Autonomous 4DOF Robotic Manipulator Prototype for Industrial Environment and Human Cooperation

Inês Garcia
Industrial Electronics Dept
University of Minho
Guimarães, Portugal
A70557@alunos.uminho.pt

Fernando Gonçalves
Mechanical Engineering Dept.
University of Minho
Guimarães, Portugal
A70569@alunos.uminho.pt

Tiago Ribeiro
Industrial Electronics Dept.
University of Minho
Guimarães, Portugal
A71157@alunos.uminho.pt

Pedro Fernandes
Industrial Electronics Dept
University of Minho
Guimarães, Portugal
A68385@alunos.uminho.pt

César Rocha
Mechanical Engineering Dept.
University of Minho
Guimarães, Portugal
A72264@alunos.uminho.pt

Ricardo Boucinha
Mechanical Engineering Dept.
University of Minho
Guimarães, Portugal
A71482@alunos.uminho.pt

Gil Lopes
Industrial Electronics Dept
ALGORITMI Center
University of Minho
Guimarães, Portugal
gil@dei.uminho.pt

A. Fernando Ribeiro
Industrial Electronics Dept
ALGORITMI Center
University of Minho
Guimarães, Portugal
fernando@dei.uminho.pt

***Abstract*— This paper describes the design and development of an autonomous robotic manipulator with four degrees of freedom. The manipulator is named RACHIE - "Robotic Arm for Collaboration with Humans in Industrial Environment". The idea was to create a smaller version of the industrial manipulators available on the market. The mechanical and electronic components are presented as well as the software algorithms implemented on the robot. The manipulator has as its primary goal the detection and organization of cans by color and defects. The robot can detect a human operator so it can deliver defective cans by collaborating with him/her on an industrial environment. To be able to perform such task, the robot has implemented a machine learning algorithm, a Haar feature-based cascade classifier, on its vision system to detect cans and humans. On the handler motion, direct and inverse kinematics were calculated and implemented, and its equations are described in this paper. This robot presents high reliability and robustness in the task assigned. It is low-cost as it is a small version of commercial ones, making it optimized for smaller tasks.**

Keywords—Handler, Robotic Manipulator, 4 DOF, Machine Learning, can detection, human cooperation, human detection, image processing, kinematics.

I. Introduction

Nowadays, the world of industrial and service robots is in continuous expansion. Increasingly, the implementation of robotic manipulators in an industrial environment is essential, both for the sake of reducing hazards of heavy jobs and increasing the manufacturing efficiency. Furthermore, the worldwide competition "RoboCup" has boosted the deployment of service robots in other environments, such as rescuing people from disasters to home chores in house environments, with the "RoboCup @Home" competition. However, one of the significant challenges of industrial robots and services is object manipulation.

This paper describes an articulated autonomous robotic manipulator with four rotational degrees of freedom (DOF), developed by a group of students from Industrial Electronics and Computer Engineering as well as Mechanical Engineering. This robot is intended to be a small and low-cost manipulator, compared to the ones available on the market and it was designed for an industrial environment. The developed handler is called "RACHIE", which stands for "Robotic Arm for Collaboration with Humans in Industrial Environment", presented in Fig. 1. This paper focuses on the development of all the robot's different systems. Thus, it is divided into five main sections, namely introduction, methodology, results, discussion, and conclusions.

A. Goal Task

In a preliminary phase, the following purpose was established for the robot: detection and transportation of cans of different colors, as well as their organization in their respective places around them. Another goal is the recognition of defective cans based on shape and color. The defective cans will be stored for later human intervention or handed over to the operator to analyze the problem and put them back on the shelf (Fig. 2). For the objects and human detection, the robot has a vision and image processing system as the external sensor of the manipulator, which uses machine learning. Therefore, this paper presents the mechanical design and structure; the implementation of actuators and sensors inside the various joints of the robot; the sizing and implementation of all the electronic

Fig. 1. Picture of RACHIE, holding a can.

978-1-7281-3558-8/19/$31.00 ©2019 IEEE

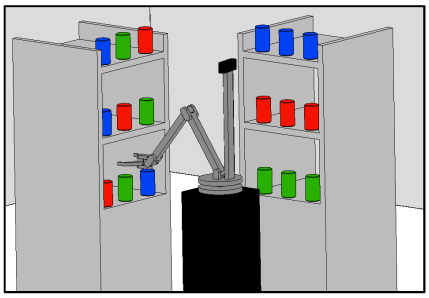

Fig. 2. Example of the robot in a work environment sorting cans.

