DEVELOPMENT OF A MICROFLUIDIC CHIP ACTIVATOR FOR A NEW TUBERCULOSIS SCREENING TOOL

Supervisor of the project at Epilab: Manon Giraud

July 8, 2023

Abdelsayed Vaky and Wei Yiming

CONTENTS

1 INTRODUCTION

Our design project is conducted at the Epilab, located at the Drahi center of Ecole Polytechnique. This small company aims at developing a new tuberculosis screening tool that is cheap and easy to use. This product would be useful especially in developing countries where many cases of tuberculosis emerge and remain undiagnozed.

In the beginning, we have been proposed two topics to choose from: to work on the activator that would automatically perform the experiment needed for this screening or to work on the interpreter that would analyze the outcome of the experiment and give the result (positive or negative for tuberculosis). We have decided to work on the activator since we found it is more interesting and versatile as it requires the 3D design of the activator, its assembling and the development of the Arduino program that would control the activator.

Our work was mainly supervised by Manon Giraud. Daphné Joly also helped us by giving advise and checking the files of our 3D design before printing. From the beginning of September until the end of March, we were going to the Drahi center every Tuesday afternoon (3-4 hours) to work on this project. Since the beginning of April, we were going on a weekly basis early in the morning before going to our M1 internships (one or several times a week as needed). We were also sacrificing some time at home to work on the parts of the project that could be done at distance (like the 3D design of the activator). Since September, we have been sending to Manon the protocol of each session in advance as well as the results of the previous session. These small weekly reports helped us a lot for the personal and the group reports. Since the beginning of May, the project was ended abruptly by Epilab (as Joni Frederick is aware of it) since they were in strong need to disassemble our prototype for a deadline they had.

2 Literature search about the topic

2.1 The need for a cheap and robust tuberculosis screening tool

Tuberculosis is one of the major causes of death from infectious disease in the world (WHO, 2022). About 10 million people get tuberculosis every year. A third of these are undiagnosed and untreated, which favors the transmission of the disease and the death of these untreated people. It should be noted that the number of diagnosed people has decreased even more, going from 7.1 million in 2019 to 5.8 million in 2020, because of COVID-19 (Barbier, 2022).

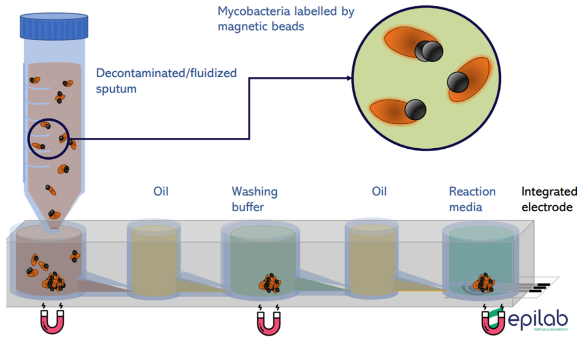
The majority of undiagnosed cases are found in developing countries where the trained people and the infrastructure needed (electric supply, good microscopes. . .) for tuberculosis detection are not present (Boyle, 2017). In addition, testing is expensive for the population in these countries. On the other hand, all the diagnostic tools developed so far are either very limited in sensitivity or very slow in detecting tuberculosis (Cruciani, 2004).

2.2 The Epilab screening tool for tuberculosis detection

In order to solve these problems, the goal of Epilab is to develop a test for tuberculosis that is robust, easy to use, portable and cheap. The test is designed to detect the complex Ag85, one of the most abundant proteins in *Mycobacterium tuberculosis* (MTB), the bacteria responsible for tuberculosis infection (Wiker, 1992).

In this test, the collected sputum is placed on a microfluidic chip composed of different wells and separated by microfluidic channels (figure 1). In the first well (at the left), containing the sputum, a solution of magnetic biomarkers (beads), specific to MT, is added. Then, a magnet is placed below the chip to attract the beads to the bottom of the first well. The magnet is slowly moved to drive the beads to the other wells, where they are progressively washed. Finally, the beads reach the last well (at the right) containing the reaction medium composed of the substrate p-AP-OG. In the presence of the enzyme Ag85, an enzymatic reaction occurs leading to the formation of the product p-AP-G (Barbier, 2022). Using an electrode integrated in this same well and a computer program, one can detect the presence of the product based on an electrochemical signal specific to the product.

Hence, if the sputum contains some bacillus of MTB, they will specifically attach the magnetic beads, and react with p-AP-OG to produce p-AP-G. This latter will be detected by the electrochemical signal which will lead to a positive test result. In the opposite case, the reaction will not occur and the test result will be negative. One can see that this test is indeed easy to realize and is portable. In addition, it is relatively performing (89% sensitivity, 79% specificity) (Barbier, 2022) compared to previous methods and it is affordable (5$/test). Therefore, it is a promising tool for the diagnostics of tuberculosis in developing and poor countries.


Mycobacteria labelled by magnetic beads
Decontaminated/fluidized sputum
Oil
Washing buffer
Oil
Reaction media
Integrated electrode
epilab
Figure 1: Schematic of the wells in the chip (done by Epilab)

2.3 The goal of our project

The goal of our project was to design and optimize the activator (figure 2) on which the chip will be deposited and which contains the magnet that will drive the motion of the magnetic beads along the chip. First, we needed to study well how the magnet should move below the chip in order to collect and drive all the magnetic beads in the chip and we needed to make the Arduino program necessary to realize this series of motions. For instance, the magnet should move slow enough to not loose magnetic beads in its way. Second, we needed to 3D design a small and portable activator.

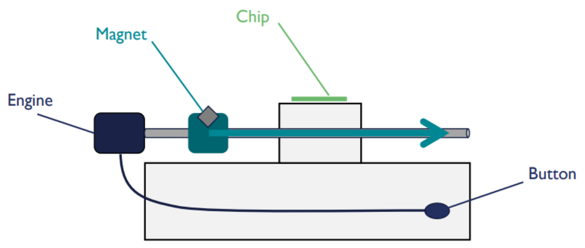


Figure 2: Schematic of the activator (done by Epilab). The activator should drive the experiment by moving the magnet below the chip to perform the tuberculosis test.

3
PROTOTYPE 1: WORK DONE ON THE PROTOTYPE PROVIDED

3.1 Introduction about the main tasks

Before beginning our project, an activator was designed and 3D printed by the Epilab team as a first prototype (figure 3). So, we decided to complete it so that it serves as a first prototype. Using this prototype, we worked on the development of the Arduino code controlling the motor and added elements that improved the user experience while using the activator to perform the experiment needed. Vaky Abdelsayed took charge of most of this task. In parallel, Yiming Wei and Jiao Zhenghao focused on the 3D design and printing of a new and better version of the activator as we will see in the following section.

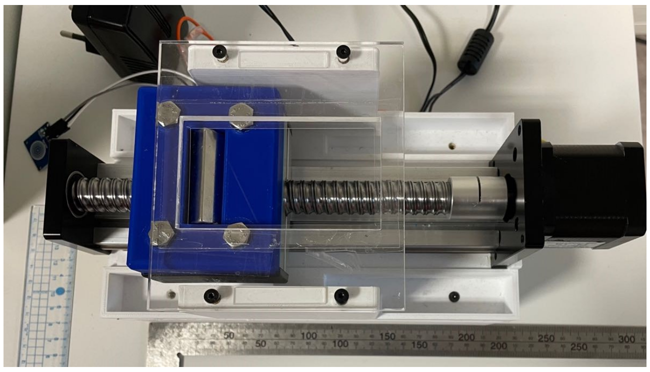


Figure 3: Prototype of the activator developed by Epilab

There were three main tasks we had to perform to complete this prototype:

- Define the optimal series of motions that the magnet needs to follow to move the beads from the first well to the last one: how much time should the magnet stay at the bottom of each well? At what speed should it move?...
- Add a sensor to define the starting position of the magnet. The magnet needs to have a reference position to "know" at which position it should start the experiment. In other words, the sensor should indicate the position of the first well to the magnet.
- Add another sensor to "sense" the presence or absence of the chip on the activator. This is important so that the magnet does not start the experiment if there is no chip on the activator. It is also important that the magnet does not go back to its starting position if a chip is present on the activator. Otherwise, the magnet would move the beads in the wrong direction (back to the first well). This part was not considered in our initial plan

(provided in December) since it can be done manually. However, as we were done early enough with the two other tasks, we considered this idea as well.

3.2 Defining the set of motions of the magnet

3.2.1 • The functions provided by Epilab needed for this task

Before we start our project, Nils Kasch (previous intern at Epilab) had developed three Arduino functions to move the magnet holder (in blue, see figure 3) in three different possible ways:

- A function to make the motor (i.e. the magnet) move forward: the function “avancer”.
- A function to make it move backwards: the function “reculer”.
- A function to make it move back-and-forth: the function “allers-retours”.

The three functions can be checked in the red part of the code provided in the appendix.

3.2.2 • What we did: September to Mid-November

First, we needed to enter all the measurements of the chip as variables in the code so that the code can be easily adapted to other versions of the chip. Indeed, the program must perform different actions according to the position of the beads in the chip. figure 4 shows how the chip looks like with the 5 wells seen previously. We have entered in the program the diameters of the 5 wells in mm with $d_{w,i}$ representing the distance of well number i (blue part of the code, appendix). We have also entered the distance between the wells i and $i+1$ in mm represented by $d_{i,i+1}$ (blue part of the code, appendix), with $d_{0,1}$ the distance between the border of the chip and the first well.

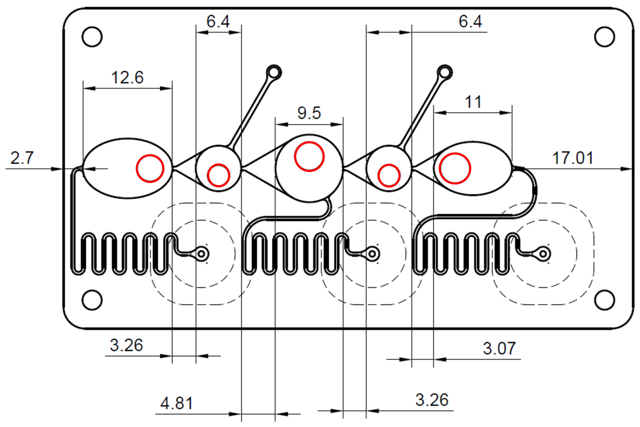


Figure 4: Scheme of the chip. The main 5 wells are represented in red (from the right to the left). The distances are in mm.

