottle.
beacon_localization	In this video, we used a white paper with a thick black line to simulate the beacon so that the webcam follows it.
propulsive_wheel_lego_a-c	Examples of the test setup of the propulsive wheel with lego wheels.
propulsive_wheel_lego_inclined_a/b	Examples of the test setup of the propulsive wheel with lego wheels with a inclined ramp at the output.

Name file	**Description**
propulsive_wheel_flexi_rotation_a/b	To see if the flexible printing wheel does not deformed too much because of the speed.
propulsive_wheel_flexi_setup_a-e	Examples of the test setup of the propulsive wheel in flexible filament.
robot_on_grass	The robot can move on grass.
robot_on_grass_curve_path_a/b	The robot can move on grass and rotate without difficulties.
robot_360_grass_and_normal_a/b	The robot follows a circular path showing it can easily switch from grass to ground and ground to grass.
robot_passed_stones_a-c	The robot can move on stones.
robot_stones_avoidance_a/b	The robot can detect the change of terrain thanks to a mechanical sensor when the wheels rise a little too much. This does not detect passage on the grass because the elevation is too low.
robot_propulsive_wheel_a-c	The video shows the ejection of PET bottles using our propulsive wheels.
robot_pick_throw_a/b	The video shows the robot picking up the bottle with the rotating brooms and throwing the bottle at approximately 7 meters.
robot_move_pick_throw_a/b	The video shows the robot mowing, picking up the bottle with the rotating brooms and throwing the bottle.
robot_all_close_up_back	We watch the movement, the pickup and the jet with a close-up on the back.
robot_all_close_up_front	Same as above with a close-up on the front.
robot_move_pick_insertion_bug	As we can see, sometimes the rotating broom picked up the bottle but it gets inside the robot without the right angle. We added two others parts to avoid this.
robot_obstacle_stop_15cm/50cm	The robot stops at 15cm/50cm from the wall.
robot_obstacles_avoidance	The robot is facing the walls but arrives to continue its path.
robot_shot_3,5/4,0/7,0m_bottom	Shot at different distance with the same rotating speed taking the bottle by the bottom.
robot_shot_3,5/4,5m_cap	Shot at different distance with the same rotating speed taking the bottle by the cap.
robot_max_dist_a-d	Maximal distance reached at the maximal speed supported by the ESC controller.
robot_obst_path_passed	We placed some obstacles to test the obstacles avoidance.
robot_obst_path_stop_failed	As above but we changed the distance to avoid it and it was not so good anymore.

Name file	Description
robot_obst_path_stones_fail_a/b	Same as above but the robot finished in the stone and gets stuck.
robot_corner_return	We put the robot in the situation when it goes to a corner and he manages it well. The passage of the grass on the ground does not encounter any problem and it dodges the obstacle and finally it will stop before making a check to Alexander.
robot_all_ultrasound_check	This video shows that the robot still move until the bottle is detected inside the body by the ultrasound sensors and after the process follows.
wheels_extension_stone	Despite our improvement to avoid getting stuck on pebbles, it can still happen.
wheels_extension_stone_avoidance	Despite the use of the wheel extension, our main concern in this video was the maneuvering back that was not long enough and made us return very quickly to the pebbles.
final_test_a/b	The two last video are showing the behaviour for the competition. In video A, he manages to do locomotion, detection, collection and start the throwing process. Unfortunately he loses the visual tracking because Julien was between the robot and the beacon so the program could not finish properly because the camera was trying to find the beacon while trying to line up. In video B, he did the same thing until the throwing and did it.But we can see a problem with the obstacle avoidance at the end, it does not work anymore. The robot was probably stuck in a state and did not get out.
team_3_presentation	It is the video that we present during the public competition.

Appendix C Gantt chart

Here we present the Gantt chart that we try to follow during the whole semester:



