

---

# SorterBot

Hydraulic arm that classifies notes according to the subject by computer vision and allows them to be sorted by date.

---

PROJECT SPRINT #4.

DATE: 18<sup>th</sup> May 2021

Víctor Batista Medeiros-1533111

Cristian Gutiérrez Gómez-1532397

Carlos Peña de Pedro-1532529

# Table of Contents

---

Project description	1
Electronic components	2
Schemes	3
Amazing contributions , extra components and 3D pieces	5
Description and use of pieces	6
Simulation strategy	8
Foreseen risks and contingency plan	14

# SorterBot

Hydraulic arm that classifies notes according to the subject by computer vision and allows them to be sorted by date.

## Project description

SorterBot is a pick and place robot that through the use of a hydraulic pump and a suction cup is able to perform two functionalities which are to classify subject notes by their title and to sort them by the relevant date that appears in those notes.

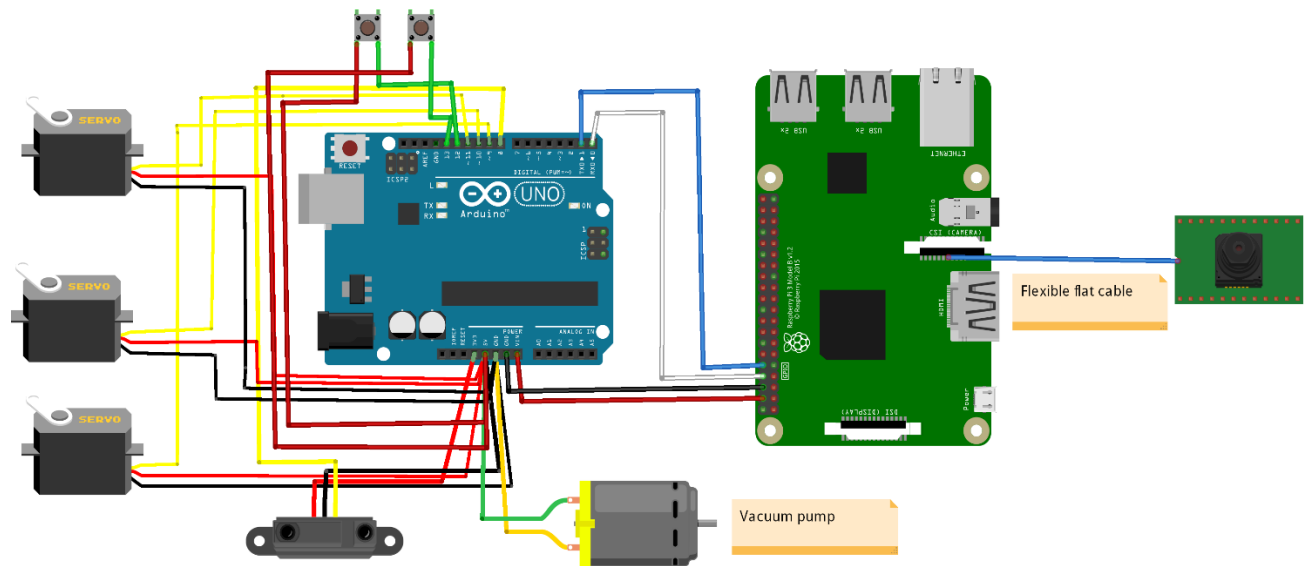
To perform the first functionality SorterBot implements computer vision by means of a camera that allows it to visualize the content of the notes. From the content it analyzes, it saves the information of the titles in order to classify them according to the subject to which they correspond. Also note that in case there are sheets that do not contain information on subjects or there are more subjects than available spaces, it classifies them in a separate set.

The second functionality, as we have already mentioned, allows us to organize the notes of a subject by dates. For this SorterBot uses computer vision that allows it to visualize the content of the notes from which it extracts the date that appears at the top of the sheet. Once the date is obtained, it uses the Hanoi algorithm with which it groups the sheets in different blocks depending on whether they correspond to an older or more current date in order to place them in an orderly fashion.