Second, we started thinking about a scenario for the experiment to determine a preliminary series of motions for the magnet:

- The magnet would first wait for some time at the first well to collect the beads.
- After that, it would start going back-and-forth from the beginning to the end of the first well to facilitate and accelerate the collection of the beads and their attraction to the bottom of the well.
- Then, it would slowly move to the second well which contains oil to separate the beads from the liquid solution (the sputum). The magnet would also move back-and-forth to "shake" the beads all over the well and make sure they get separated from the rest of the sputum.
- Afterwards, it would move forward to the third well containing the washing buffer and would go back-for-forth all over the well to wash the beads from any unwanted particles remaining from the sputum.
- Then, it would go to the fourth well containing oil and do as previously described.
- Then, the magnet would go to the last well and move back-and-forth to spread the beads all over the electrode surface (to make the detection more efficient).
- Finally, after detection of the result on the electrode, the magnet should go back to the first well to start a new experiment.

We started with this preliminary scenario which is simple but already contains many parameters: the number of times the magnet would go back-and-forth in each of the 5 wells (5 parameters) and the speed at which the magnet moves (1 parameter). We implemented this scenario using Nils' functions as shown in the code (see pink part, appendix).
We did some tests to optimize these parameters. To not waste expensive solutions (such as the artificial sputum or the washing buffer), we put water containing blue ink in wells 1, 3 and 5 and oil in wells 2 and 4. Then we added 1 ml of TB-beads, magnetic microbeads relying on hydrophobic interaction to capture MTB, in the luer tube connected to the first well (see figure 1 for a reminder).
We did this experiment several times to start optimizing the 6 parameters. We decided to set the first 5 parameters to 20 back-and-forth movements, except for the third well which we set to 60 since we would like to make sure that the beads are washed well. These values were found already efficient at moving most of the beads to the last well. We also set the velocity of the motor to 1.5 seconds per one revolution (which corresponds to a step of around 5 mm) since a lower velocity made the magnet lose many of the magnetic beads on its way.

3.2.3 • Real experiment

After doing this quick optimization, we were able to show that, with the parameters we set, most of the magnetic beads were following the magnet, passing through most wells and all the way up to the last well as we can see in figure 5: the beads are clearly seen in black at the last well, on the electrode. It should be noted that a different older version of the chip is used in this experiment since the newer version of the chip was not available and is still being designed. We just changed the dimensions needed in the code to perform this experiment. But it would work exactly the same with the newer version of the chip shown in figure 4

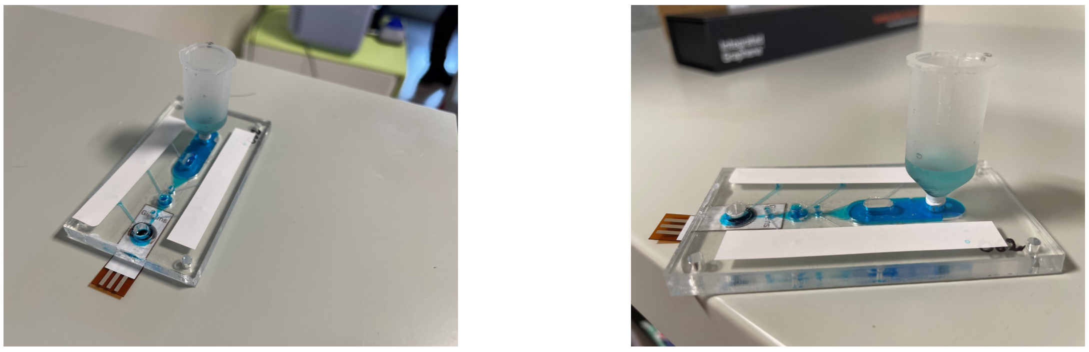


Figure 5: Real experiment with the new parameters.

3.2.4 • Future step

In the future, we will of course need to do the experiment with artificial sputum to test if the magnetic beads need to spend more time in one of the washing wells in case the positive signal is not clear enough indicating that the beads were not washed properly for instance or that they needed more time to reach the bottom of the well due to viscous forces in this matrix.

3.3 Adding the two sensors to the prototype

3.3.1 • Choosing the right sensors: November

To add the two of sensors to our model, we first needed to decide which type(s) of sensor would be suitable for the tasks we want. To specify the starting position, we thought about a simple sensor that would be fixed at a specific position such that when the magnet holder goes back, it would stop once it touches (or presses) this sensor. Hence, we thought about two types of sensors: a capacitive touch sensor, which is activated when an object touches it, or a mechanical sensor which is activated when it is pressed (like a button). We found cheap and small sensors from these two types and we asked Epilab to order them for us (references: capacitive touch sensor and mechanical sensor). We also ordered a magnetic hall sensor which could be a good idea for the first task (reference: hall sensor) since we already have a magnet in our setup. We thought about using the sensor to "sense" the position of the magnet and stop the magnet holder at the right starting position. It should be noted that this sensor never came (delivery problem) so we used the two other sensors which worked fine.

3.3.2 • Activating the sensors: December

Once the sensors were delivered, we started reading about the capacitive touch sensor using the different information indicated in the reference website. However, we needed to search more about the functioning of the sensor and how to activate it. We watched this tutorial and used the same code provided to activate the sensor. We were indeed able to activate it such that once the sensor is touched, it is activated. Then, we replaced the capacitive sensor by the mechanical one and we were able to activate (resp. disactivate) it with the same code by pressing (resp. releasing) the sensor switch.

3.3.3 • Making a theoretical graph of the new code: January

To control the functioning of the sensor using the existing Arduino code, we needed to be clear about which parts of the code would need to include the sensors and what the sensors would be doing. A description of what the current state of the code is doing was provided previously in section 3.2.2. In the current code, when the button of the activator is pressed, the magnet holder starts all the steps mentioned previously (it waits for some time at the bottom of the first well, then it starts moving back-and-forth...). After the magnet holder finishes all the steps mentioned, it goes back by the distance needed to go back to the first well (distance between the first and fifth well). This is not accurate and not robust: it may drift slowly with time. Since we might need the magnet to go back from a different position than the last well for instance. For example, if we want to stop an experiment in the middle while the magnet is at the third well, we would need the magnet to go back by a different distance to reach to first well. So, we really need a sensor to indicate the starting position (sensor 1).
We also need to "sense" the presence or absence of the chip on the activator using a second sensor as previously explained (sensor 2). Therefore, we made the theoretical graph shown in figure 6 to imagine and set more clearly what the code will be doing exactly after the addition of the two sensors. According to this graph, when the button is pressed, the magnet would wait for a signal from sensor 2 indicating that there is no chip on the activator. Then, the magnet would go back until the magnet holder touches sensor 1 (in case the magnet is not already at the starting position). After that, the magnet would wait for a signal from sensor 2 indicating that a chip was placed on the activator. Then, the magnet would do the whole experiment until it reaches the last well and would wait there for a signal from sensor 2 indicating that the chip was removed from the activator. After that, the magnet could safely go back to start a new experiment and so on. The button can be used to start or stop the activator at any point we want during this process.

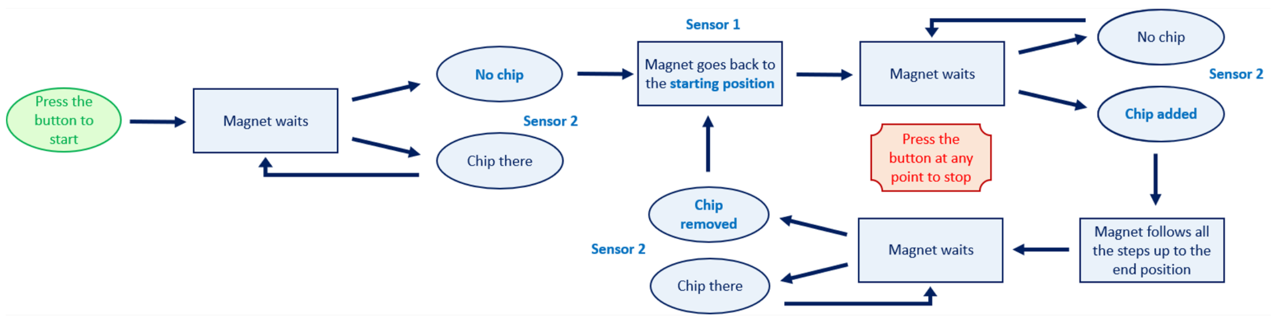

Press the button to start
Magnet waits
No chip
Chip there
Sensor 2
Sensor 1
Magnet goes back to the starting position
Magnet waits
No chip
Chip added
Sensor 2
Press the button at any point to stop
Magnet follows all the steps up to the end position
Magnet waits
Chip removed
Chip there
Sensor 2

Figure 6: Graph showing the role of the two sensors in the code.

The button is already present on the activator and is controlled by the function "Check-Button" also coded previously by Nils Kasch (red part of the code, appendix). Our goal was to add the two sensors to complete this graph.

3.3.4 • Addition of the sensors to the code: January to March

Addition of sensor 1

We have decided to use the capacitive touch sensor for now since it can be activated the same way as the mechanical sensor. We started by using the capacitive sensor since it was more sensitive in principle and could thus be better at the detection. Then, we would decide later which sensor is best for this task
Since sensor 1 needs to interact with the motor we adapted the "CheckButton" function to make the sensor stop the motor every time the sensor is touched. We named this new function “CheckSensorc” (brown part in the code, appendix). So every time the magnet holder touches the sensor, this latter stops the motor and then re-activates it to make it move forward to start an experiment from the right starting position.
However, this task was not straightforward and needed to be done on several weeks since the addition of this function made us unable to stop the activator with the button due to some interference between “CheckButton” and “CheckSensorc”. Finally, we were able to identify the parameters that were used in common between the two functions and we decided to redefine separate parameters for each of the two functions to make sure there are no interference (separate parameters defining the current state of the motor, separate parameters for the previous state. . .) and we were able to control sensor 1 with our Arduino code without problems. The correct versions of the two functions are the ones provided in the appendix. A short video attached with the report shows the functioning of “CheckSensorc” (title: "functioning sensor 1"). It shows the magnet holder stopping once we touch the sensor and then moving forward from the position at which it stopped.
To use the sensor 1 correctly, one last step remained: the sensor should be set on the activator at the right position. Thus, we took this into account in our second prototype as we will see in the next chapter.