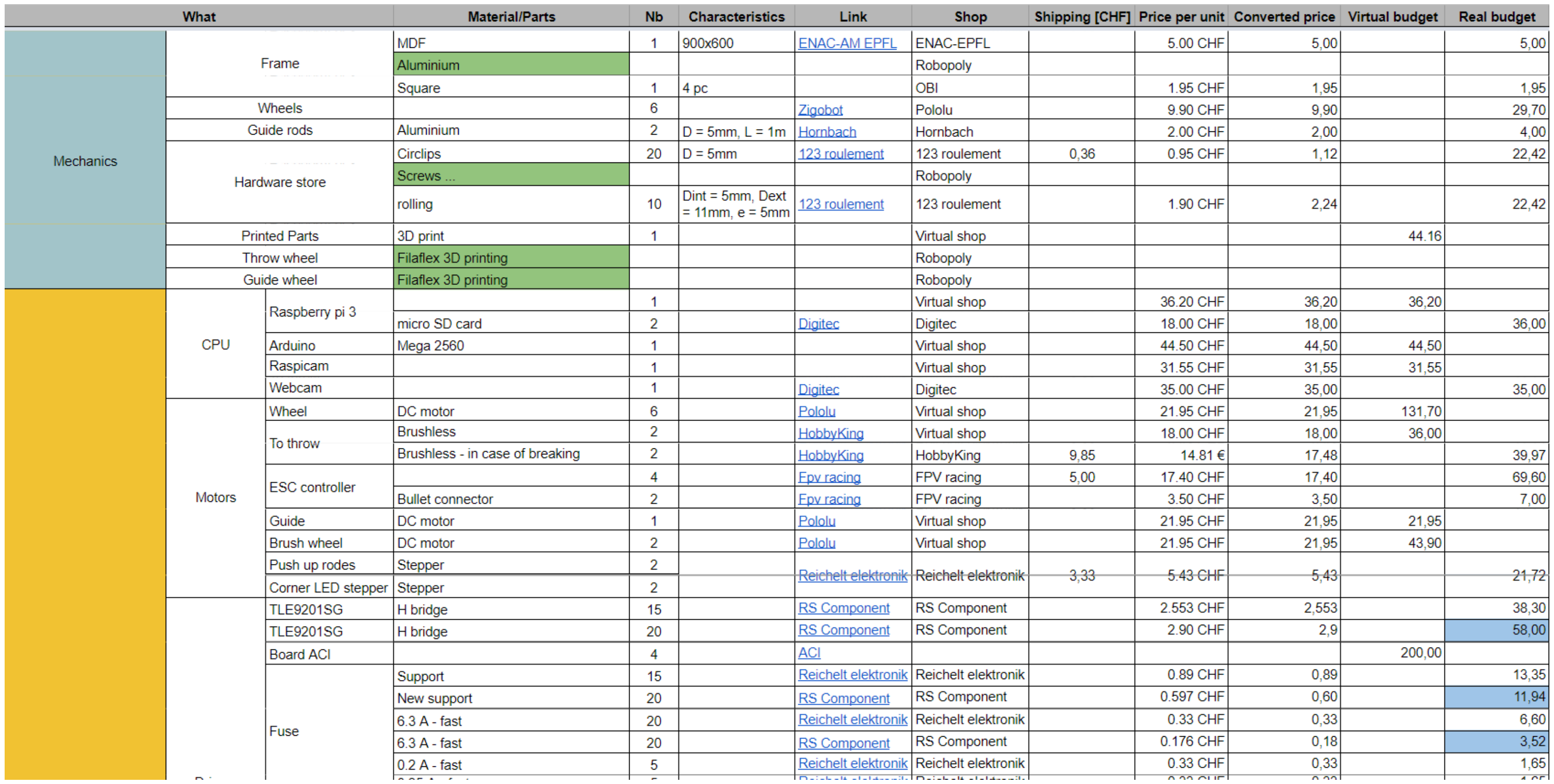
Month	Februray		March				April					Mai				June	
Weeks	1	2	3	4	5	6	-	7	8	9	10	11	12	13	14	Revision	
Monday	19	*26*	*5*	12	19	26	2	*9*	*16*	*23*	*30*	7	14	21	28	4	11
Events							▒	m3							░	7+8	12
Function analysis	■	■	■	■			▒										
CAD					■	■	▒										
Correction						■	■										
Orders					■		■	■									
Prototyping							■	■	■	■							
Code (github)					■	■	■	■	■	■	■	■	■	■	■		
Test + finishing							▒							■	■		
Celebration							▒									■	■

Legend		
Vacation ▒	m3	
Final area ░	7	Thursday 07 june : Private competition
	8	Friday 08 june : Report deadline
	░	Tuesday 12 june : Public competition at 2pm

