Flying cellulo for rehabilitation

Louis JOURET - 269882

Professor: Pierre Dillenbourg
Advisor: Victor Borja

December 2021

Contents

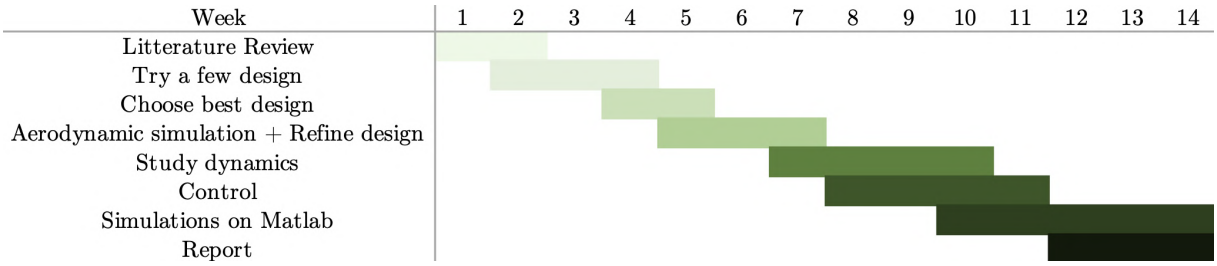
1 Introduction

This project aims to study the feasibility of haptic feedback in a 3D space. There is an increasing demand for such solutions, mainly in rehabilitation since they remove the need for a daily appointment with a therapist and enable the gamification of rehabilitation. Some solutions already exist but remain very expensive since they are based on 3-axis arms. Nowadays, a lot of progress in the field of aerial robotics has been done and it is time to consider using the principles of flying drones to take over this task.

The second goal of the project is to propose a proof of concept. Thus a first design will be drawn which will serve us to analyze the technical challenges that have to be overcome to make it a viable solution for rehabilitation. Among these challenges, we count the difficulty to draw a graspable drone that remains completely safe and the writing of a control algorithm that is fast enough to create a user-friendly experience.

This work will, first of all, see what has already been done in the field of machine-assisted rehabilitation. Second of all, we will design a first concept that fulfills all the mechanical requirements. Once the design has been done, the dynamics of the model are studied to finally write two different control algorithms. The technical work ends with tests of the first material purchased and the first test of haptic feedback. Finally, we will discuss if a flying rehabilitation drone is a feasible solution.

Below is the Gantt chart initially defined at the start of the semester:



2 Existing solutions

There already exist solutions that are designed for rehabilitation. They are mainly based on a robotic arm that creates haptic feedback to the patient's hand. One of the most famous products is MIT-Manus. The patient puts his elbow in a support which is held by a robotic arm. The patient then performs different movements which are corresponding to divers exercises shown on the screen attached to the robotic arm.

Figure 1: MIT-Manus [1], Credit: H. I. Krebs, Newman Lab, MIT

Hocoma, which is one of the world leader in advanced technologies for movement rehabilitation, has a complete range of 3 products called Armeo®, designed for the rehabilitation of the arm depending on the severity of the handicap.

(a) Armeo®Power

(b) Armeo®Spring

(c) Armeo®Senso

Armeo®Power [2] is the most powerful solution and can assist and guide the patient's hand in its movements. Armeo®Spring [3] is a lighter product designed to support the arm's weight and is thus less powerful. These products come with their monitor on which the patient follows instructions. Armeo®Senso [4]is the lightest product designed for people that can already support the weight of their own arm.

So if a patient needs a product that can create haptic feedback and support the arms weight, the only solutions available are very expensive and impossible to bring home. Thus a flying Cellulo which is capable of supporting the patient's arm and giving haptic feedback would add value to the market if it can achieve these goals.

3 Design

First of all, the power of the Cellulo has to be determined. Since the product needs to assist the patient's hand and that it should also be able to create resistance, the weight of an adult's hand is taken as a reference load. Thus the drone should be able to lift the hand without the patient's help. A simple weighting of 3 different hands in different positions, gives a maximum weight of 1.5 kg.

Secondly, the fingers have to be protected from the blades. The hand must either be far away from any moving part or the latter have to be covered by a protective grid.

The project is also considering controlling the final product with common video game consoles such as Wii, PlayStation or XBox. Therefore it is necessary to work with a communication protocol that is supported by these platforms. Bluetooth happens to be compatible with the three of them.

Finally, the patient must be able to use the product over a whole workout session, which is not possible with the current battery technology. Hence, the drone has to be supplied with power from the ground. This is not a limiting factor for the user-friendliness of the product since it does not need to fly far away from the ground station.