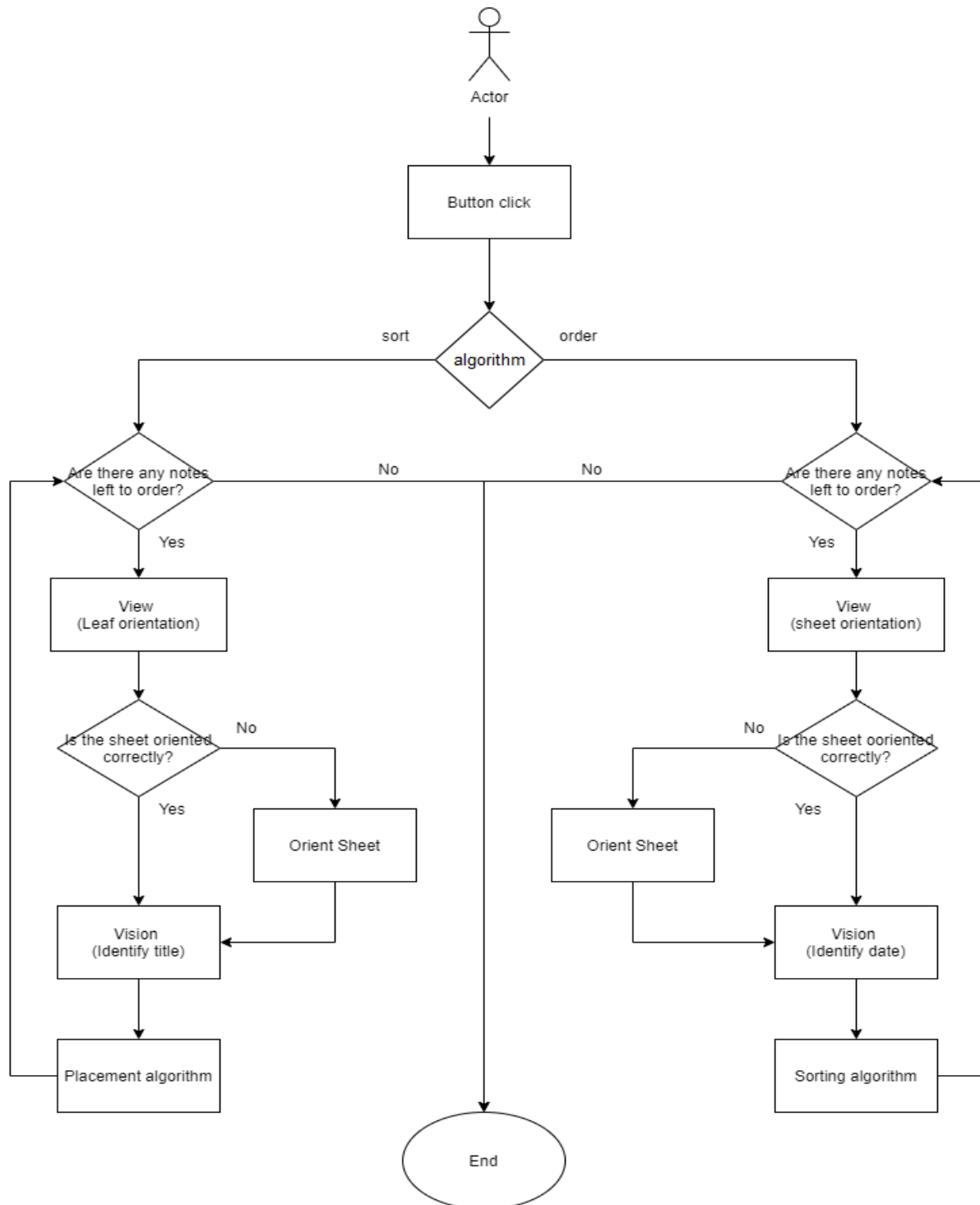
## Electronic components

- 1x Arduino
- 1x Vacuum pump
- 1x Suction cup
- 1x Raspberry pi
- 1x Raspberry pi camera
- 1x Power supply unit (8v)
- 3x Servomotor (1 x MG995 ->15kg, 2x MG90S -> 2,2kg) -> require de 4.8 a 7.3 v
- 1x Infrared proximity sensor
- 1x Robot Skeleton