Figure 37: Gantt chart

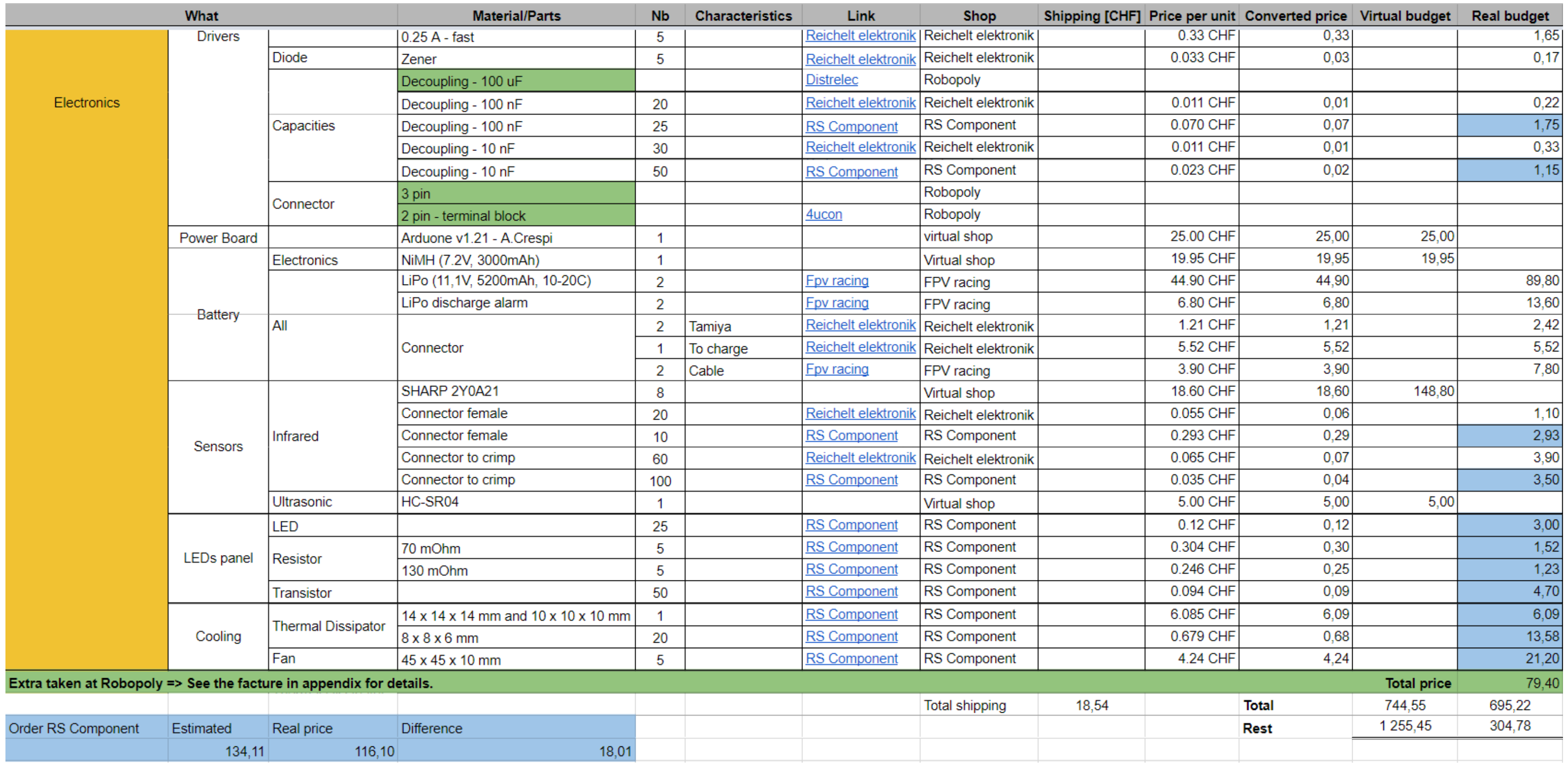
Appendix D Budget management

Here we present the budget management during all the whole semester:



What			Material/Parts	Nb	Characteristics	Link	Shop	Shipping [CHF]	Price per unit	Converted price	Virtual budget	Real budget
Mechanics	Frame		MDF	1	900x600	ENAC-AM EPFL	ENAC-EPFL		5.00 CHF	5,00		5,00
			Aluminium				Robopoly					
			Square	1	4 pc		OBI		1.95 CHF	1,95		1,95
	Wheels			6		Zigobot	Pololu		9.90 CHF	9,90		29,70
	Guide rods		Aluminium	2	D = 5mm, L = 1m	Hornbach	Hornbach		2.00 CHF	2,00		4,00
	Hardware store		Circlips	20	D = 5mm	123 roulement	123 roulement	0,36	0.95 CHF	1,12		22,42
			Screws ...				Robopoly					
			rolling	10	Dint = 5mm, Dext = 11mm, e = 5mm	123 roulement	123 roulement		1.90 CHF	2,24		22,42
	Printed Parts		3D print	1			Virtual shop				44.16	
	Throw wheel		Filaflex 3D printing				Robopoly					
	Guide wheel		Filaflex 3D printing				Robopoly					
	CPU	Raspberry pi 3		1			Virtual shop		36.20 CHF	36,20	36,20	
			micro SD card	2		Digitec	Digitec		18.00 CHF	18,00		36,00
		Arduino	Mega 2560	1			Virtual shop		44.50 CHF	44,50	44,50	
		Raspicam		1			Virtual shop		31.55 CHF	31,55	31,55	
		Webcam		1		Digitec	Digitec		35.00 CHF	35,00		35,00
	Motors	Wheel	DC motor	6		Pololu	Virtual shop		21.95 CHF	21,95	131,70	
		To throw	Brushless	2		HobbyKing	Virtual shop		18.00 CHF	18,00	36,00	
			Brushless - in case of breaking	2		HobbyKing	HobbyKing	9,85	14.81 €	17,48		39,97
		ESC controller		4		Fpv racing	FPV racing	5,00	17.40 CHF	17,40		69,60
			Bullet connector	2		Fpv racing	FPV racing		3.50 CHF	3,50		7,00
		Guide	DC motor	1		Pololu	Virtual shop		21.95 CHF	21,95	21,95	
		Brush wheel	DC motor	2		Pololu	Virtual shop		21.95 CHF	21,95	43,90	
		Push up rodes	Stepper	2		~~Reichelt elektronik~~	~~Reichelt elektronik~~	~~3,33~~	~~5.43 CHF~~	~~5,43~~		~~21,72~~
		Corner LED stepper	Stepper	2								
		TLE9201SG	H bridge	15		RS Component	RS Component		2.553 CHF	2,553		38,30
		TLE9201SG	H bridge	20		RS Component	RS Component		2.90 CHF	2,9		58,00
		Board ACI		4		ACI					200,00	
		Fuse	Support	15		Reichelt elektronik	Reichelt elektronik		0.89 CHF	0,89		13,35
			New support	20		RS Component	RS Component		0.597 CHF	0,60		11,94
			6.3 A - fast	20		Reichelt elektronik	Reichelt elektronik		0.33 CHF	0,33		6,60
			6.3 A - fast	20		RS Component	RS Component		0.176 CHF	0,18		3,52
			0.2 A - fast	5		Reichelt elektronik	Reichelt elektronik		0.33 CHF	0,33		1,65