Addition of sensor 2

Description of the tasks

The addition of sensor 2 was a harder task for two reasons:

- First, it would be responsible for several tasks and not just one. Indeed, it should both check for the absence of the chip before the magnet goes back and for the presence of a chip before allowing the program to start. Finally, in case the user removes the chip from the activator at any point during the experiment, the sensor needs to detect it so it can reset the activator, putting it back to its starting configuration.
- Second, it is not similar to the button function (like sensor 1) since it would not just be pressed or touched once for activation. It would be pressed during the whole time a chip is present on the activator. In other words, the sensor should continuously check for the presence of a chip to give a signal to the magnet holder to go back once a chip is removed. Similarly, the sensor should continuously check for the absence of a chip while there is no chip so that a signal is sent to start a new experiment once a chip is added to the activator (see figure 6 for a reminder).

To solve the first difficulty, we decided to clearly define the different tasks this sensor will be performing. To solve the second difficulty, we decided to theoretically imagine how to implement the “while loop” necessary to perform the corresponding task by continuously checking for the presence or absence of the chip. The description of the three tasks are provided in table 1.

	Location	Descriptions
Loop 1	Button function (when button pressed)	While chip there: wait
Loop 2	Sensor 1 function (when magnet sensor pressed)	While chip not there: wait
Loop 3	At the end position	While chip there: wait

Table 1: Description of the loops of sensor 2.

As we can see, the first task would be to wait while a chip is on the activator. This task should be performed right after the button is pressed so that the magnet waits in case a chip is on the activator (to not go back and ruin the experiment). Once the chip is removed, the magnet can safely go back to the starting position to start a new experiment. The second task consists in making the magnet wait once it reaches the starting position. The magnet should wait until a chip is added so that it can start the experiment. This task would be performed right after the magnet goes back to the starting position indicated by sensor 1. The final task would be to wait once the experiment is done (i.e. the magnet reaches the last well) until the chip is removed. After that, the magnet can safely go back to the starting position to start a new experiment.

Implementation of the tasks

Having split the tasks with such a detailed description, it became clear to us that we basically needed two functions to control sensor 2. On the one hand, we implemented a function consisting in a "while" loop to make the magnet wait while a chip is there. This function can just be called once the button is pressed to perform task 1. It should also be called once the magnet finishes the experiment to perform task 3. On the other hand, we implemented a function that does exactly the opposite: it makes the magnet wait while a chip is not on the activator. This function would be called once the magnet holder touches sensor 1 to perform task 2. We implemented the two functions which correspond respectively to "CheckSensorm" and "CheckSensormopp" (brown part in the code, appendix). In addition, the video attached with the report (title: "functioning sensor 2") shows the functioning of these two functions on the mechanical sensor, which is the one we chose for this task.
The addition of this second sensor was not straightforward since the implementation of such while loops that would continuously (each second) check for the presence or absence of a chip led to many errors. We had to search for specific information about how to exactly implement what we want and we realized that in the definition of the setup section (in the code, see appendix), the sensor input should be set to "INPUT-PULLUP" and not just to "INPUT" (as previously used for sensor 1 and the button), which is not precise enough for such long loops. It took us a few sessions since we needed to read more about Arduino coding and try different possibilities to solve this error and other errors that we had in the code.

3.3.5 • Future task

As in the case of sensor 1, we have been able to prove the functioning of sensor 2 with our hands. We just need to include it in the activator below or next to the chip so that it can sense the chip on the activator. We considered this in our second prototype.

4
PROTOTYPE 2: DESIGN OF A NEW ACTIVATOR

As we will see, there were many things that we wanted to improve in the prototype provided by Epilab. So, we decided to design our own activator to include these improvements and to include the two sensors that we connected to the activator. Wei (and Jiao until he left in December) took charge of making a new design and did most of the work in this part.

4.1 Softwares and machines used

At the beginning of our project, Gareth Paterson spent an afternoon introducing us to 3D modeling, 3D printing and laser cutting. He showed us the softwares to use and the printing machines at the Drahi center and he made us do small tests to make sure we understood. In the following, we will go over the tools to which he introduced us and that we used for our 3D modelling.

4.1.1 • 3D modeling

First, regarding 3D modelling, the two softwares commonly used are SolidWorks and Creo. They are parametric solid modeling softwares used to 3D model parts, assemblies or complex models They have user-friendly interfaces offering a wide range of options which make them easy to use for beginners. Hence, these two softwares were quite straightforward to use, in addition to the fact that Wei and Jiao had some previous experience with these programs. Thus, we have mainly used these two softwares to design the different pieces of the new prototype.

4.1.2 • 3D printing

One of the most popular 3D printing software programs is Cura. Cura is an open-source software that converts 3D models into printable objects by slicing the model into layers and generating a set of instructions for the 3D printer to follow. It also offers a wide range of options for printing, like the addition of custom support structures, infill patterns, print speed control...
It is advantageous since it can be used with many different 3D printers. Like Creo and Solidworks, it has a user-friendly interface that is easy to understand by a beginner and to use for customization of printing settings. Hence, we have also worked with Cura for 3D printing once the model of the different pieces of the activator was done. To print a given piece, the protocol to use Ultimaker Cura was straightforward and can be checked in the appendix.

4.1.3 • Laser cutting

We also needed to use laser cutting for some pieces of the activator as we will see later. We performed laser cutting using the Trotec engraver provided at Drahi. We have used CorelDraw, a graphics design software that is commonly used in 2D, 3D printing and laser cutting. To print a given piece of the activator using Trotec engraver, the protocol used was also straightforward and is provided in the appendix.

4.2 Our new model

Let us start by showing the old model provided in the beginning. Then, we will talk about the improvements we thought of. Finally, we will present how we made a new model to include these improvements and how we printed it.

4.2.1 • The old model/prototype provided

Here are the original model (figure 7) and prototype (figure 8) provided by Epilab.

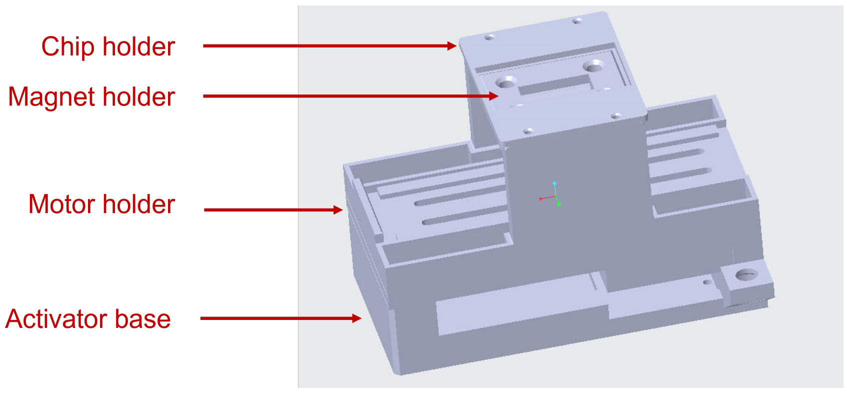


Figure 7: Model of the old activator

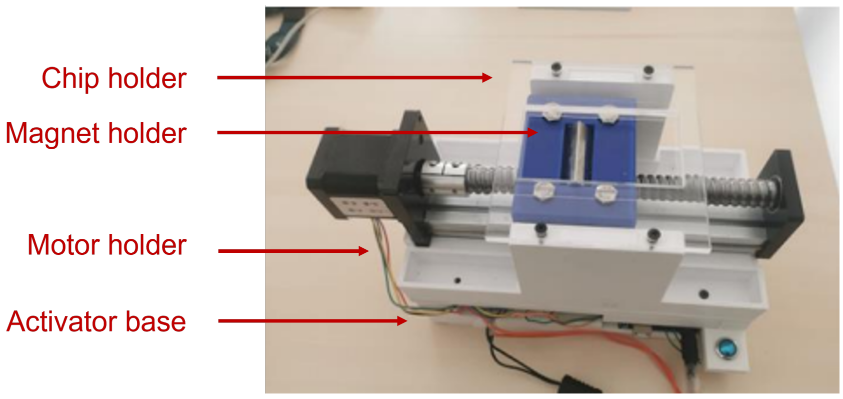


Figure 8: Picture of the old activator prototype

This model includes 4 main pieces (figure 8):

- The base of the activator (at the bottom) containing the motor driver, the Arduino board and the wires as well as the button (blue circle) that turns on the activator.
- The motor holder right above the base. This piece serves as a base for the motor which is assembled with it. This piece has an extension from the right and the left to hold the chip holder.
- The chip holder (the part in transparent at the top) placed on the motor holder.
- The magnet holder (in blue) that contains the magnet and that is placed on the motor to move the magnet below the chip.

4.2.2 • List of the improvements: December and January

Table 2 shows the ideas we got to improve the original model as well as the solutions we proposed to implement each idea in our new model.

Improvement	**Solution proposed**
1- Find the optimal magnet and design a new magnetic holder with the corresponding dimensions.	- Do experiments to find the optimal magnet. - Take its dimensions and design the magnet holder accordingly.
2- Make the model not magnet dependent in case we want to replace the magnet by a more optimal one at some point.	Add pins to the chip holder in a "lego" style so that it can be placed at a higher distance from the magnet holder. In this case, the only part that would need to be re-designed is the magnetic holder, based on the dimensions of the new magnet.
3- Make all the connections go from inside to avoid having wires outside the activator.	Add empty holes in the different pieces of the activator so that all the connections can fit inside.
4- Cover the top of the activator to start making the prototype look like a final product.	To avoid adding a piece, we thought of extending the chip holder to make it cover the top of the activator.
5- Add sensor 1.	- Measure at which position the sensor should be placed. - Decide which type of sensor would be used. - Add a way to be able to glue the sensor at the position needed from the magnet holder depending on the design.
6- Add sensor 2.	- Measure the dimensions of the mechanical sensor. - Extend the chip holder to have an additional place to glue the sensor next to the chip such that the switch is facing the chip and is pressed when a chip is added.
7- Make the model smaller and lighter to make it more portable.	We can think of making the base hold the motor and the Arduino board at the same time to reduce the design height.

Table 2: List of improvements to be made in our new prototype.