## Hardware Scheme



## Software Architecture



## Amazing contributions

- It is a project that has a direct and wide application since it is automating a very frequent problem, the sorting of physical documents and their classification into different categories, thus allowing its use in any work or leisure environment.
- Our robot differs from other pick & place robots[1] because it can apply different strategies and algorithms to classify and group documents. It should be noted that these strategies are based on the treatment of images of documents and on obtaining their information, thus allowing us to use it in any case where there is text, since it does not focus on the form but on the content.

## Extra components and 3D pieces

- *2 Buttons*
- Body
- Rotational arm
- Proximity sensor holder
- Prismatic arm
- Flexible suction tube
- Camera support
- Suction tube storage and prismatic overlay (Vacuum zone)
- Base arm support
- 6 bars
- Nuts and bolts
- Servomotor Box

## Description and use of pieces

The main part of SorterBot is the body which apart from being an important part as far as the weight distribution of the structure is concerned, is the part where the vacuum pump and the robot's computers are kept, which in this case are the arduino and the raspberry pi.

In the body itself we find two buttons placed on its square base that allow us to select between the sorting algorithm (red button) and the organization (green button).

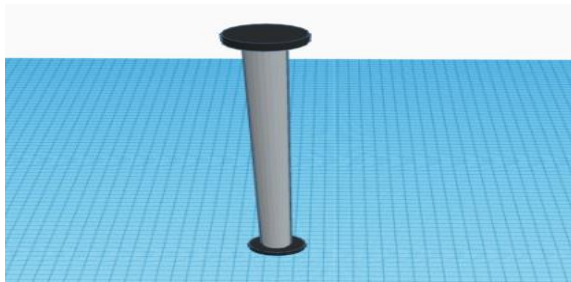
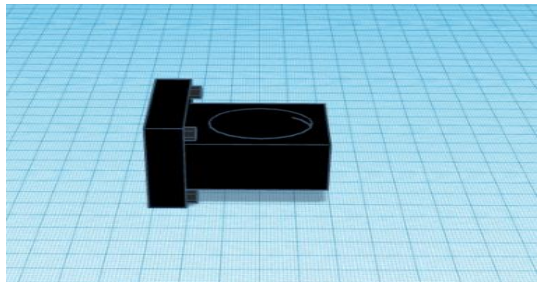
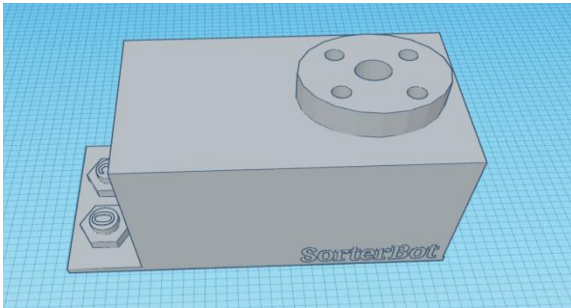
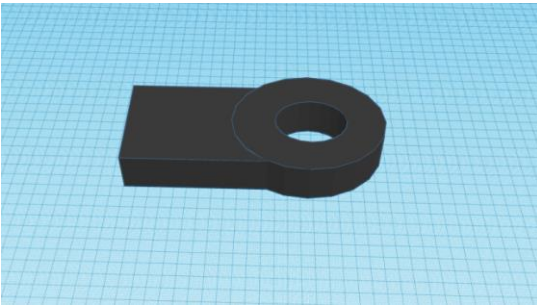
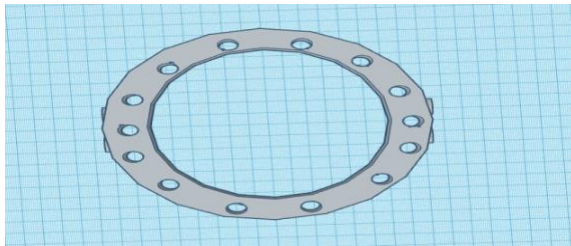
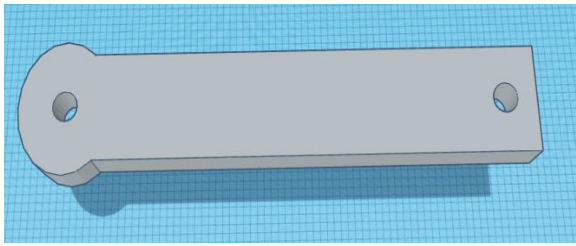
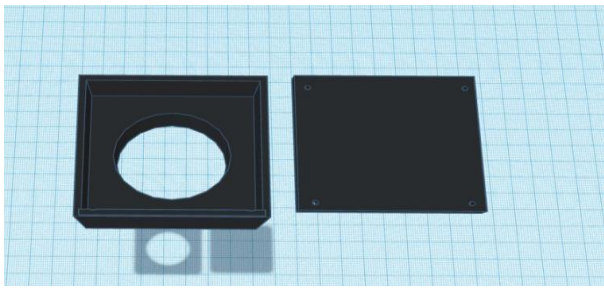
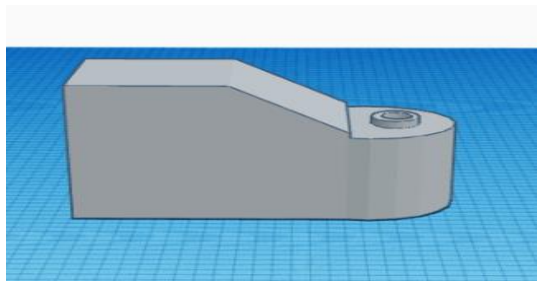
Next to this body we find a rotational arm connected by 4 bars that allows us to increase the mobilization of the body in the axis of the and, in addition to perform a 360 degree angle by means of the servomotor that is incorporated in the servomotor box located on top of the arm.