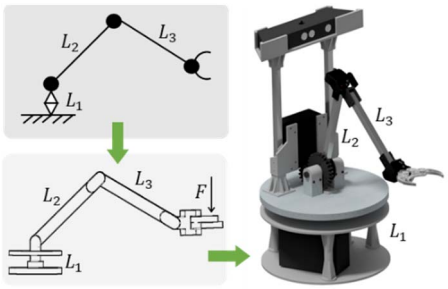

Fig. 3. Manipulator's design evolution.

parts involved; the manipulating movement and its kinematic analysis; and the development of object recognition and manipulation algorithms.

B. State of the art

Most of the robotic manipulators can be classified into four main categories [1]:

- Redundant Manipulators: serial manipulators which have more degrees of freedom than those required for the task. This allows their movement to be optimized and more adaptable, allowing different approaches and obstacle avoidance. The manipulator RACHIE belongs to this category;
- Parallel Manipulators: manipulators which have one end-effector connected to the base through several actuators. Mostly used in situations where high accuracy and stiffness of the manipulator is required;
- Cable-driven Manipulators: manipulators where the position of the end-effector or a payload is controlled by the movement of a series of cables attached to a fixed base;
- Mobile Manipulators: the result of mounting a redundant manipulator in a mobile base, resulting in a much more expanded workspace and better positioning.

II. Methodology

A. Hardware development

In this section, the hardware developed will be presented, both the mechanical components as well as the electronics integration.

1) Mechanics

The mechanical design is a project essential feature since it is fundamental to guarantee the structure robustness, without compromising the robot's accuracy. Fig. 3 presents the evolution of the manipulator's design (from a 2D model to a 3D model developed on the Autodesk Inventor tool). As illustrated, the manipulator is composed of four degrees of freedom. One of them has a full 360º of freedom (Torso: θ_1), while the others are limited, one up to 90º (Shoulder: θ_2) and two up to 180º (Elbow: θ_3 and Wrist: θ_4). The servo motors were chosen considering the bars length (0) and weight, as well as the servo's own weight. The equations (1), (2) and (3) present the torque evaluation for each servo, respectively [2].

Where P_a is the weight of the arm, P_b the weight of the forearm, P_c the weight of the grabber, P_L of the payload, P_{s_3} of the wrist servo and P_{s_2} of the elbow servo. It should be noted that most of the robot structure was designed and printed with a 3D printer in PLA, with exceptions of the beams that are made of aluminum to provide higher robustness. The resistance of the beams was assessed evaluating the critical solicitation regarding static analysis. In this critical situation, the maximum load (a radial component of gravitational forces is maximized) is applied to the bars, therefore the servos would be reaching their maximum torque. After verifying if the bars could support this maximum load, and the bending stress caused by it, the structural integrity of the manipulator is guaranteed. In order to increase the torque output of the servos, spur gears were used in the torso and shoulder, allowing the use of less powerful servos.

TABLE I. Bars lengths

	Value	Unity
L1	170	mm
L2	250	mm
L3	250	mm
L4	100	mm
L5	48.5	mm
L6	130	mm

$$M_2 = \left(\frac{L_2}{2}\right)P_a + L_2P_{s_2} + \left(L_2 + \left(\frac{L_3}{2}\right)\right)P_b + (L_2 + L_3)P_{s_3} + \left(L_2 + L_3 + \left(\frac{L_4}{2}\right)\right)P_c + (L_2 + L_3 + L_4)P_L \quad (1)$$

$$M_3 = \left(\frac{L_3}{2}\right)P_b + L_3P_{s_3} + \left(L_3 + \left(\frac{L_4}{2}\right)\right)P_c + (L_3 + L_4)P_L \quad (2)$$

$$M_4 = \left(\frac{L_4}{2}\right)P_c + L_4P_L \quad (3)$$

2) Electronics

At the electronics level, the robot is divided into two parts, the high-level software, running on MSI Cubi 2 Board PC and the low-level software, running on a ChipKit Uno32 embedded microcontroller system. The PCB board was designed and built (Fig. 4), with the software CadSoft Eagle, which contains all the electronic circuits necessary for the manipulator. This board contains a Chipkit Uno32 processor as well as the sensory and