4.2.3 • First idea proposed: January

We took the second, the fourth and the seventh improvements proposed as our starting points since they require the biggest change to the whole design . We proposed a first design that includes the proposed solutions (figures 9 and 10). First, on the purple piece (part to which the arrow is pointing, figure 9), we can see a place to hold the chip. This chip holder is extended into a cover for the activator which realizes our fourth idea. It should be noted that we thought of splitting the "cover" in two parts: the piece shown in purple containing the chip holder and another piece (left piece in figures 9) which could be used in the far future to add a small screen that would serve as a small interface to interact with the manipulator. Second, we can see in figure 10 how the two parts of the cover can be assembled with the base in a "lego" manner which realizes our second improvement. In addition, we can see that the base is made wide enough to contain the motor (the part added in the middle of the base in the right figure of figure 10) and the Arduino board and the motor in the remaining space below the motor. This idea of combining the motor holder with the base containing the Arduino board, the motor drive and the wires is thus realizing our seventh improvement.

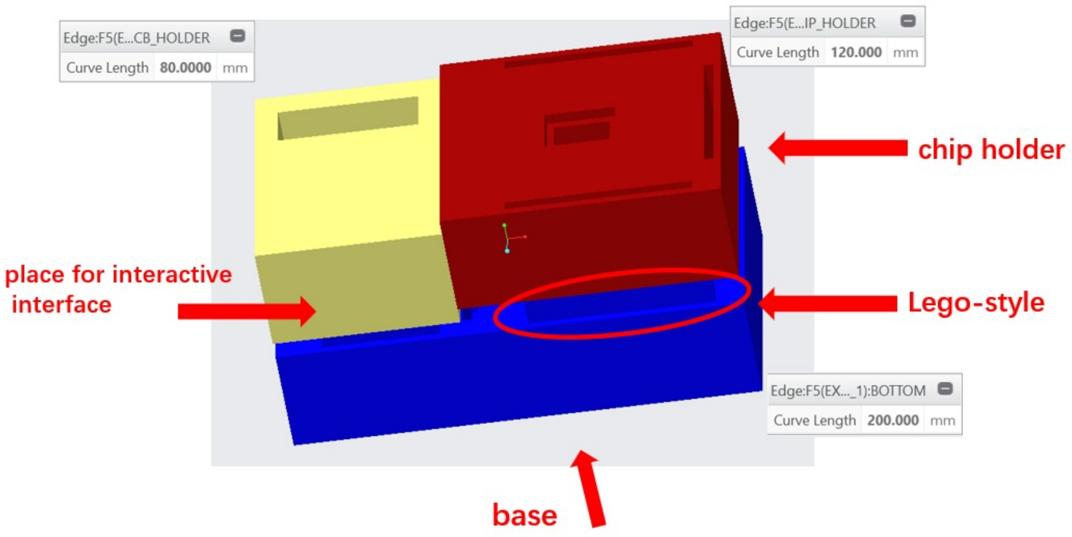


Figure 9: Lego-style model

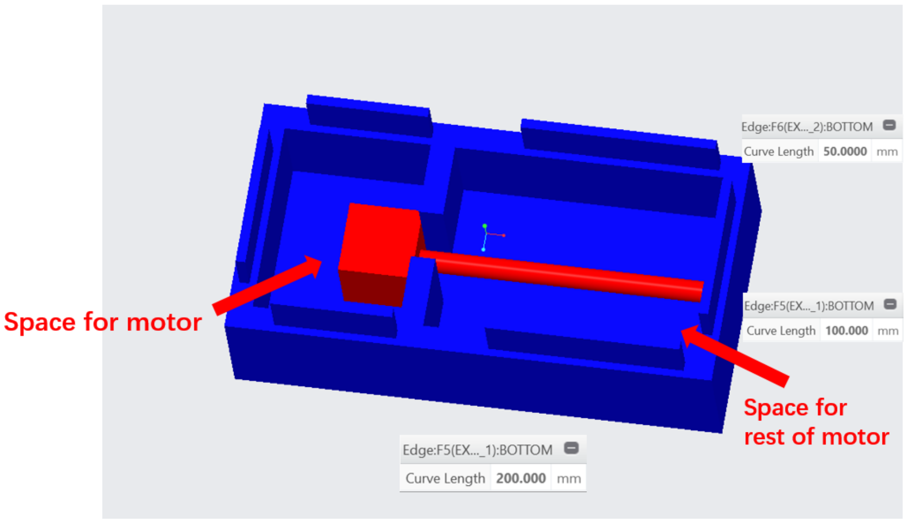


Figure 10: Lego-style model-Inside

4.2.4 • New modified model: February

However, when we started taking exact measurements, we realized that if we place the Arduino board, the motor drive and the wires next to the motor, we would need the base and the whole model to be much wider or longer compared to the original prototype. So, we are not really making the model smaller and lighter as wanted. Therefore, we decided to modify the original idea by modifying the design of the base.
Instead, we thought of having the Arduino board and the motor driver placed next to the motor but in a vertical configuration. In this case, their height would be smaller than that of the motor, so the base will still have the same height as that of the motor like in our previous idea. So, we would get a gain in the height (smaller) compared to the original prototype of Epilab. In addition, in such configuration, the motor driver and the Arduino board would just make the base slightly wider than the first prototype as we will see in the following. Hence, the gain in the height (smaller compared to the original prototype) would clearly surpass the loss in the width (slightly wider) as we will see in the following section.
Having this idea in mind, we started taking the measurements necessary to design our model.

4.2.5 • Steps followed to design the new model: February and March

In this part we will provide a detailed description of the steps we followed to measure the different components of the activator and to design our model. The exact measurements are provided in the CAD file containing the design and that is attached with this report. This file can be opened using different 3D design softwares (Solidworks, Fusion...).

Design of the base
As we can see in figure 11, the model of the base presented on the right shows our idea where the base is mainly split in two parts. The part at the bottom of the right figure is dedicated to the motor while the part at the top is dedicated to the Arduino board, the motor driver and to place the wires. This latter is split in three parts since we thought it is better to have a separate compartments dedicated to the motor driver, the Arduino board and an additional place in case an additional Arduino board is needed after for instance. We have also opened a part between the two main parts of the base so that the wires can go from the motor driver to the motor.
Since the motor driver is the biggest element that would be placed in the upper part, we started by checking its measurements shown on the left of figure 11 (reference:motor driver measurements); and accordingly, we adapted the dimensions of the upper left compartment as shown on the right part of the figure.
Then we adapted the dimensions of the second compartment to that of the Arduino board. Finally, as mentioned, the third compartment was left in case we need to add another Arduino board or anything additional to the setup in the future. In this extensional part, we have also added a hole facing the outside of the base so that it serves as a room for the button of the activator.
All the details and measurements can be checked in the CAD file of the design.

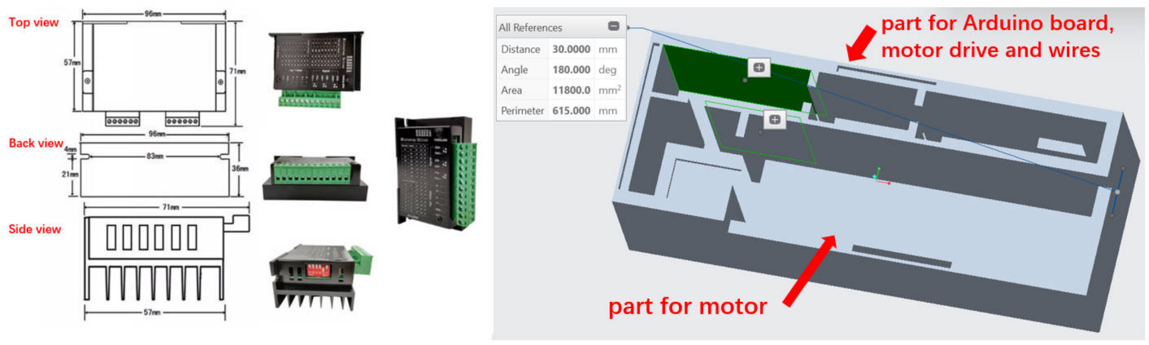


Figure 11: Measurement of the motor drive and its position

After that, we moved to the measurements of the motor to adapt the dimensions of the lower part. We checked the measurements of the different parts of the motor on the internet (reference: motor measurements). As can be seen on the left of figure 12 and on the real image on the top right, the motor is mainly composed of four parts: a piece at the beginning, a piece at the end, the axis of the motor in the middle (on which the magnet holder will move) and a cube attached to the axis and that can move on the axis when the motor is active. This is the part on which the magnet holder is placed. We have checked the dimensions of these different parts and completed them as represented in red (in mm) on the left of the figure. We also had the idea of adding a "barrier" to fix the motor without the need to "glue" it on the setup, which would be practical in the future for disassembling the motor from the activator (for example, if we want to use the motor for a another model or prototype). Thus, we thought of adding the part represented in red in the design so that it can fix the motor of our prototype by holding the black rectangle contained in the first piece (blue arrow, figure 12).

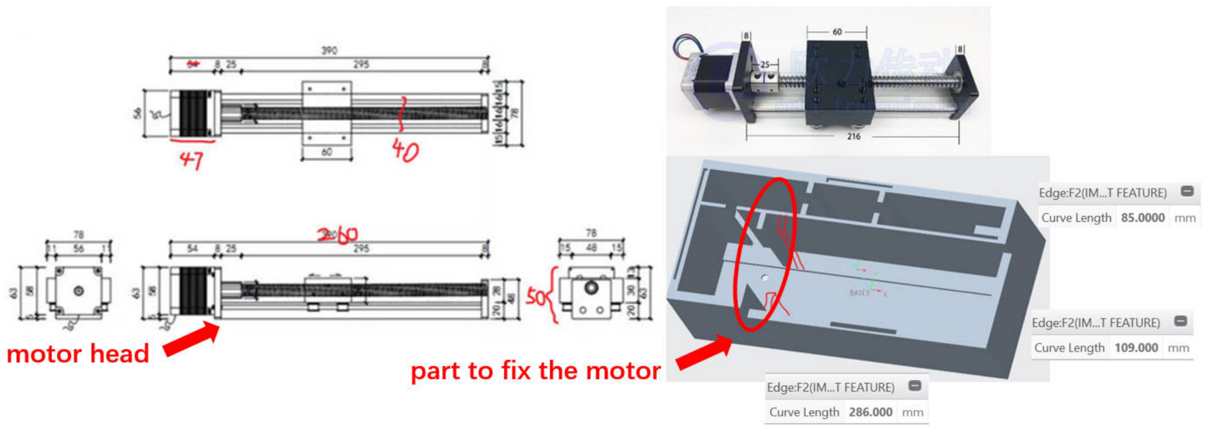
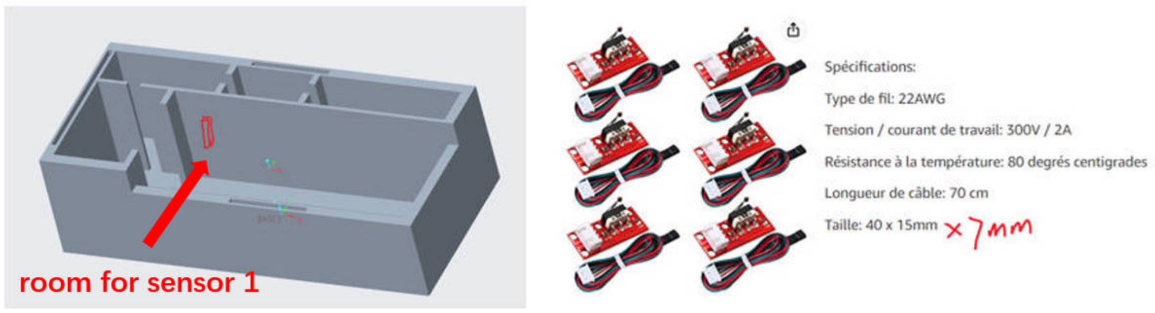


Figure 12: Measurement of the motor (in mm) and its position

Then, to add sensor 1, we thought of just adding a hole between the two parts of the base as represented in red on the left of figure 13. The sensor would be glued so that it is facing the magnet holder such that when this latter goes back and touches the sensor, the motor would stop at the right position corresponding to the starting point. This hole is also an optimal place since the sensor needs to be connected to the Arduino board through 3 wires so it needs to be close to the board. We have decided to use the mechanical sensor since we found that the capacitive sensor is sensitive from both sides (it is activated if touched from any side) which might make it too sensitive if touched by anything during the experiment which might lead to errors. So we checked the dimensions of the mechanical sensor and we also measured its height (in red) to adapt the dimensions of the hole.



Spécifications:

Type de fil: 22AWG

Tension / courant de travail: 300V / 2A

Résistance à la température: 80 degrés centigrades

Longueur de câble: 70 cm

Taille: 40 x 15mm x 7mm

Figure 13: Position of sensor 1 and measurements of the mechanical sensor.

Finally, as can be seen in figures 12 and 13, we have added small holes at the top of three sides of the base to realize our "lego" like idea.

Design of the cover
Then, we moved to the second piece of our design: the cover containing the chip holder. To make it easier for printing, we divided the cover in two parts (figures 14 and 15) that will respectively cover the part of the compartment at the bottom right containing the motor axis (figure 11) and cover the rest of the base. For the second piece, the dimensions were just adapted to that of the compartments it needs to cover. In addition, the "lego" like pins were added at the right positions to fit with those on the base. For the chip holder part, as can be seen in figure 14, there is a place to hold the chip. So we measured the dimensions of the chip to design this part. We have also added a hole in the center of the chip holder as can be seen in the figure, so that the magnet can face the chip without obstacles. We made sure this hole is bigger than the part of the chip that will undergo the experiment which basically corresponds to the part all the part from the first to the last wells. Then, we added an extension of the chip holder (at the forefront of the chip holder, figure 14) that has the same dimensions as the mechanical sensor so that we can place sensor 2 at this level. Sensor 2 would be placed so that its switch is facing the chip such that the sensor is activated (pressed on) while a chip is on the activator. Finally, we have added the pin needed to assemble this piece with the base.

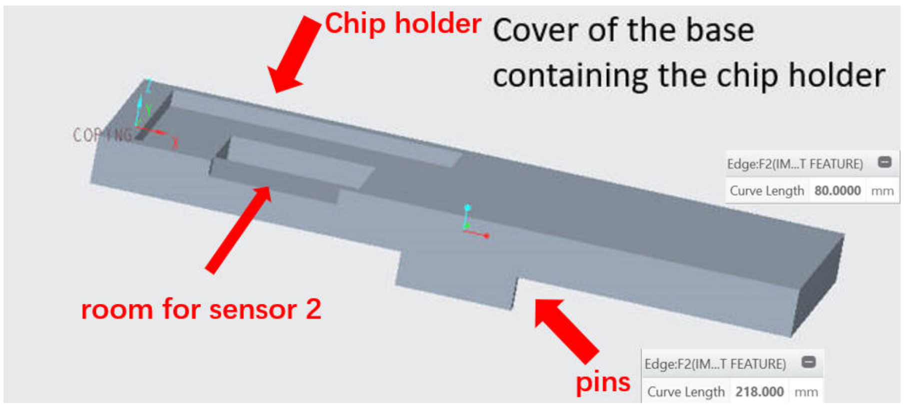


Figure 14: Measurement of chip and its position

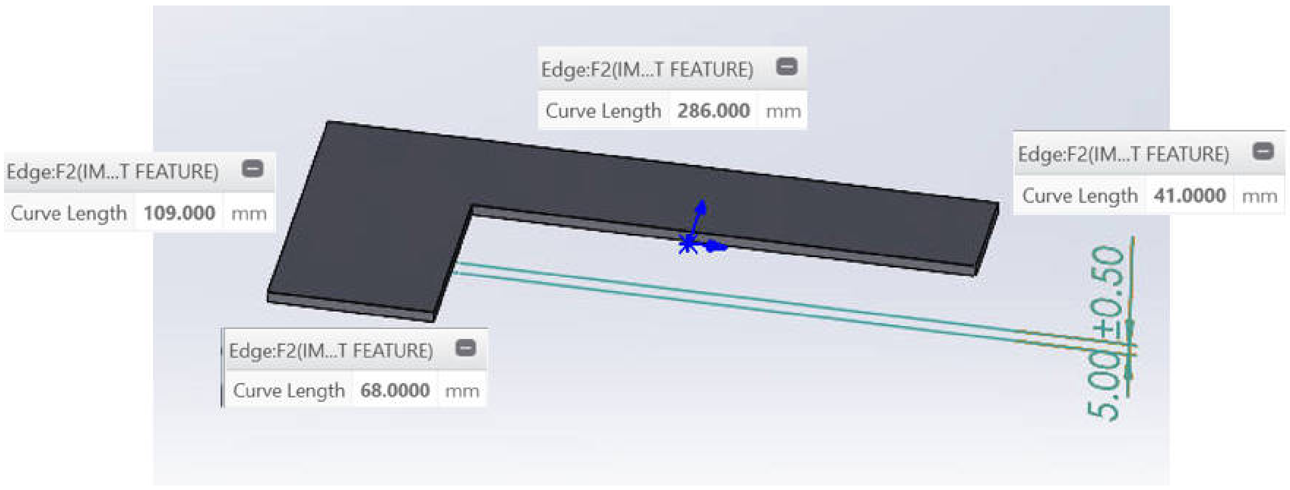


Figure 15: Cover of the rest of the activator

Design of the magnet holder

After that, we designed our last piece: the magnet holder. First, we needed to check which magnet is optimal since Epilab had several magnets that were not tested before. We did a few experiments to find this optimal magnet. We put a luer tube containing 1 ml of TB-beads on a small PMMA piece to close the luer tube from the bottom. Then we placed the luer tube on a magnet and calculated the time the beads needed to go to the bottom, i.e. the time at which the bottom appears black and the liquid becomes almost transparent. The four magnets tried, their dimensions, the different configurations tried and the time taken for the beads to go down are indicated in figure 16. We found that the fifth and sixth experiments were the optimal ones. So we decided to choose the last magnet, which has a height of 60 mm and a square base of width 15 mm, since the fifth experiment consisted in a configuration that would clearly need a much bigger and higher magnet holder. Then, to design the new magnet holder, we decided to only have two screws on the magnet holder (instead of 4) so that we can have enough place for this new magnet. The new magnet holder is presented in figure 17.

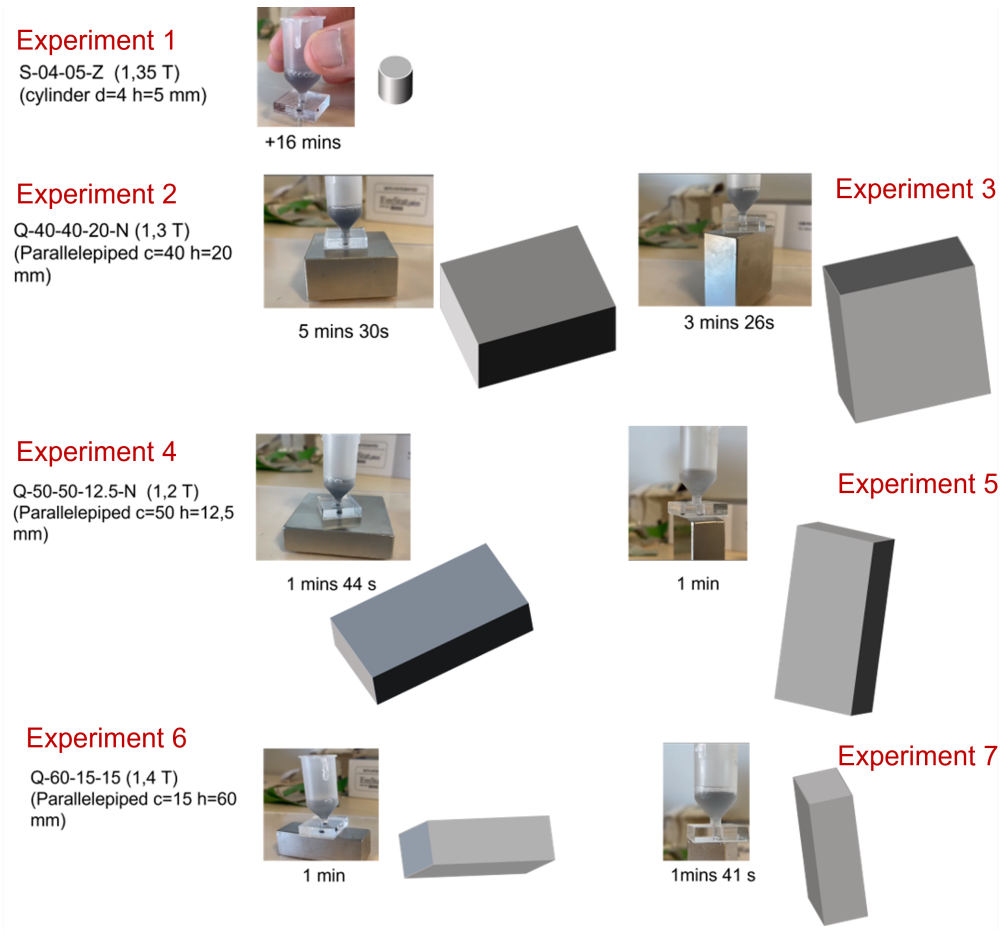
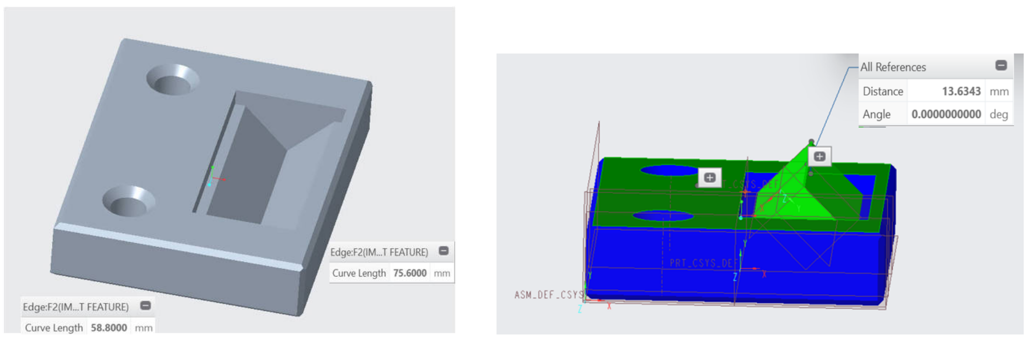


Figure 16: Experiments to choose the optimal magnet


Edge:F2(IM...T FEATURE)
Curve Length 75.6000 mm

Edge:F2(IM...T FEATURE)
Curve Length 58.8000 mm

All References
Distance 13.6343 mm
Angle 0.0000000000 deg

Figure 17: New Magnet Holder

Looking at the table of improvements we proposed 2, we can clearly see that the seven improvements proposed were taken into account in this new design.

4.2.6 • Final modifications and printing the pieces: March and April

After finishing our design, we consulted Daphné Joly before printing. She gave us some advise about the technical details needed for printing the different pieces. For instance, the walls should be thick enough for the prototype to be robust but they should also be thin enough to make the prototype as light as possible. A 3 mm thicknesses is thus a good choice for most of the "walls" for example. We also needed to minimize all the curvatures present in our three pieces to minimize the printing time and make the model less complex. She modified such details and sent us back the final version of the CAD file that is attached with our report and which we used for printing. Finally, we rechecked quickly all the measurements and moved to 3D printing.

- First, we printed the magnet holder. The final design of the magnet holder is shown in figure 17 and was made using Creo. Then, we placed the magnet holder on the motor (which is still on the old prototype) (figure 18). We could see that it was fitting the part dedicated to it on the motor very well. In addition, our chosen magnet could also perfectly fit inside it.

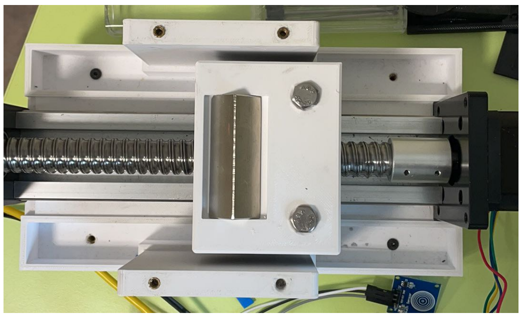


Figure 18: Picture of the new magnet holder added to the old prototype

- Second, we printed the base. The final version of the base is provided in figure 19 which was modelled using Solidworks. We modified our original idea of having an additional compartment in the part presented at the top. We decided to only have two compartments for the motor driver and the Arduino board so that there is enough room for the wires that surround them.

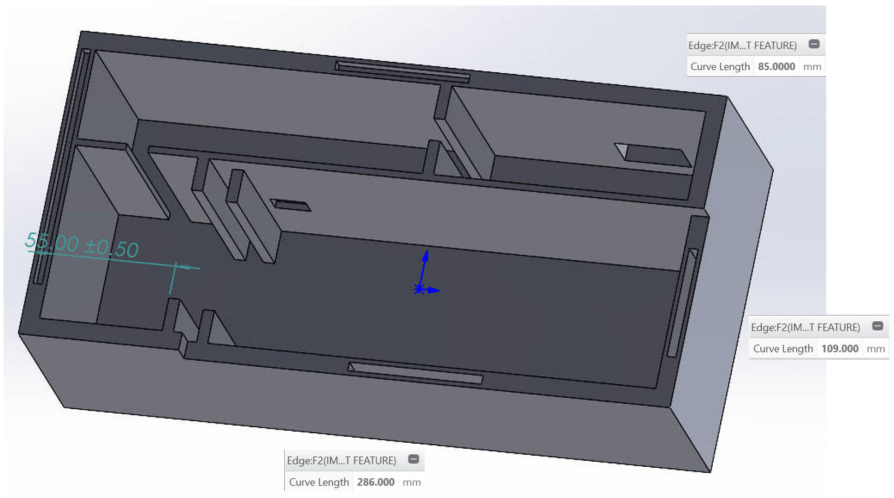


Figure 19: Final model of the base

- We also printed the button that will be fixed in the base. For this we used the same design as the one of the old prototype and printed the button using a 3D printer as well.
- Then, we imported the model on CURA software, which was proposing more than 24 h for printing (figure 20). In order to reduce this time without affecting the quality of the base, we decided to use segment printing, meaning that the lower part of the model would be printed with a higher density compared to the upper part. This is not a problem for the quality since it is the lower part that serves as a support for the base so the density of the upper part can be decreased. Therefore, we were able to optimize the printing time which was reduced to 16 h. The printed base can be checked in figure 21.

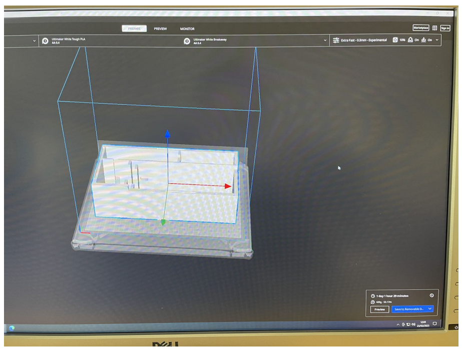


Figure 20: Original time to print the base

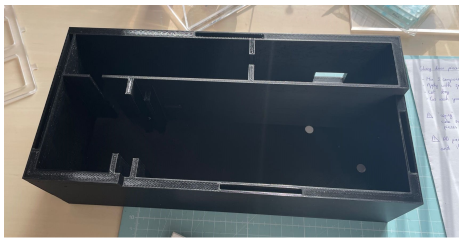


Figure 21: Picture of the printed base

- Third, we printed the cover. The final model of the cover is provided in figure 22. It was done using Solidworks. Both pieces consist exactly in what we have seen before. However, the piece on the left contains 2 minor additions. We added a small hole (top right of the figure) so that the wires of the sensor can go through. We also added a small extension (at the right of the chip holder on the figure) so that it is easier for the person doing the experiment to put the chip on the activator (by placing his finger in this extension).

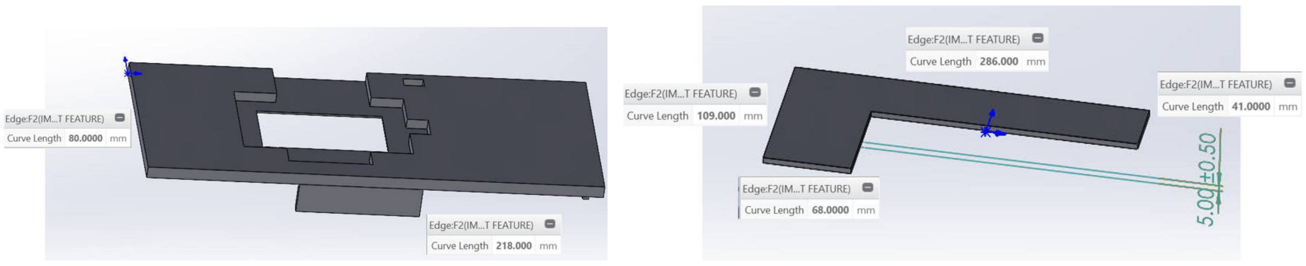


Figure 22: Final design of the cover

To manufacture these two parts, we found that 3D printing is not optimal because it would take a very long time due to the amount of small holes and details we have in these pieces. Therefore, we decided to use laser cutting. To do that, we had to split each of the pieces in different layers and then to glue them after they are cut. Taking the right piece as an example (figure 22), we divided it in two layers, a layer of 3 mm (figure 23) and a layer of 2 mm (figure 24) that contains cavities for the pins to be inserted in them. We cut the pins as well. After cutting these two layers, we glued them together and we inserted the pins in the cavities and glued them. As can be seen, here are pictures of the two parts of the cover printed (figures 25 and 26).