On top of this arm we find the piece that allows us to partially cover the prismatic arm as it moves along the z-axis, in addition to being the receiver of the flexible suction tube that is connected to the vacuum pump that goes from the base to the storage and generates the suction force of the suction cup [2]. It should be noted that this part allows us to create a vacuum zone.

Finally we find the camera support attached to the rotational arm and the proximity sensor support attached to the prismatic arm.

These two supports allow us in the first case to hold the camera of the raspberry pi perpendicular to the points, as well as to protect it and in the second case to place the sensor near the suction cup to calculate the distance to the object.



Prismatic Arm	Camera Support
	
Body	Proximity Sensor Bracket
	
Arm Support	Rotational Arm
	
Servomotor Box	Vacuum zone
	

## Simulation Strategy

To simulate the project the first thing we do is to define the 3D design of the robot using the TinkerCad application.

Once the design is done we export it in MTL format to be able to import it into the Coppelia Sim platform keeping its colors.

In this platform we add the pertinent links that correspond to the servomotors of our robot and we add a sensor to be able to visualize objects and another sensor to detect the proximity[3] to these.

Once the robot is assembled we have defined several scenarios both for the realization of the algorithms and for testing.

The first of them is a scenario in which we have defined a work table, a set of 6 file cabinets that are at the following angles 45,90,-90,120,-120,150,-150,180, and a series of sheets containing information of various subjects that are oriented at the zero position of the robot

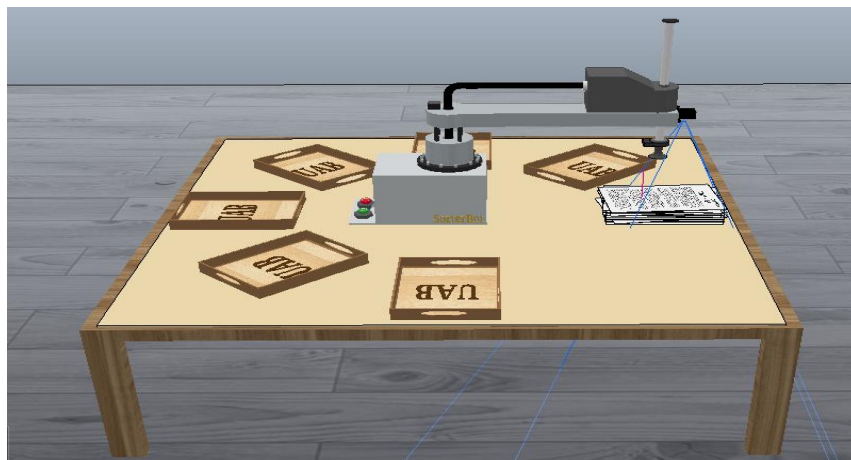


Image 1: Clasification scenario.

These angles have been chosen because of the size of the sheets, otherwise the binders would collide.

Also note that the angle -45 is reserved for the orientation of the leaves, otherwise it could be the case that there are two leaves in which the top one is well oriented and the bottom one is badly oriented and consequently an erroneous calculation would be made as shown in the following image:

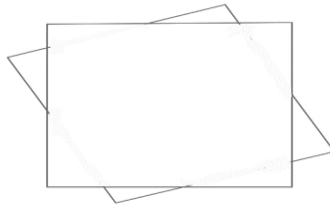


Image 2: Set of leaves with the lower leaf oriented wrongly.

In this scenario the robot is responsible for performing the classification algorithm for which it takes each sheet independently by using the proximity sensor that allows us to stop the arm before colliding with the sheets, and takes them to the orientation angle to see whether or not it is necessary to rotate them. Once the sheet is well oriented, the extraction of the text and identification of the title is carried out and then, by means of a strategy of percentage [4] of appearance and similarity of words, it is placed in a filing cabinet or other. If there is no subject associated with the sheet, an empty file cabinet will be found to place it, and if there are no file cabinets, it will be placed in the file cabinet located at 180 degrees.

As a second scenario we have used the same as the previous one but in this case the sheets are already classified in their respective filing cabinets as shown in the following image:

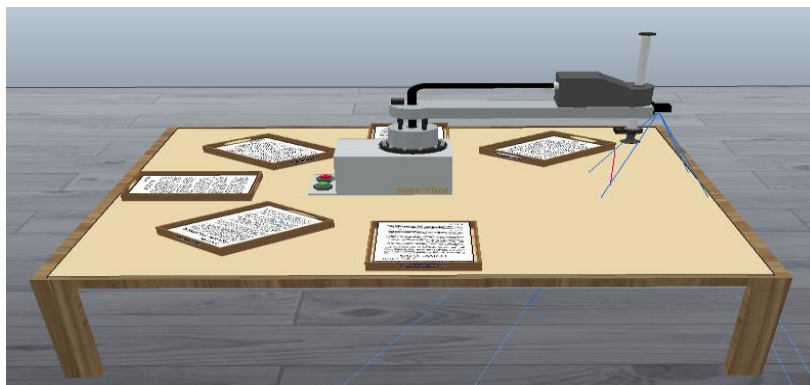


Image 3: Management scenario.

In this scenario we perform the sorting algorithm for which the robot goes to each file cabinet, in a set order, and sorts the sheets by helping the two nearby file cabinets to perform the same.

It is divided into two phases:

1. **Read dates:** Reads all dates from the stack, storing in an auxiliary angle.
2. **Stack Sort Algorithm:** Proceeds to sort the sheets, using a maximum of two binders and leaving them in a third one, which corresponds to the binder where the stack to be sorted was.