actuator system. The robot has four Inertial Measurement Units or IMUs (CMPS 11), one at each joint, one Xbee network device used for debugging, five servo motors (which correspond to the manipulator joints, Torso – 360º, 20kg/cm, 5-7.4V, 3A; Shoulder – 180º, 15kg/cm, 6-7.2V, 100mA nominal; Elbow – 180º, 15kg/cm, 6-7.2V, 100mA nominal; Wrist – 180º, 15kg/cm, 6-7.2V, 100mA nominal; End-Effector – 180º, 3kg/cm, 4.8-6V), two safety limit switches at the z-axis, which allow an emergency stop to prevent an incorrect movement which may damage the structure. The robot has one buzzer, and an RGB LED for debugging.

Due to some sensor connection restrictions, different methodologies for board communication and sensors/actuators were designed. The Kinect 360 camera is connected via USB to the MSI Cubi 2 board. Fig. 5 presents the architecture of the electronic system as well as the communication protocols implemented between the various devices. Since the power supply and the Torso servo motor are fixed to the robot base, to lower the manipulator's center of mass and, consequently, increase its stability, a continuous 360º slip ring connector was implemented between the fixed and movable base parts. This connector provides a fully continuous rotation without curling up the cables.

B. *Kinematic Analysis*

After defining the base geometry and the manipulator degrees of freedom, the robot kinematics was handle [3].

1) Direct kinematics:

This kinematics allows the end-effector coordinates determination as a function of the angles from the various joints. This process starts with the attribution of the coordinate axes being Z0 to Z4 corresponding to the joint rotation axis. Axis X0 to X6 work in the direction of the bar while the Y0 to Y6 axes are at 90 ° from X and Z (Fig. 6).

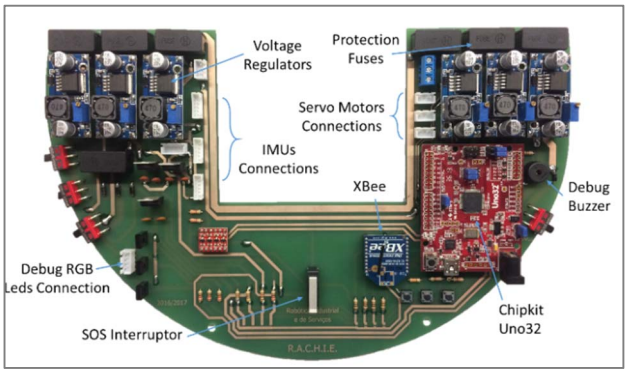

Fig. 4. RACHIE main PCB developed.

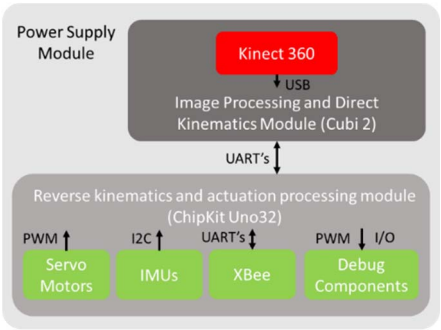

Fig. 5. Electronic system architecture.

The Denavit-Hartenberg parameters were determined from the base, reference {0}, to the end-effector, reference {5}, i.e., the RACHIE claw and from the base to the camera, reference {6} (0and TABLE III.). The coordinates from the base to the end-effector ({0} → {5}) allow a grip position control for the task to be performed by the robot. The relative coordinates of the camera ({0} → {6}) serve the purpose of knowing the relative position of the camera in relative to the base. Thus, it determines precisely the difference between the location that the camera detects to the place that the end-effector moves. A precise end-effector motion is achieved depending on the position required by the camera. The T_5^0transformation matrix can be calculated by (4).

$$\begin{bmatrix} c_1C_3 & -c_1S_3 & s_1 & c_1C_3L_4+c_1C_2L_2+c_1c_2L_2 \\ s_1C_3 & -s_1S_3 & -c_1 & s_1C_3L_4+s_1C_2L_3+s_1c_2L_2 \\ S_3 & C_3 & 0 & S_3L_4+S_2L_3+s_2L_2+L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