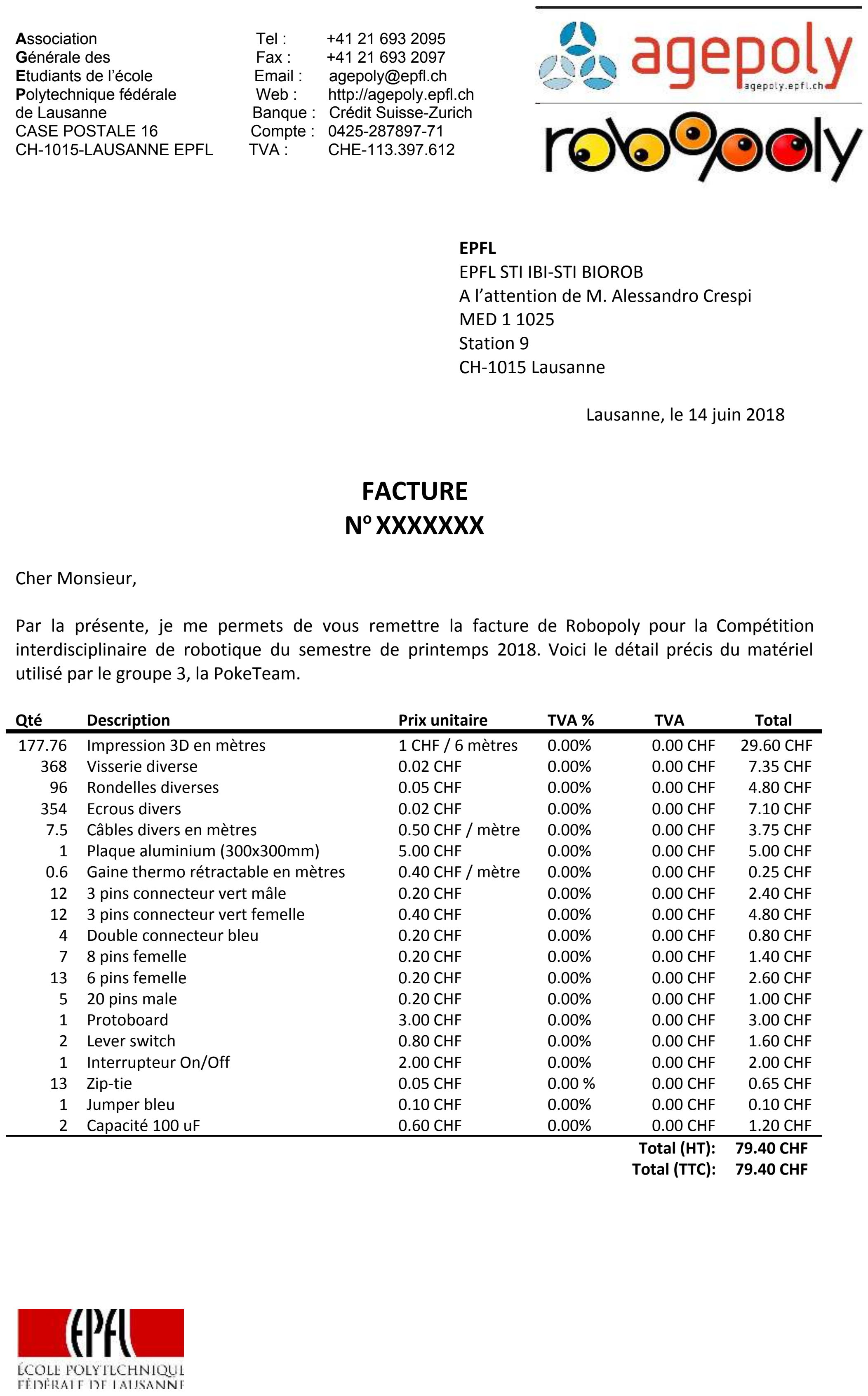
Figure 38: Budget management for real and virtual budget (part 1)



What			Material/Parts	Nb	Characteristics	Link	Shop	Shipping [CHF]	Price per unit	Converted price	Virtual budget	Real budget
Electronics	Drivers		0.25 A - fast	5		Reichelt elektronik	Reichelt elektronik		0.33 CHF	0,33		1,65
		Diode	Zener	5		Reichelt elektronik	Reichelt elektronik		0.033 CHF	0,03		0,17
		Capacities	Decoupling - 100 uF			Distrelec	Robopoly					
			Decoupling - 100 nF	20		Reichelt elektronik	Reichelt elektronik		0.011 CHF	0,01		0,22
			Decoupling - 100 nF	25		RS Component	RS Component		0.070 CHF	0,07		1,75
			Decoupling - 10 nF	30		Reichelt elektronik	Reichelt elektronik		0.011 CHF	0,01		0,33
			Decoupling - 10 nF	50		RS Component	RS Component		0.023 CHF	0,02		1,15
		Connector	3 pin				Robopoly					
			2 pin - terminal block			4ucon	Robopoly					
	Power Board		Arduone v1.21 - A.Crespi	1			virtual shop		25.00 CHF	25,00	25,00	
	Battery	Electronics	NiMH (7.2V, 3000mAh)	1			Virtual shop		19.95 CHF	19,95	19,95	
			LiPo (11,1V, 5200mAh, 10-20C)	2		Fpv racing	FPV racing		44.90 CHF	44,90		89,80
			LiPo discharge alarm	2		Fpv racing	FPV racing		6.80 CHF	6,80		13,60
		All	Connector	2	Tamiya	Reichelt elektronik	Reichelt elektronik		1.21 CHF	1,21		2,42
				1	To charge	Reichelt elektronik	Reichelt elektronik		5.52 CHF	5,52		5,52
				2	Cable	Fpv racing	FPV racing		3.90 CHF	3,90		7,80
	Sensors	Infrared	SHARP 2Y0A21	8			Virtual shop		18.60 CHF	18,60	148,80	
			Connector female	20		Reichelt elektronik	Reichelt elektronik		0.055 CHF	0,06		1,10
			Connector female	10		RS Component	RS Component		0.293 CHF	0,29		2,93
			Connector to crimp	60		Reichelt elektronik	Reichelt elektronik		0.065 CHF	0,07		3,90
			Connector to crimp	100		RS Component	RS Component		0.035 CHF	0,04		3,50
		Ultrasonic	HC-SR04	1			Virtual shop		5.00 CHF	5,00	5,00	
	LEDs panel	LED		25		RS Component	RS Component		0.12 CHF	0,12		3,00
		Resistor	70 mOhm	5		RS Component	RS Component		0.304 CHF	0,30		1,52
			130 mOhm	5		RS Component	RS Component		0.246 CHF	0,25		1,23
		Transistor		50		RS Component	RS Component		0.094 CHF	0,09		4,70
	Cooling	Thermal Dissipator	14 x 14 x 14 mm and 10 x 10 x 10 mm	1		RS Component	RS Component		6.085 CHF	6,09		6,09
			8 x 8 x 6 mm	20		RS Component	RS Component		0.679 CHF	0,68		13,58
		Fan	45 x 45 x 10 mm	5		RS Component	RS Component		4.24 CHF	4,24		21,20
Extra taken at Robopoly => See the facture in appendix for details.											**Total price**	79,40
							Total shipping	18,54		**Total**	744,55	695,22
Order RS Component	Estimated	Real price	Difference							**Rest**	1 255,45	304,78
	134,11	116,10	18,01									