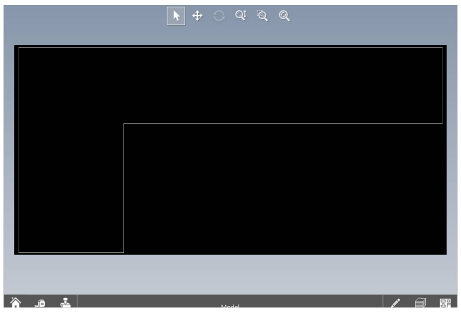


Figure 23: First layer

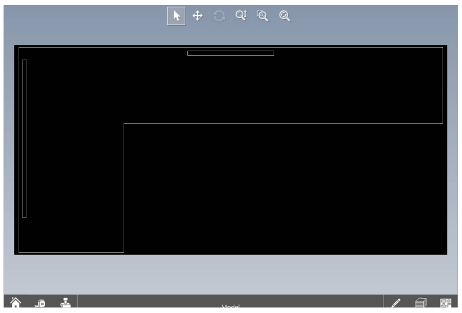


Figure 24: Second layer

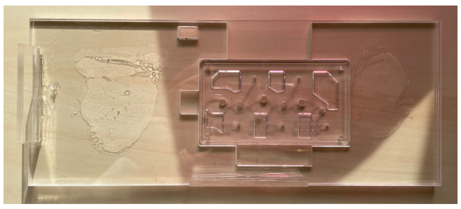


Figure 25: First piece of the cover printed

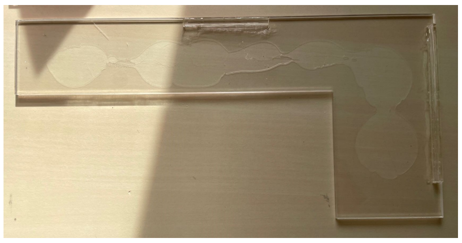


Figure 26: Second piece of the cover printed

4.2.7 • Assembling our prototype: end of April

Having printed all the parts of our new prototype, we moved to assembling them. Unfortunately, as Joni knows, we were contacted by Epilab at the beginning of May and were told that we cannot continue working on the project anymore since they got an urgent deadline and needed parts of our prototype for it (magnet, motor...). So we were stopped from finalizing our prototype. We will present here how we started the assembling and will also present the last few theoretical steps that we were planning to follow to finalize assembling and have a final prototype.

As a first step, we needed to disassemble the old prototype to get the motor, the motor driver and the Arduino board including the different cables. We thus unscrewed all the screws in the old prototype to get these components. Then, we had to weld some of the cables connected to the Arduino board and the motor driver since some of them were temporarily held together using tape in the old prototype, which is very unsafe. This took some time and we needed the help of Théo Dubois (intern at Epilab). Having done that, we placed the motor driver and the Arduino board in their dedicated compartments in the base of our new prototype. After that, we placed the motor in the base of the prototype. Then, we had to glue the sensor that would serve as a reference point for the starting position of the motor at the hole that we have dedicated to it as can be shown in figure 27. We also needed to screw our new magnet holder, containing the magnet, on the motor (figure 28).

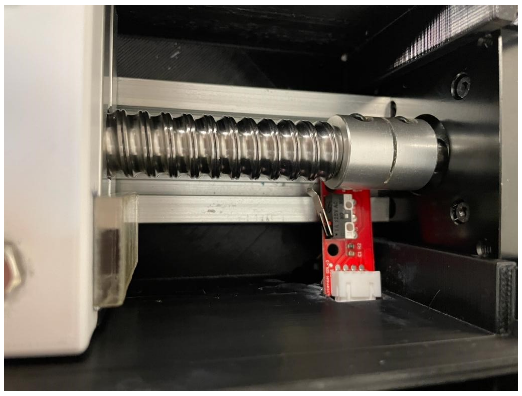


Figure 27: Gluing sensor 1 in the base.

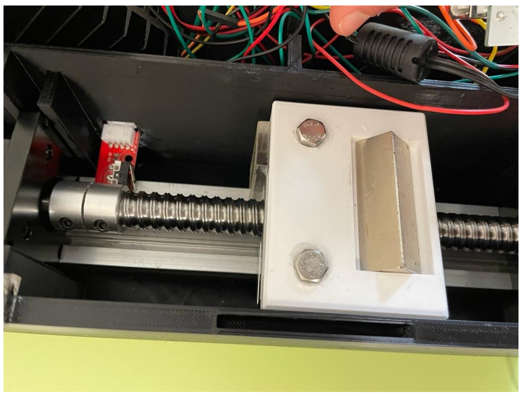


Figure 28: Addition of the magnet on the magnet holder.

On the other hand, we glued the chip sensor next to the chip holder on the cover. Since it was not well fixed, we also added a tape so that it is fixed well at the right position. Figure 29 shows the sensor placed next to the chip holder when a chip is not there (left) and when a chip is added on the activator (right) in which case the sensor is pressed and activated.

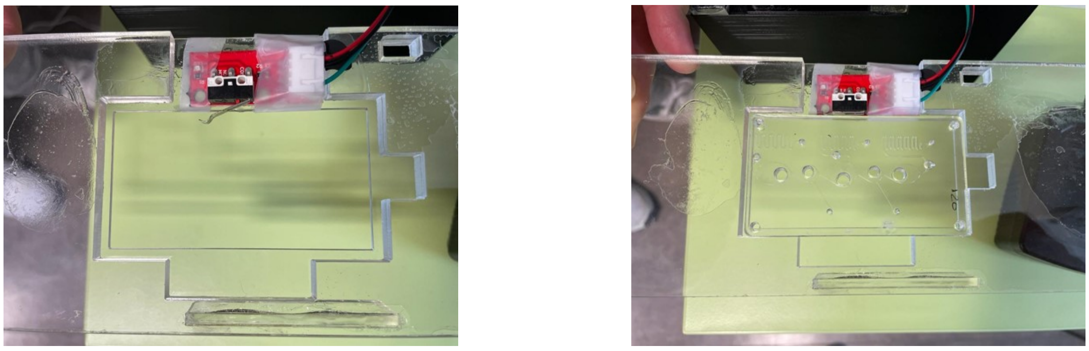


Figure 29: Gluing sensor 2 on the activator (left). Sensor 2 is pressed when a chip is added (right).

As a last step, we connected the button to the Aduino board and placed it in its dedicated compartment in the base.

However, there were still a few things to adjust. First, when putting the chip in its place, we realized that the the position between the starting well and the sensor should be decreased by a small distance (few millimeters). To not reprint the base, we just thought of printing a small piece that would have a thickness equal to this distance and that we can glue on the part of the magnet holder that is supposed to touch the sensor so that the magnet holder touches the sensor at the right position (position at which the magnet is facing the first well). Second, when putting the cover on the base, we found that the distance between the two should be increased by a few millimeters. Thanks to our "pins design", we just needed to print small cubes having dimensions equal to this additional height we want, and we glued them at the top of the base and then added the cover, which was now at the position we want from the base.

Our prototype (figure 30) was then ready for testing. Using our Arduino code, we rechecked and adjusted all the connections on the Arduino board since some of the cables were wrongly switched during the assembling process. Then, we tested the code that was perfectly working on all the components (button, chip sensor and motor) except for the sensor at the starting position. Since we were using a capacitive touch sensor to code the sensor at the starting position, we replaced the mechanical sensor we have glued by a capacitive sensor and it indeed worked.

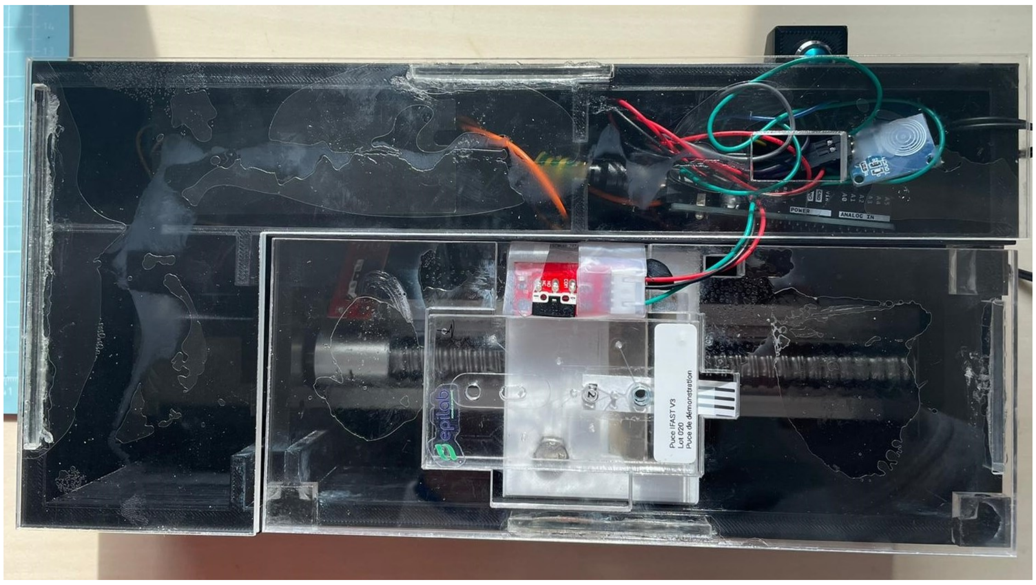


Figure 30: Assembled prototype.

So as can be seen, almost everything was done in the assembling and there was only the problem of sensor 1 that needed to be fixed to finalize and have a functioning prototype. There are two possibilities to solve this issue. A possibility would be to reprint the base with the right dimensions for the capacitive touch sensor (on which our code would work directly). For this we could just adjust the dimensions of the part dedicated for sensor 1 in the base and reprint the base in around 16 hours as previously done. Another option could be to spend some time trying to find which part of the code should be changed to take into account this change of sensor.Both are simple solutions that could be done in one or two sessions but sadly we could not continue the project which was ended abruptly.

REFERENCES

[1] E. Barbier, T. Fouchet, A. Hartmann, E. Cambau, M. Mougari, C. Dubois, M. Lubetzki and M. Rochelet, Rapid electrochemical detection of Mycobacterium tuberculosis in sputum by measuring Ag85 activity with disposable carbon sensors. *Talanta* (2022) 123927

[2] D.S. Boyle, Unitaid Diagnostics Technology Landscape, fifth ed., 2017, https://doi.org/10.13140/RG.2.2.32003.81441. May 2017.

[3] M. Cruciani, C. Scarparo, M. Malena, O. Bosco, G. Serpelloni, C. Mengoli, Metaanalysis of BACTEC MGIT 960 and BACTEC 460 TB, with or without solid media, for detection of mycobacteria, *J. Clin. Microbiol.* 42 (2004) 2321–2325

[4] H.G. Wiker, M. Harboe, The antigen 85 complex: a major secretion product of Mycobacterium tuberculosis, *Microbiol. Rev.* (1992), https://doi.org/10.1128/ mr.56.4.648-661.1992.

[5] World Health Organization, in: Global Tuberculosis Report 2020, World Health Organization, Geneva, 2020.

[6] Shah, J. Snider, B. Clarke, T. Kozutsky, S. Lacki, Maciej Hosseini, Ali. (2019). Large-scale 3D printers for additive manufacturing: design considerations and challenges. The International Journal of Advanced Manufacturing Technology. 104. 1-15. 10.1007/s00170-019-04074-6.

[7] Gassner, Anne-Laure Abonnenc, Mélanie Chen, Hong-Xu Morandini, Jacques Josserand, Jacques Rossier, Joel Busnel, Jean-Marc Girault, Hubert. (2009). Magnetic forces produced by rectangular permanent magnets in static microsystems. Lab on a chip. 9. 2356-63. 10.1039/b901865d.

APPENDIX

Main Arduino code to control the activator

```
int touchPinc = 4;
int valc = 0;
int relayPinc = 8;

int touchPinm = 9;
int valm = 0;
int relayPinm = 6;

double stepsPerRevolution = 3200; // 3200 : nombre de steps pour un tour de moteur
//les distances sont en millimètre

double d_w1 = 4; //longueur du premier puits
double d_w2 = 3 ; //longueur du deuxième puits
double d_w3 = 4 ;//longueur du troisième puits
double d_w4 = 3;//longueur du quatrième puits
double d_w5 = 3.6 ;//longueur du cinquième puits

double d0_1 = 23.01; //distance entre le bord du chip et l'entrée du premier puits
double d1_2 = 3.07 ; //distance entre la sortie du premier puits et l'entrée du deuxième
double d2_3 = 3.26 ;//distance entre la sortie du deuxième puits et l'entrée du troisième
double d3_4 = 4.81 ;//distance entre la sortie du troisième puits et l'entrée du quatrième
double d4_5 = 3.26 ;//distance entre la sortie du quatrième puits et l'entrée du cinquième

double d_aller_retour_1 = d_w1 ; //distance parcourue pour faire un aller ou un retour dans le premier puits
double d_aller_retour_2 = d_w2 ; //distance parcourue pour faire un aller ou un retour dans le deuxième puits
double d_aller_retour_3 = d_w3 ; //distance parcourue pour faire un aller ou un retour dans le troisième puits
double d_aller_retour_4 = d_w4 ; //distance parcourue pour faire un aller ou un retour dans le quatrième puits
double d_aller_retour_5 = d_w5 ; //distance parcourue pour faire un aller ou un retour dans le cinquième puits

double d_retour_pt_depart = d_w1+d_w2+d_w3+d_w4+d_w5+d0_1+d1_2+d2_3+d3_4+d4_5;//distance parcourue pour
revenir au points de départ
double d_tour_moteur = 5 ;//distance parcourue lorsque le moteur fait un seul tour complet

int nb_allers_retours_1 = 20 ; //nombre d'allers-retours dans le premier puits
int nb_allers_retours_2 = 20 ; //nombre d'allers-retours dans le deuxième puits
int nb_allers_retours_3 = 60 ; //nombre d'allers-retours dans le troisième puits
int nb_allers_retours_4 = 20 ; //nombre d'allers-retours dans le quatrième puits
int nb_allers_retours_5 = 20 ; //nombre d'allers-retours dans le cinquième puits

double t = 1500; //durée d'activation ou de désactivation
//cette durée définit la vitesse du moteur qu'on ne peut régler qu'en changeant ce paramètre

int currStatem = 1; //Etat actuel du capteur mecanique
int prevStatem = 1; //Etat Précédent du capteur mecanique
int currStatec = 1; //Etat actuel du capteur capacitive
int prevStatec = 1; //Etat Précédent du capteur capacitive
int currStateb = 1; //Etat actuel du bouton
int prevStateb = 1; //Etat Précédent du bouton
bool running = false; //Etat du programme
```

```
//___________________________________________________________________

void setup() {
  pinMode(stepPin, OUTPUT);
  pinMode(dirPin, OUTPUT);
  pinMode(button,INPUT_PULLUP);

  Serial.begin(9600);
  pinMode(touchPinm, INPUT_PULLUP);
  pinMode(relayPinm, OUTPUT);
  pinMode(touchPinc, INPUT);
  pinMode(relayPinc, OUTPUT);
  }

//___________________________________________________________________


void loop(){
  CheckButton();
  steps();
}

void steps(){
  if (running) avancer (8,t);
  if (running) delay(3000);
  if (running) avancer (d_w1/2,t);
  if (running) allers_retours(d_aller_retour_1,t/50 ,nb_allers_retours_1);
  if (running) delay(3000);
  if (running) avancer (d1_2+d_w2,t);
  if (running) allers_retours(d_aller_retour_2,t/50 ,nb_allers_retours_2);
  if (running) delay(3000);
  if (running) avancer (d2_3+d_w3,t);
  if (running) allers_retours(d_aller_retour_3,t/50 ,nb_allers_retours_3);
  if (running) delay(3000);
  if (running) avancer (d3_4+d_w4,t);
  if (running) allers_retours(d_aller_retour_4,t/50 ,nb_allers_retours_4);
  if (running) delay(3000);
  if (running) avancer(d4_5+d_w5,t);
  if (running) allers_retours(d_aller_retour_5,t/50 ,nb_allers_retours_5);
  if (running) CheckSensorm();
  if (running) delay(9000);
  if (running) reculer(500,t/20);
}


//___________________________________________________________________
```

```
void CheckSensorm() {
 int Statem = digitalRead(touchPinm);
       if (Statem == LOW){
        digitalWrite(relayPinm, HIGH);

        while(digitalRead(touchPinm)==LOW){
         delay(1000);
         }
        }
        digitalWrite(relayPinm, LOW);
}

//______________________________________________________________________
void CheckSensormopp() {
 int Statem = digitalRead(touchPinm);
       if (Statem == HIGH){
        digitalWrite(relayPinm, LOW);

        while(digitalRead(touchPinm)==HIGH){
         delay(1000);
         }
        }
        digitalWrite(relayPinm, HIGH);
}

//______________________________________________________________________
void CheckSensorc() {
 digitalWrite(relayPinc, HIGH);
 currStatec = digitalRead(touchPinc);
 if (currStatec==1)
 {
      CheckSensormopp();
      delay(5000);
      steps();
      digitalWrite(relayPinc, LOW);
 }
 else
   {
      digitalWrite(relayPinc, HIGH);
       // Switch is now released
   }
  prevStatec = currStatec;
}

//______________________________________________________________________
void CheckButton() {
 currStateb = digitalRead(button);
  if(currStateb != prevStateb)
  {
      // A transition occurred
      if(currStateb == LOW)
      {
        // Switch is now pressed
        if(not running){
          CheckSensorm();
          running = true;
          reculer(30, t/20);
       }
        if(running){
        digitalWrite(relayPinc, HIGH);
        running = false;
        }
      }
      else
      {
         // Switch is now released
      }
      prevStateb = currStateb;
    }
}

//____________________________________________________________

//cette fonction permet d'avancer d'une distance d donnée
//choisie lors de l'appel de cette fonction
//d est la distance à parcourir
//t est à choisir: grand si on veut avancer lentement, faible sinon

void avancer (double d, double temps){
  digitalWrite(dirPin, HIGH);
  double nombre_de_pas = d * stepsPerRevolution / d_tour_moteur ;
  for (int i = 0; i < nombre_de_pas; i++) {
    digitalWrite(stepPin, LOW);
    delayMicroseconds (temps);
    digitalWrite(stepPin, HIGH);
    delayMicroseconds (temps);
    CheckButton();
    if(not running){
      return;
    }
  }
}

//____________________________________________________________

//cette fonction permet de reculer d'une distance d donnée
//choisie lors de l'appel de cette fonction
//d est la distance à parcourir
```

```
//t est à choisir: grand si on veut avancer lentement, faible sinon

void reculer (double d, double temps){
 digitalWrite(dirPin, LOW);
 double nombre_de_pas = d * stepsPerRevolution / d_tour_moteur ;
 for (int i = 0; i < nombre_de_pas; i++) {
   digitalWrite(stepPin, HIGH);
   delayMicroseconds (temps);
   digitalWrite(stepPin, LOW);
   delayMicroseconds (temps);
   CheckSensorc();
   if(not running){
     return;
   }
 }
}

//________________________________________________________________________


//cette fonction permet de faire des allers-retours d'amplitude d donnée
//choisie lors de l'appel de cette fonction
//d est la distance à parcourir
//t est à choisir: grand si on veut avancer lentement, faible sinon

void allers_retours (double d, double temps, int nb_aller_retour){

 double nombre_de_pas = d * stepsPerRevolution / d_tour_moteur ;
 for (int j = 0; j < nb_aller_retour; j++) {
   digitalWrite(dirPin, LOW);
   for (int i = 0; i < nombre_de_pas; i++) {
     digitalWrite(stepPin, HIGH);
     delayMicroseconds (temps);
     digitalWrite(stepPin, LOW);
     delayMicroseconds (temps);
     CheckButton();
     if(not running){
       return;
     }
   }

   delay(10);

   digitalWrite(dirPin, HIGH);
   for (int i = 0; i < nombre_de_pas; i++) {
     digitalWrite(stepPin, LOW);
     delayMicroseconds (temps);
     digitalWrite(stepPin, HIGH);
     delayMicroseconds (temps);
```

```
        CheckButton();
        if(not running){
          return;
        }
      }
      delay(10);
    }
  }
```

Protocol to 3D print a piece using Ultimater Cura:

- Import the STL file.
- Choose the printer.
- Verify that materials wanted are contained in the chosen printer.
- Set the printing parameters (eg. "Profiles" for layer thickness and "Infill" for solid part filling)
- Click on "Slices" to make the software to slice the model.
- Insert the USB key into the printer to start printing.

Protocol to cut a piece with Trotec engraver (using CorelDraw software):

- Turn on the machine and place the plate to be cut.
- Calibrate the laser focus by using the probe to mount the plate/grid until it comes off.
- Open CorelDraw and select the pattern to be cut.
- Set the thickness to "very thin line" and the color to red. Go to File Print and select the Trotec printer properties, ensure that "minimize to jobsize" and "inner geometries first" are selected.
- Print the pattern. Access the Trotec JobControl software and connect the machine with "Ready".
- Select the cutout and move it to the desired location. Adjust the parameters for power, speed, and frequency. Start cutting by clicking on "Start".