Where s_1, s_2, c_1, c_2, C_2, S_2, C_3, and S_3 are calculated by the equations (5), (6), (7), (8), (9), and (10).

$$C_3=\cos(\theta_2+\theta_3+\theta_4) \quad (5)$$

$$S_3=\text{sen}(\theta_2+\theta_3+\theta_4) \quad (6)$$

$$C_2=\cos(\theta_2+\theta_3) \quad (7)$$

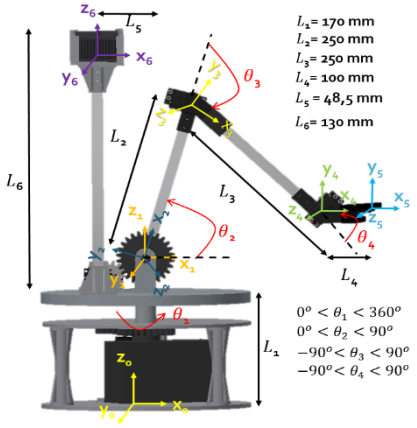

Fig. 6. Assignment of the coordinate axes to the manipulator's joints.

TABLE II. DENAVIT-HARTENBERG PARAMETERS, FROM BASE TO END-EFFECTOR.

	a_{i-1}	α_{i-1}	d_i	θ_i
{0} → {1}	0	0	L1	(θ1)
{1} → {2}	0	90	0	(θ2)
{2} → {3}	L2	0	0	(θ3)
{3} → {4}	L3	0	0	(θ4)
{4} → {5}	L4	0	0	0

TABLE III. DENAVIT-HARTENBERG PARAMETERS, FROM BASE TO THE CAMERA.

	a_{i-1}	α_{i-1}	d_i	θ_i
{0} → {1}	0	0	L1	(θ1)
{1} → {6}	-L5	0	0	(θ2)

$$S_2=\text{sen}(\theta_2+\theta_3) \quad (8)$$

$$s_1=\text{sen}(\theta_1), s_2=\text{sen}(\theta_2) \quad (9)$$

$$c_1=\cos(\theta_1), c_2=\cos(\theta_2) \quad (10)$$

Simplifying the end-effector coordinates, reference {5} (x_e, y_e and z_e) results in the system of equations (11). Through these three equations, it is possible to define the position of the end-effector from RACHIE's base to its claw, resulting in its three axes displacement in relation to the mechanical assembly's four possible rotations.

$$\begin{cases} x_e=c_1C_3L_4+c_1C_2L_3+c_1c_2L_2 \\ y_e=s_1C_3L_4+s_1C_2L_3+s_1c_2L_2 \\ z_e=S_3L_4+S_2L_3+s_2L_2+L_1 \end{cases} \quad (11)$$

1) Inverse Kinematics

This kinematic allows determination of the angles required for the servos to reach the desired position. The calculation of the inverse kinematics was carried out starting from a geometric analysis of Fig. 7. The dimensions of the bars and limits of the angles were the same as those expressed in Fig. 6. The initial expressions were written as a wrist position function, coordinates (X_p, Y_p and Z_p), through equations (12), (13) and (14).

$$X_p= X_e-L_4c_1 \quad (12)$$

$$Y_p= Y_e-L_4s_1 \quad (13)$$

$$Z_p= Z_e \quad (14)$$

So, it is possible to work the equations to obtain the values of θ_1, θ_2, θ_3 and θ_4 as a function of X_e, Y_e and Z_e, as represented by the system of equations (15). To validate the equations defined in the direct and the inverse kinematics, simulations were performed on the Solidworks software, using a simplified RACHIE model. The trajectory calculation can be obtained by establishing a starting and ending point in the software. The angles needed to reach these points are obtained by inverse kinematics, and the software generates the ideal path between them, simulating the behavior of the robot.