Figure 39: Budget management for real and virtual budget (part 2)

We also present the invoice of Robopoly for the components taken:



Association **G**énérale des **E**tudiants de l'école **P**olytechnique fédérale de Lausanne	Tel :	+41 21 693 2095	
	Fax :	+41 21 693 2097	
	Email :	agepoly@epfl.ch	
	Web :	http://agepoly.epfl.ch	
	Banque :	Crédit Suisse-Zurich	
CASE POSTALE 16	Compte :	0425-287897-71	
CH-1015-LAUSANNE EPFL	TVA :	CHE-113.397.612	

agepoly
agepoly.epfl.ch
robopoly

EPFL
EPFL STI IBI-STI BIOROB
A l'attention de M. Alessandro Crespi
MED 1 1025
Station 9
CH-1015 Lausanne

Lausanne, le 14 juin 2018

FACTURE
N° XXXXXXX

Cher Monsieur,

Par la présente, je me permets de vous remettre la facture de Robopoly pour la Compétition interdisciplinaire de robotique du semestre de printemps 2018. Voici le détail précis du matériel utilisé par le groupe 3, la PokeTeam.

Qté	Description	Prix unitaire	TVA %	TVA	Total
177.76	Impression 3D en mètres	1 CHF / 6 mètres	0.00%	0.00 CHF	29.60 CHF
368	Visserie diverse	0.02 CHF	0.00%	0.00 CHF	7.35 CHF
96	Rondelles diverses	0.05 CHF	0.00%	0.00 CHF	4.80 CHF
354	Ecrous divers	0.02 CHF	0.00%	0.00 CHF	7.10 CHF
7.5	Câbles divers en mètres	0.50 CHF / mètre	0.00%	0.00 CHF	3.75 CHF
1	Plaque aluminium (300x300mm)	5.00 CHF	0.00%	0.00 CHF	5.00 CHF
0.6	Gaine thermo rétractable en mètres	0.40 CHF / mètre	0.00%	0.00 CHF	0.25 CHF
12	3 pins connecteur vert mâle	0.20 CHF	0.00%	0.00 CHF	2.40 CHF
12	3 pins connecteur vert femelle	0.40 CHF	0.00%	0.00 CHF	4.80 CHF
4	Double connecteur bleu	0.20 CHF	0.00%	0.00 CHF	0.80 CHF
7	8 pins femelle	0.20 CHF	0.00%	0.00 CHF	1.40 CHF
13	6 pins femelle	0.20 CHF	0.00%	0.00 CHF	2.60 CHF
5	20 pins male	0.20 CHF	0.00%	0.00 CHF	1.00 CHF
1	Protoboard	3.00 CHF	0.00%	0.00 CHF	3.00 CHF
2	Lever switch	0.80 CHF	0.00%	0.00 CHF	1.60 CHF
1	Interrupteur On/Off	2.00 CHF	0.00%	0.00 CHF	2.00 CHF
13	Zip-tie	0.05 CHF	0.00 %	0.00 CHF	0.65 CHF
1	Jumper bleu	0.10 CHF	0.00%	0.00 CHF	0.10 CHF
2	Capacité 100 uF	0.60 CHF	0.00%	0.00 CHF	1.20 CHF
				Total (HT):	**79.40 CHF**
				Total (TTC):	**79.40 CHF**

EPFL
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE