

RoBorregos Container Handler Robot - Team Description Paper

Rodríguez A., Sol I., Alvarez J., Virgen S. & Uvalle J.

Instituto Tecnológico y de Estudios Superiores de Monterrey (ITESM)

Abstract— The following article presents the strategy proposed by the RoBorregos team to overcome the Latin American Robotics Competition Open Challenge 2018. Through a detailed explanation and diverse images are elucidated: a description of the aimed algorithm, the mechanical and electronic designs, a brief allusion of the microcontroller communication, the employed signal filters, as well as the implemented navigation and computer vision techniques.

I. INTRODUCTION

As a result from the current industrialization, enterprises demand transporting tons of material everyday. As a solution, several industries manage to bring cargo and transportation services into the market. However, this tasks are commonly provided by manual mechanisms that generate error rate and put the lives of workers at risk. Thought the develop of this Container Handler Robot, the possibility to aim into an autonomous operation rather than manually is brought. By simulating a real world situation, the robot must be able to grab, transport and deposit containers in its proper zone, depending on the color of the container. In order to accomplish this duty, an algorithm is proposed, as well as certain technical characteristics in hardware and software.

II. STRATEGY

Our strategy is based on the rules and specifications of this year's challenge. As the robot accomplishes the different tasks, such as grabbing a container, data is stored to make upcoming decisions. This approach comprises two sections: "container selection" and "return path". In addition to the logic, round time is measured at every second; if logic has not ended but remaining time is less than 30 seconds, the robot will automatically lead to the finish button and press it.

A. Container selection

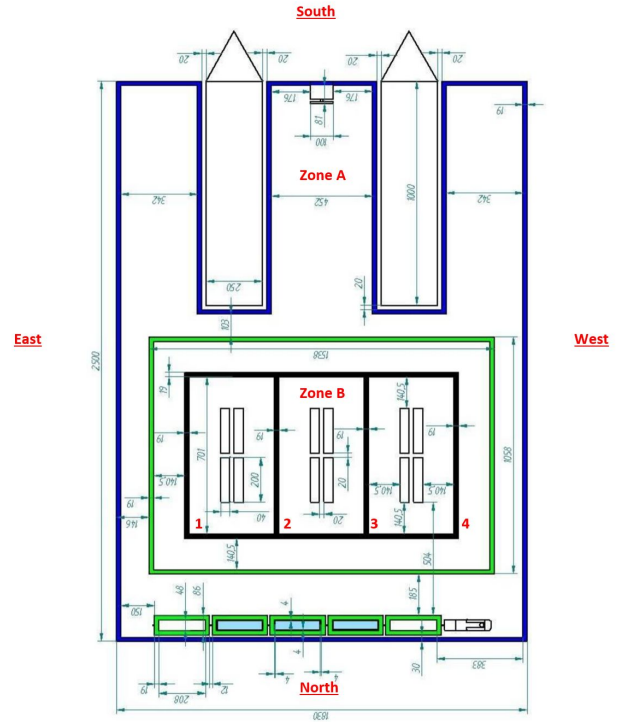


Figure 1. Orientation and symbolism

As a starting orientation, the robot is placed pointing towards the train which we consider it as north, as shown in Figure 1. Then, since our coupling mechanism is placed at the right side of the robot, our priority at the beginning is picking up blue containers from the stacks located at the right side of the robot and then start making decisions regarding their priorities and orientation. A color is considered as available if there is at least one container of that color at the top of any of the stacks on the right side of the robot. The decision criteria algorithm is described in Figure 2:

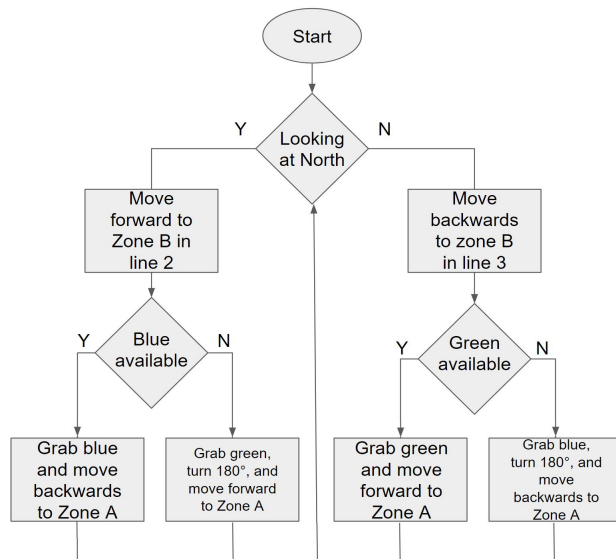


Figure 2. Container selection algorithm

B. Return path

Once the container has been chosen and secured by the mechanism, it is time to proceed into the proper zone and deposit the container. However, since time is crucial, the shortest path must be traced depending on several variables such as container color and current robot spot. For better understanding, the specific strategy is detailed in Figure 3:

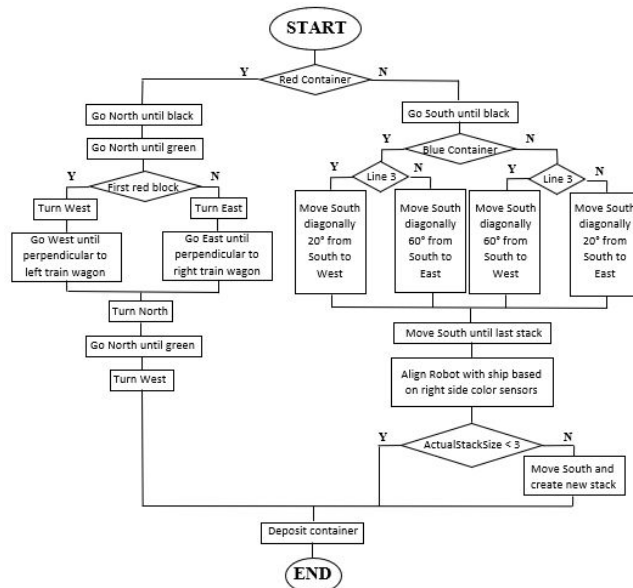


Figure 3. Return path algorithm

III. MECHANICS

A. The coupling mechanism

The goal of our robot is to detect, secure and transport different containers in the least amount of time. Since all containers have the same dimensions and weight, we focused on designing a mechanism capable of holding and manipulating the containers in the directions that best suit

our strategy. It must be capable of translating vertically and extend the container holder an additional 5 cm along its longitudinal axis. The front side of the system is set facing the right side of the robot.

The magnetic properties of structural steel allow us to use neodymium magnets to hold the containers in place. As depicted in Figure 4, the mechanism consists of two magnets attached to a part in one of the extremes of a rack, which is supported by a piece that serves as guide and holds the pinion's Servo Motor in place. Once the rack stretches, the magnets follow a linear path until they encounter the container support pieces (container holder), that is the intended position to attach the container. If necessary, the rack can be stretched an additional distance, pushing the container supports ahead; if retracted, the supports will return to their original position with the help of a tension spring. Four vertical linear guides hold the mechanism in place at the desired height.

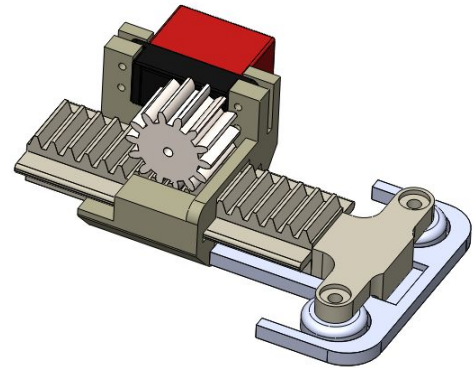


Figure 4. Coupling System

B. Material

The structure and mechanical components of the robot were designed in Dassault Systemes' SolidWorks 2016 version. It consists mostly of Laser Cut Acrylic and 3D printed pieces of Polylactic Acid (PLA) and PetG filament. We have specifically designed sensor holders (e.g., distance sensors, cameras, gyroscope) and support structures to be easily printed.

Bases and flat surfaces are made of acrylic or solid polycarbonate sheets, the use of either material depends on the stress conditions presented on the piece. Due to its mechanical properties, polycarbonate is the material of choice for surfaces subjected to a high level of mechanical stress. The joints and unions between parts are held by nuts and bolts of 3mm in diameter and superglue in places where it's needed.

To facilitate the movement of the robot and easily perform complex movements we chose four Vex 4in Mecanum Wheels and four 12V Pololu Gearmotors M.P. with 48 CPR Encoders placed in a square position. The task of coupling the motors and wheels will be taken by solid aluminum parts machined on a Lathe.

The axis for vertical translation (linear guides) of the coupling mechanism is made up of 6mm diameter aluminum rods and linear bearings of the same internal radius

connected to the main structure. The pinion in the coupling mechanism is actuated by a Power HD Servo Motor of 20Kg torque. 2 circular neodymium magnets fix the containers (Structural steel) to the mechanism, the magnitude of their magnetic properties was chosen depending on the attraction force between the magnets and the container.

IV. ON-BOARD PROCESSING

A. Hardware and Software Specs

The hardware used in the robot is analyzed in this section. The Raspberry Pi Camera Module v2 was chosen because of its size, the overall dimensions of the camera are small, so that we don't have issues with the movement and placement of the robot. To control the camera we decided to use a Raspberry Pi Zero, which is a single-board computer with a 1GHz ARM11 processor and 512 MB RAM. We selected this device because of its great size to processing power ratio. Raspbian, OpenCV and Python are used to take the images from the arena, process the data, find the containers and determine colors and distances in the field. The Atmel Atmega2560 was chosen as our main microcontroller because of the quantity and selection of ports, ranging from a generous amount of general purpose digital pins to i2C, SPI and even several serial ports, all of this is important since this microcontroller handles the motors, sensors and servos.

The design and construction of the PCB board is extremely important for any robot, having a reliable electronic system ensures overall stability. There's always a common point of failure when a prefabricated board (in this case and arduino) is used in another system, and that is the connection between said system and the board in question, the common practice is to use male header pins a connection medium, but this often causes more problems than it solves, a different approach was followed in order to avoid this possible problem.

In essence, an Arduino is simply an atmel microcontroller with a special bootloader flashed into it. Additionally, there are helper circuits embedded into the board, such as usb to serial converters, voltage regulators and visual indicators (LEDs) whose only purpose is to create an easier development experience. These helper circuits occupy a lot of board real estate and make the design process a little cumbersome. Using just the microcontroller and building only the necessary helper circuits into our main PCB makes designing easy, since the microcontroller occupies only a fraction of the space an Arduino mega does. Additionally, and more importantly, the IC gets directly soldered to the PCB, creating a secure mechanical and electrical connection and giving us the ability to choose exactly which and how many pins to use, thus saving incredible amounts of space.

“Clean and efficient power makes a system stable” (Mihai, 2014). That's why the XL4016E1 by XLSEMI was chosen as our power management IC. It provides a framework to create a high efficiency “buck” switching power supply that can handle up to an 8A load making it

perfect to drive our DC motors and servos, the simple-design nature of this IC allows it to be embedded in the main board, once again saving space and avoiding potential problems caused by connections to external regulators.

The Adafruit motor shield is one of the best ways to drive DC motors, it is capable of controlling up to 4 of them at 1.8A continuously, it is a battle tested solution and a great option for any robot, but it's form factor makes it hard to integrate into our “barebones” system. This shield uses the TB6612FNG by Toshiba, which is a dual H bridge in a small SMD form factor, by using this IC directly on our board we get the reliability of the adafruit shield and the benefits of using a plain integrated circuit.

This whole sections have been about saving space and reliability, ultimately the goal is to create a stable electronic system that occupies the least amount of space possible, providing more clearance for the mechanical system which in the end is the one that will allow us to score points. A prototype board was designed as a proof of concept and is presented below in Figure 5 and Figure 6.

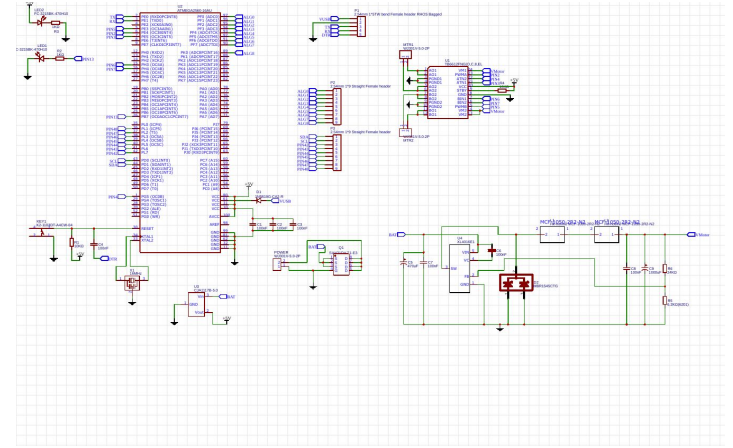


Figure 5. Electronic diagram of the proof of concept board.

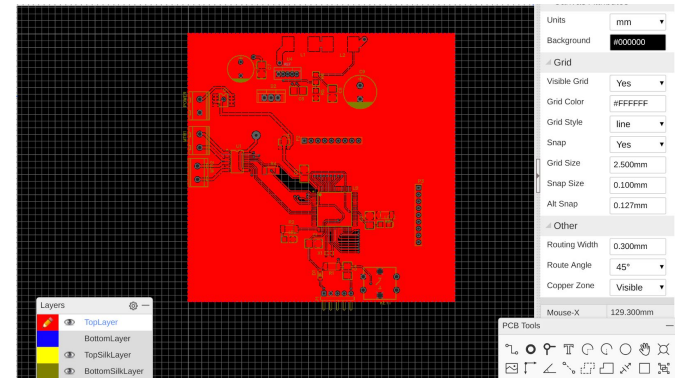


Figure 6. Top layer of the pcb board.

B. Microcontroller Communication

As we stated before, the first action taken by the robot is going to the containers zone. Once the robot is correctly

located, the Raspberry Pi takes a photo and analyses it to determine the top containers' colors and establish the following steps via serial.

Both, microcontroller and Raspberry Pi, listen at the serial input and take different actions regarding the received data. As soon as the microcontroller finishes its required action, as could be going to the container zone, it sends a flag to the Raspberry Pi using the serial communication to request taking a photo. Finally, when the Raspberry analyzes the taken picture, it sends a specific character describing the situation, such as 'B' if the two top containers are blue.

Raspberry PI

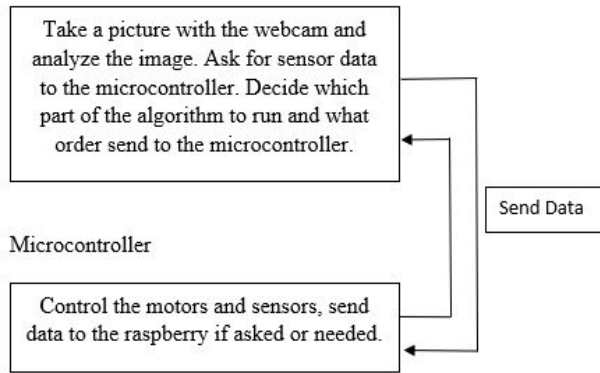


Figure 7. Interaction between the two main cores of the robot

C. Signal Filtering

A distance measuring Sharp sensor GP2Y0A02YK0F, an Ultrasonic sensor HC-SR04, a BNO055 orientation sensor and a TSC3200 color sensor were integrated to the robot. Since it is very important to accurately read the sensor signals, we decided to apply filtering techniques. When signal noise is very significant, but precision is not necessarily needed, a median filter is applied, pursuing the approach taken by DaPhoosa (2018). Besides, when signal noise is not highly noticeable but precision is essential, a Kalman filter is implemented, in accordance to the mathematical algorithm coded by Lauszus (2012). The result graphs are shown in Figures 8 and 9, respectively.

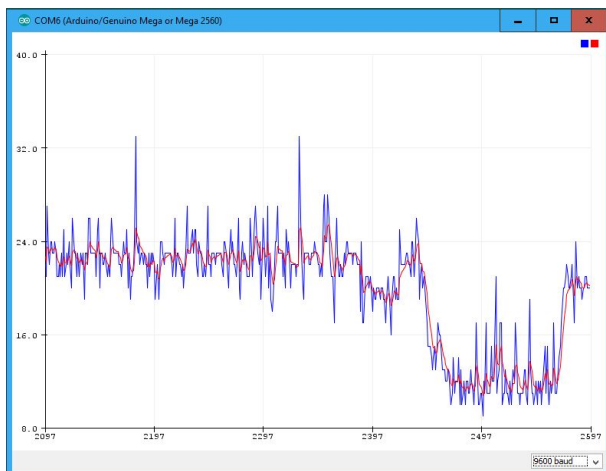


Figure 8. Medial Filter

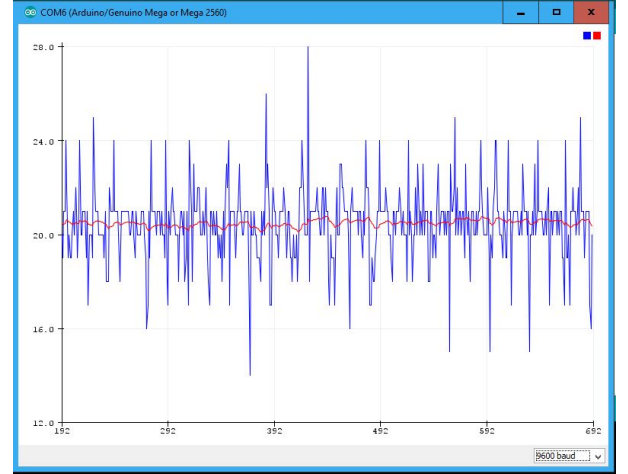


Figure 9. Kalman Filter

D. Navigation System

One of the major challenges of this competition is time, therefore, movements should be fast but still guarantee preciseness. In this regard, we applied Proportional Integral Derivative (PID) to improve movements. Following the Arduino PID library guide (Beauregard, 2011) and based on particular constants and the difference between the desired angle and the output of the BNO055 orientation sensor, an output is calculated to ensure exactness.

On the other hand, some movement routines demand to move specific distances. Hence we have settled to employ Pololu magnetic encoders. Encoders steps are captured through an Arduino interrupt function every time a wheel spins a complete revolution. Finally, using those steps, a distance traveled is estimated.

E. Computer Vision

This section describes the computer vision developed using OpenCV and Python-Numpy, inspired in the OpenCV-Python Tutorials (Mordvintsev & K., 2013). Since the environment is widely open to any conditions of light and background noise, the algorithm to detect containers should be robust enough to handle these uncertainty conditions. The computer vision analysis is fundamental for future decision making. The main goal is to detect the color of the containers at the top of each stack in order to calculate time costs.

The first step is to apply masks of three different ranges of color (red, green and blue) to the same image and transform it into three binarized images tagged with their own respective color. The input must be a hsv image, taken by the robot's lateral cameras. If a pixel's value is greater or less than the threshold, it is assigned either black or white respectively. For instance, the output is an image with only black or white pixels. It is thought that white is most present than black on a random image, so the threshold is iterated from 5% to 60% of the total resolution, this iteration is needed due to the unknown light conditions. The previous process is named adaptive thresholding.

The erosion of the image separates the edges of the detected rectangles so that they can be processed individually. According to Mordvintsev & K. (2013), Gaussian blurring is useful to eliminate black background noise. At this point the image is ready to obtain its contours using a routine provided by OpenCV that will return a numpy lists of all contours in the image.

Once the contours are found, the rectangles will be obtained from them. The criteria to obtain rectangles is the extent of the contour; this is, the number obtained by dividing the area of the object by the area of the minimum enclosing rectangle. If the quotient is close to 1, the contour must be a rectangle. In Figure 10, the green rectangle represents the minimum enclosing rectangle and the red perimeter encloses the contour area.



Figure 10. Extent of contours to obtain rectangles

$$\text{if} \left(\frac{\text{contour area}}{\text{min. enclosing rectangle area}} > \epsilon \right) \Rightarrow \text{contour is a rectangle}$$

where ϵ is around 0.8 (1)

After the execution of this step, the information obtained results in a list of all the rectangles detected knowing their respective color. However, since the competition is in an open zone, rectangles not belonging to a single *stack* but to the environment can be easily detected as well. This is known as image *noise*.

In order to depreciate the *noise* detected, the implementation of another algorithm is needed. The algorithm iterates through every detected rectangle regarding their contour area and try to match the rectangles with the most similar horizontal axis range and group them. After that it searches for contour intersections to know if they are touching each other somehow. This help us to identify easily if they are part of the same stack. Consequently, just keeps track of the two stacks with the biggest amount of *containers*, if there are more than one stack with the same amount of *containers* is detected, the one with the biggest area is maintained. At last, the designated color of each top container is returned regarding its vertical center position.

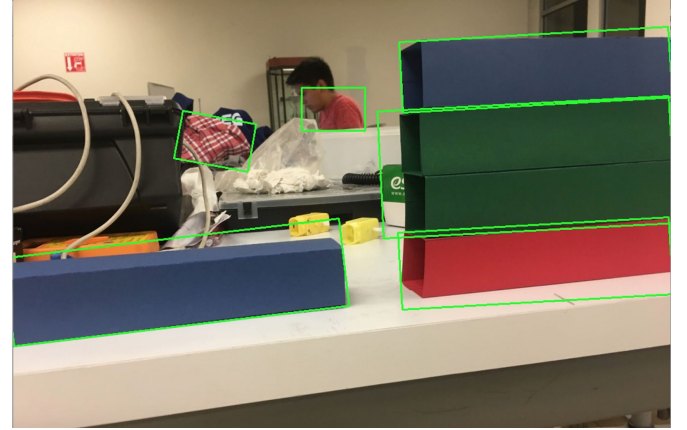


Figure 11. Potential containers found

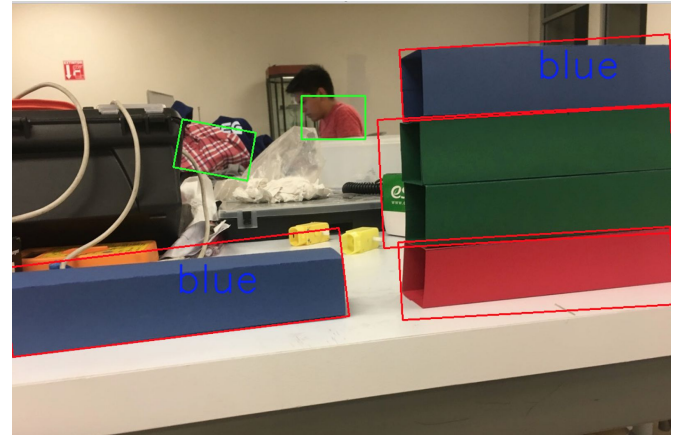


Figure 12. Containers surrounded on red color after filtering

V. CONCLUSION

Cargo and transportation services require autonomous processes to facilitate the management of containers with different scheduled destinations. Therefore, to accomplish the desired goals within a time limit it's necessary to have a well established strategy in order to develop an algorithm that can easily adapt to external conditions. Our algorithm decides which path and actions to take depending on several variables such as container color and the robot's position, with the main intention of minimizing time spent. The robot counts with computer vision and sensors (e.g. BNO055) to orientate itself through the environment, in addition to filter algorithms that facilitate the interpretation of the data received.

The elaboration of this text has helped us evaluate our progress so far, as well as address important details in the main strategy (e.g., orientation of the mechanism, priority for containers). It is important to emphasize that the robot that will be used for this year's LARC IEEE Open competition is being developed with the best technology within reach. Most, if not all of the ideas and designs present in this text are going to be incorporated into the final version.

VI. ACKNOWLEDGMENT

All the team members are grateful with the Instituto Tecnológico y de Estudios Superiores de Monterrey for giving all their support to RoBorregos and giving us the opportunity to participate on this competition. Also the team express their sincere gratitude to our advisors Sergio Camacho, Juan M. Hinojosa and Vianney Lara for all their support on the team initiative and their reviews on this document. Ultimately, to team sponsors (Cerrey, John Deere, Yazaki, Dextra Technologies, Ventus Technology, Tech Make, Oneila, Akky) that believe in the development of technology and project as this kind.

VII. REFERENCES

- [1] Beauregard, B. (2011). *Arduino PID - Guía de uso de la librería*. Retrieved from: <http://brettbeauregard.com/blog/wp-content/uploads/2012/07/Gu%C3%A1-da-de-uso-PID-para-Arduino.pdf>
- [2] DaPhoosa (2018). *Median Filter*. Retrieved from: <https://github.com/daPhoosa/MedianFilter>
- [3] Dassault Systèmes. (n.d.) *Guía del instructor para la enseñanza del software SolidWorks*. Retrieved from: https://www.solidworks.com/sw/docs/instructor_WB_2011_ESP.pdf
- [4] Lauszus, K. (2012). *Kalman Filter*. Retrieved from: <https://github.com/TKIElectronics/KalmanFilter>
- [5] Makrolon. (n.d.). *Línea de láminas de policarbonato*. Retrieved from: http://www.stabilit.com/images/Folleto%20Makrolon%20AGO16.compressed_3a7c31
- [6] Mihai, A. (2014.) *The place and the importance of power electronics*. Retrieved from: http://www.euedia.tuiasi.ro/lab_ep/ep_files/Lab_no_1_c1.pdf
- [7] Mordvintsev, A., & K., A. (2013). *OpenCV-Python Tutorials*. Retrieved from: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_contours/py_contour_features/py_contour_features.html?highlight=contours
- [8] Naylamp (2016). *Tutorial Sensor de Distancia SHARP*. Retrieved from: https://naylampmechatronics.com/blog/55_tutorial-sensor-de-distancia-sharp.html
- [9] NTN Global. (n.d.). *LINEAR BEARINGS*. Retrieved from: https://www.ntnglobal.com/en/products/catalog/pdf/2300E_16.pdf
- [10] Pololu. (2016). *25D mm Metal Gearmotors*. Retrieved from: <https://www.pololu.com/file/011042/25d-metal-gearmotor-dimension-diagram.pdf>