C. Software

1) Software architecture

The central processor used is ChipKit Uno32, a 32-bit microprocessor. Its purpose is divided into four modules: actuators, sensors, communication with other processors and debug. The five servo motors controlling the joints belong to the actuator's module, while the four IMUs belong to the sensor's one. The IMU sensors allow a loop closer from the respective degrees of freedom, improving the manipulator's movements precision. In the sensor module, there is also the emergency button and the man-machine interaction buttons (worked by an interrupt for a better system performance). The communication module has a UART connection between the MSI Cubi 2 board and the XBee, which provides a wireless UART communication. In the Debug and information module, there is an LCD, a buzzer, an RGB LEDs ribbon, and status LEDs. The software developed is a state-machine system. There are three different algorithms that the CPU processes:

a) Human-machine interaction Interruption

This algorithm raises the man-machine interaction flag, so when the robot finishes the current process it has to change its status from "Processing a can to a shelf" to "Processing a defective can to an operator".

a) Processing a can to a shelf

This algorithm starts by moving to the initial position where it checks if there is any can to pick up by Kinect image processing. If there is no can, the robot performs periodic verifications every second until there is one. When a can is ready to be picked up, the MSI Cubi 2 board receives the can position regarding the Kinect reference {6} and sends the coordinates of the can regarding reference {1} as well as the color of the can. The ChipKit microcontroller calculates the inverse kinematics and the local to place the can. Then it takes

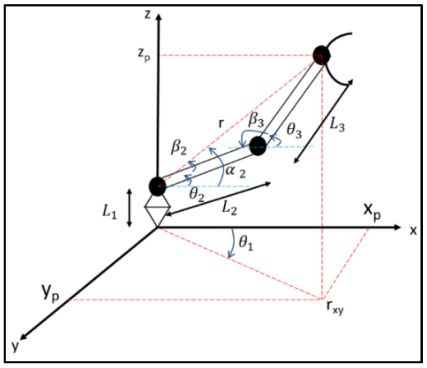

Fig. 7. Schematic representation of the angles and positions to obtain the inverse kinematics.

$$\theta_1= \tan^{-1}\left(\frac{Y_e}{X_e}\right)$$

$$\theta_2 = \tan^{-1}\left(\frac{\sqrt{(X_e-L_4c_1)^2+(Y_e-L_4s_1)^2}}{Z_p-L_1}\right)-\cos^{-1}\left(\frac{{L_3}^2-{L_2}^2-{Z_e}^2-(Y_e-L_4s_1)^2-(X_e-L_4c_1)^2}{-2L_2\sqrt{{Z_e}^2+(Y_e-L_4s_1)^2+(X_e-L_4c_1)^2}}\right)$$

$$\theta_3 = \pi-\cos^{-1}\left(\frac{{Z_e}^2+(Y_e-L_4s_1)^2+(X_e-L_4c_1)^2-{L_3}^2-{L_2}^2}{-2L_2L_3}\right) \quad (15)$$

$$\theta_4 = \cos^{-1}\left(\frac{{Z_e}^2+(Y_e-L_4s_1)^2+(X_e-L_4c_1)^2-{L_3}^2-{L_2}^2}{-2L_2L_3}\right)-\tan^{-1}\left(\frac{\sqrt{(X_e-L_4c_1)^2+(Y_e-L_4s_1)^2}}{Z_p-L_1}\right)$$
$$+\cos^{-1}\left(\frac{{L_3}^2-{L_2}^2-{Z_e}^2-(Y_e-L_4s_1)^2-(X_e-L_4c_1)^2}{-2L_2\sqrt{{Z_e}^2+(Y_e-L_4s_1)^2+(X_e-L_4c_1)^2}}\right)-\pi$$

the can and leaves it in the place it was calculated. Finally, to decide whether to move to the algorithm of "Processing a defective can to an operator", it checks if there was an interruption of the human-machine interaction; otherwise, it will redo this algorithm.

b) Processing a defective can to an operator

This algorithm starts by checking if there is a defective can to be delivered to an operator, using color and shape detection of the image processing algorithm. If not, it will print an error message and leaves this state. If it does leave, it goes to the position where an operator should be and checks the presence of a human, by face detection (Fig. 12). If not, an error image is printed out of this algorithm. If so, the robot will get the defective can and return to the position where the human is to confirm that he is still there. If so, the central processor receives the operator's hand coordinates, calculates the inverse kinematics and delivery, terminating the algorithm and returning to the algorithm of "Processing a can to a shelf". If the operator is not there to receive the can, an error message is printed, the can is put back where it was, and the state is finished.

2) Image processing

a) Machine learning implementation and training

In RACHIE's vision system, a Microsoft Kinect is used to obtain color images, through an RGB camera, as well as depth images collected with an arrangement between infrared (IR) emitter and IR sensor. This arrangement enables good quality color images and the respective depth of the analyzed environment. With this information, it was possible to implement algorithms that allow recognizing people and objects. Therefore, RACHIE can identify human faces and cans, due to the implementation of Haar feature-based cascade classifier, a machine learning object detection algorithm used to identify objects in an image [4]. The cascade classifier training required images that contained faces or cans (positive images) and another set with images that had neither of the target features (negative images). The classifier extracts interest Haar features from each image and tries to detect it over the input image. The key advantage of a Haar-like feature over most other features is its fast calculation speed.

b) Cans and operator detection algorithms

The vision system is essential in inverse kinematics computation process, which discovers the object coordinates to be manipulated, or for the operator to interact. The algorithm developed for the can detection process has the following sequence: a new image acquisition is requested from the Kinect camera, an RGB and a depth image are supplied to the program. The algorithm processes the RGB image regarding the colors (red, green and blue) and discovers the center of the cans (x_c and z_c); the depth image is analyzed to find out the value of y_c required. Finally, through its inverse kinematics, coordinates are calculated relative to the end-effector and are supplied to the ChipKit microcontroller.

The human/operator detection algorithm follows the following logic: it is required that a human is found for interaction, whereby it is performed a human recognition. If an operator is detected and is ready to interact, the algorithm responds affirmatively and sends to the microcontroller the coordinates of the can from reference {1}. If not, it sends a negative message.

3) Manipulator control

As previously stated, the IMUs sensors, implemented on each joint of the manipulator, present an essential feature in the manipulator's performance. Since the servo motors are not able to accurately know the angle/state that they are, the IMUs accelerometers and gyroscopes can provide that information through Kalman filter, in order to calculate the kinematics. Thus, the manipulator control is designed as a closed loop and has a PID control to compensate and establish the right position of each joint for different object weights, as well as its desired trajectory.

III. RESULTS

A. RACHIE workspace

RACHIE's workspace was calculated, using the Autodesk Inventor tool. The workspace corresponds to the coordinates set that the end-effector can reaches. Fig. 8 presents the 2D and 3D simulated workspace. Considering the shoulder as the reference point and its angle to the end-effector α: the manipulator presents the following limits: for cases where α is between -30º and 0º, the minimum position is limited by the abscissa being bigger than 176 mm. The limitation for the maximum is the radius, and the limit varies from 420 mm to 595 mm. For cases where α is between 0º and 45º, the minimum position is limited by a radius that varies from 176 mm to 324 mm and maximum position is limited by a radius of 595 mm. For cases where α is between 45º and 70º, the minimum position is limited by a radius of 324 mm and the maximum position a radius of 595 mm. For α between 70º and 90º, the minimum position is limited by a radius of 324 mm and the maximum varies between a radius of 468 and 595 mm. In total, RACHIE has a work envelope of 592,1 dm3.

B. Object weight analysis

Fig. 9 and Fig. 10 present the relation between the load mass manipulated and the end-effector height, regarding the initial position. The manipulator behavior was analyzed for three different positions: rest, standard and extreme. When comparing closed and open loop behavior in a rest position, for smaller weights the closed loop allows the servos to compensate the consequences of the weight, guaranteeing a fixed position for the end effector. However, as the weight increases, the servos stop being able to make this adjustment, the results for both open and closed loop become almost parallel lines, the weight causing the same variation in deviation. In the standard and extreme positions, the results are similar, having the closed loop a more accurate end-effector position. However, the maximum open loop load that can be supported is superior. That is because in closed loop the servos apply greater loads due to the effort of returning to the desired position.

Fig. 8. RACHIE 2D and 3D workspace.

Fig. 9. End-effector positions, in function of the initial one, versus the load manipulated at the manipulator rest position.