We make the abstraction of viewing the file cabinets with all their sheets as a stack with elements:

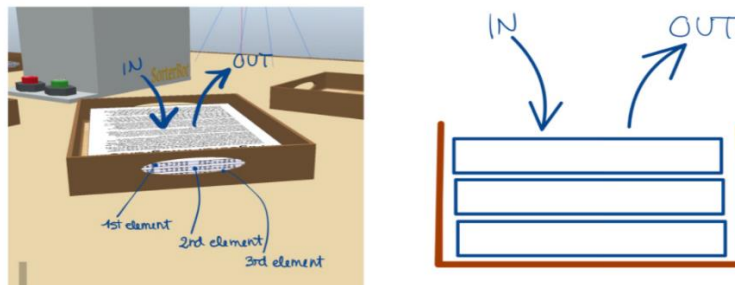


Image 4: The stack abstraction.

### Pseudocode Stack Sort Algorithm

*:param unsorted\_date\_list:*

*:param a1:* angle where the stack is located after reading the dates

*:param a2:* angle at which we will leave the stack sorted

*:param atemp:* an angle to be used to assist in sorting (will remain empty)

returns *:param sorted\_date\_list:*

WHILE unsorted\_date\_list HAS ELEMENTS:

    Grab the first paper sheet on the stack (FirstSheet)

    WHILE sorted\_date\_list HAS ELEMENTS:

        IF date(ordered\_date\_list[-1]) > date (FirstSheet):

            Leave the current sheet to the the axillary “atemp” stack

            Grab the first paper sheet on the “a2” stack

            Leave the current sheet to the the “a1” stack

```

ELSE:
    BREAK
IF sorted_date_list HAS NO ELEMENTS:
    Leave the current sheet at the original stack angle "a2"
ELSE:
    Grab the sheet for the axillary "atemp" stack

```

Also, in this scenario we perform two tests:

- **test\_resultat\_correcte**  
It consists of storing the result of the sorting in a list along the total sorting of all file cabinets.  
Finally it is compared with the expected result to see if the sorting has been done correctly.  
This test is useful to detect errors in the camera, the sorting algorithm and the general and functional operation of the MVP.
- **test\_arxivador\_buit**  
The purpose of this test is to check that when there is an empty tray, the robot ignores this tray and continues with the sorting.

The fourth scenario consist in only one filing cabinet in position -90 and a pile of misguided sheets as shown in the following image:

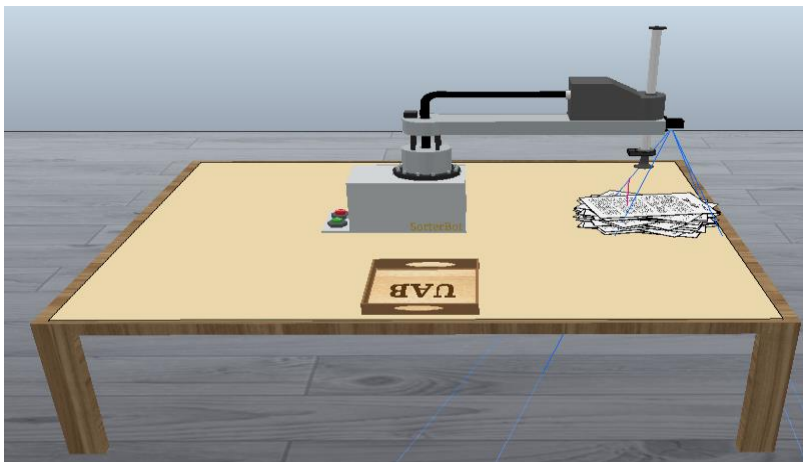


Image 5: Management scenario.

In this scenario we test the orientation algorithm for which the robot picks a sheet from the origin pile, then it moves it to orientation -45 where aren't any noises and the camera takes a photo of the sheet. Then we use computer vision techniques to take the longest line in the image corresponding to the sheet's height, we detect angle of that line and then we proceed to correctly orientate it and place it in the filing cabinet.

The test consists in calculate the total error, the sum of the absolute error of all the sheets after being properly orientated.

```
Error hoja 1 -0.011108392068607031
Error hoja 2 -0.030936431229150685
Error hoja 3 -0.030014355691207584
Error hoja 4 0.032741422402878584
Error hoja 5 -0.05751952747884559
Error hoja 6 0.03661413966227656
Error hoja 7 0.00796149609406882
Error hoja 8 -0.00011861769405641098
Error hoja 9 -0.02879858201893626
Error hoja 10 -0.007652316348583099
Error hoja 11 -0.03953563939420235
Error hoja 12 -0.0017988442298815244
Error hoja 13 0.0007078351955129847
Error hoja 14 0.001930439056948785
Error hoja 15 -0.00028254223413171076
Error hoja 16 -0.017303373645802367
Error hoja 17 0.06359338688382365
Error hoja 18 -0.0013070706096414142
Error hoja 19 0.21476596378371937
Error hoja 20 -0.0026867688219880392
Error total orientació p ginas en grados : 0.5873771445442628
```

As we can see in a sample of 20 sheets the sum of existing errors was below 1 degree.

Finally, as the last scenario, we find a scenario in which we have a series of sheets with titles with different characteristics such as different sizes, numbers, strange symbols... In this case this scenario is used to perform the title test in which the correct extraction of these titles will be checked with the corresponding elimination of elements that do not provide information. To evaluate the robustness of this title, the comparison of similarity with the correct titles is applied to make the sum of the percentages of similarity error.

It should be noted that the results obtained for this test have a 1.5 percent error rate.

## Foreseen risks and contingency plan

<b>Risk #</b>	<b>Description</b>	<b>Probability</b> (High/Medium/Low)	<b>Impact</b> (High/Medium/Low)	<b>Contingency plan</b>
1	Excessive suction force.	High	High	Correctly adjust the suction valve output.
2	Inadequate height when depositing foils.	High	High	Use of the motion sensor to correctly adjust the displacement of the prismatic axis
3	Failures in the title detection system..	Low	High	Extensive screening and classification of folios by title
4	Obstruction of the robot by an object not foreseen in the working area.	Low	High	Periodic review of the work area.
5	Failure to raise the prismatic axis causing the robot to hit the sheets.	Low	High	Automatically raise the prismatic axis to a height greater than the maximum thickness of the piles of foils.

<b>Risk #</b>	<b>Description</b>	<b>Probability</b> (High/Medium/Low)	<b>Impact</b> (High/Medium/Low)	<b>Contingency plan</b>
6	Exceeding the maximum number of subjects to be graded.	Low	Low	Move the excess subject sheets to the default pile.
7	Folios without content or title to classify.	Low	Low	Move the sheets to the default pile.
8	Sheets with different date formats.	Medium	Medium	After reading the dates transform them all to the same format.
9	Undated sheets.	Medium	Low	If a sheet has no date, when using the sort function it will be deposited in a pile in the opposite place to the original one.
10	Overlapping orientations	High	High	When two sheets are stacked, if the first one is straight and the bottom one is twisted, it will detect the orientation of the second one, so it will be necessary to move each sheet to an empty area before correcting its orientation



## References

- [1] FANUC America Corporation (16 mar 2018). High Speed SCARA Robot for Pick & Place [YouTube]. <https://youtu.be/-m1oKuFkSTE>
- [2] How to Control Vacuum Pump Air Pump Suction for Robotic Arm (January 30, 2021). [Web Page]. <https://create.arduino.cc/projecthub/ronfrtek/how-to-control-vacuum-pump-air-pump-suction-for-robotic-arm-5de318>
- [3] Armesto, L. (12 feb 2020). Cómo Usar los Sensores de Proximidad para Permanecer entre Paredes [YouTube]. <https://youtu.be/j4dMnAPZu70>
- [4] Gitau, C. (2020, 11 enero). Fuzzy String Matching - Towards Data Science. Medium. <https://towardsdatascience.com/fuzzy-string-matching-in-python>