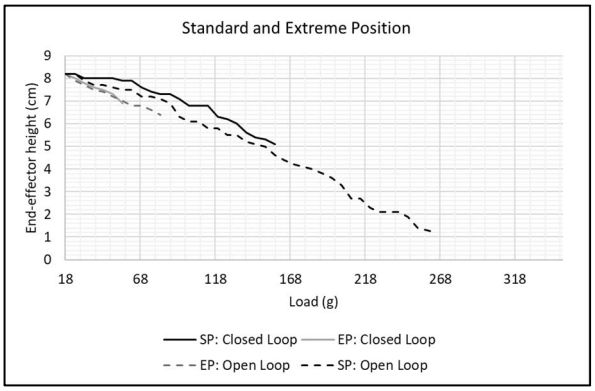

Fig. 10. End-effector positions, in function of the initial one, versus the load manipulated, at the manipulator standard and extreme position.

C. *Image Processing*

Fig. 11 and Fig. 12 present images obtained with the Kinect camera. In Fig. 11 it is possible to see the RGB and depth image, necessaries for the can detection. The RGB image retrieves the can's x and z coordinates and the depth image, the y coordinate. On the operator detection, the software analyses the image in order to find a human (operator) face, which is within the red circle in the image.

IV. Discussion

Through the verification of several manipulator positions in the physical model, as well as the virtual model, it was possible to determine that the kinematic analyses corresponded to the results obtained in reality. Several tests were performed considering the robot mechanics, for the most extreme cases of operation, such as high loads, high speeds, among others. The design validation was confirmed, the robot can handle loads up to 150 g, being the torque of the motors enough to move the system to the standard position. The implementation of a collapsible structure in aluminum and PLA proved to be a reliable and robust low-cost approach. Regarding the results obtained from the vision system implementation, the combination of images obtained with the Kinect camera (RGB and depth) allowed a reliable can coordinate acquisition concerning the camera, with high accuracy. Furthermore, the Haar feature-based cascade classifier implemented on the image analysis proved to be an effective solution since the manipulator learned the features to detect.

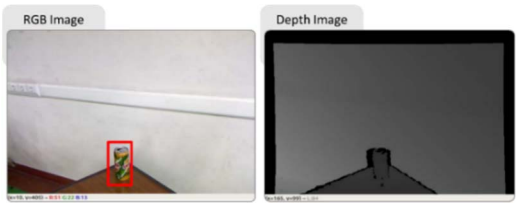

Fig. 11. Images from can detection algorithm

Fig. 12. Image regarding the operator detection algorithm.

Through the implemented kinematics, it was possible to compute the same coordinates, now as a function of reference {1} of the manipulator, with high accuracy and precision. The human face recognition system implemented also showed good performance during the machine-man interaction process. Due to the high number of mechanical and electronic components interconnected to each other, the failure of one of them could compromise the entire system. For this, it was fundamental to guarantee the operation of all the systems before starting the construction of the robot. This was obtained employing theoretical analyses and initial simulations that are fundamental to the well-being of this work. All the actual components behaved as designed, allowing to avoid the emergence of unforeseen events that could compromise the entire project.

V. Conclusions

A 4 DOF robotic manipulator is presented. The development of this manipulator involved multidisciplinary work. Efficiency could only be achieved by the continuous communication between the electronic and mechanical elements, in order to understand the possibilities and limitations of the two areas. This enabled a rapid definition of requirements, which in turn led to an optimization that made it possible to create a better result with the available components. Given the investment of the team on the RoboCup@Home competition, this manipulator is a great addition to the group. In the future, the implementation of an omnidirectional base and a localization algorithm (with a LIDAR sensor) [5] on the RACHIE manipulator, can be an essential improvement, being this robot capable of supporting CHARMIE robot developed by MinhoTeam,

VI. References

[1] L. Jin, S. Li, J. Yu, and J. He, "Robot manipulator control using neural networks: A survey," Neurocomputing, 2018.

[2] P. R. N. Childs, Mechanical Design Engineering Handbook. 2013.

[3] John J. Craig, Introduction to robotics Mechanics and Control. 2005.

[4] P. Viola and M. Jones, "People detection in rgb-d data," Proc. 2001 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 511–518, 2001.

[5] T. Ribeiro, I. Garcia, D. Pereira, J. Ribeiro, G. Lopes, and F. Ribeiro, "Development of a prototype robot for transportation within industrial environments," 2017 IEEE Int. Conf. Auton. Robot Syst. Compet., pp. 192–197, 2017.