Thus, the basic requirements that have to be respected whatever the design of the product, are the following:

1. thrust of up to 1.5 kg + own weight
2. safety from the blades
3. Bluetooth as communication protocol
4. no battery on-board

3.1 Feasible architectures

There are a lot of very different architectures possible to make a drone fly. However, each comes with its perks and drawbacks. Let's take a closer look at the ones that would be capable of achieving the desired goal:

3.1.1 Quadcopter

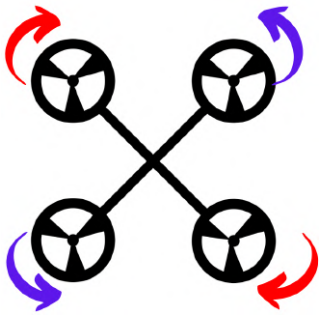


Figure 3: motor layout of a quadcopter

The quadcopter uses 4 motors which are equidistant to the drone's center. It is the most seen and popular architecture due to its stability and controllability. Due to its 4 motors, it is the most powerful of the following solutions. However, this comes with high power consumption. But since we are supplied from the ground and are thus not limited by power consumption. The biggest disadvantage of the quadcopter is its footprint. Due to the four propeller, it is not possible to create a compact solution.

3.1.2 Tricopter

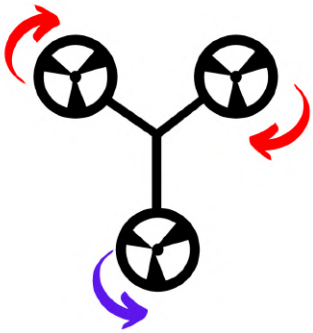


Figure 4: motor layout of a tricopter

The tricopter can counter the quadcopter's footprint problem. Only using three propellers makes it compacter and lighter. However, this has its toll on the thrust and the torque applied to the center of mass. Hence, it is less controllable than the quadcopter. Furthermore, the tricopter suffers from an asymmetric yaw momentum. Since we don't have an even number of motors, the motor that spins counterclockwise has to either have a bigger propeller or spin faster than the two clockwise propellers.

3.1.3 Bicopter

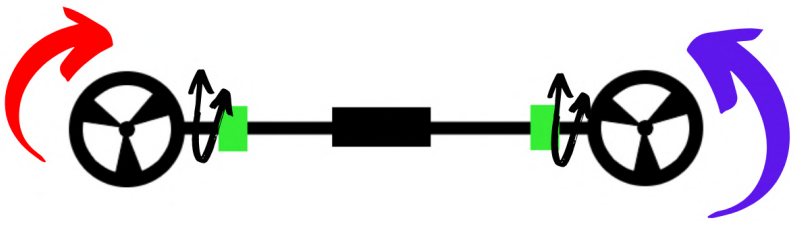


Figure 5: motor and servo (green) layout of a bicopter

The bicopter configuration is an even more extreme architecture in which only two motors are used. This reduces the footprint even more and thus allows the most compact solution. However since two motors in a plane lack a degree of freedom to be able to be used in our project, two servos have to be added to the structure to incline the motors. In Figure 5 the servos are represented in green.
In addition, the bicopter is heavily limited by its thrust since only two motors have to deliver the whole force. Since the project requires lifting a payload of 1.5 kg, this comes as a big downside.

3.1.4 Final choice

Due to the nature of the project, the drone will have to lift heavy payloads but remain compact. Unfortunately, the only solution that would be able to lift a payload of 1.5 kg without taking a hit on its controllability is the quadcopter. Thus it is the architecture that was chosen for the realization of this project. Before designing the 3D model, the propeller size has to be chosen since it will impact the design significantly. After consulting various datasheets, we settle for 5 inches. It is a good compromise between size and thrust. Generally, the bigger the size, the more thrust we get for a given power. Furthermore, we can already decide which onboard computer we will use. The Raspberry Pi has a Bluetooth module and great documentation. Unfortunately, it is quite heavy. Thus we purchased the all-new Raspberry Pi Zero 2W which has all the communication features we need but weighting 5 times less.

3.2 Design iterations for the 3D model

Once the system's architecture has been chosen, the first 3D model can be created. The models have to respect the safety of the patient and let enough airflow through the chassis. Limiting the air obstruction is primordial to guarantee sufficient thrust and stability. Thus the first 3D model that satisfies all the design requirements will go through a fluid dynamics simulation to test its aerodynamic efficiency. If the simulation shows some flaws in the design, the latter is changed and the simulation is rerun. This iteration is done until the simulations deliver satisfactory results. This design is then taken on a real testbench, where the thrust is measured, with and without the chassis, to check if the chassis is not obstructing the airflow too much.

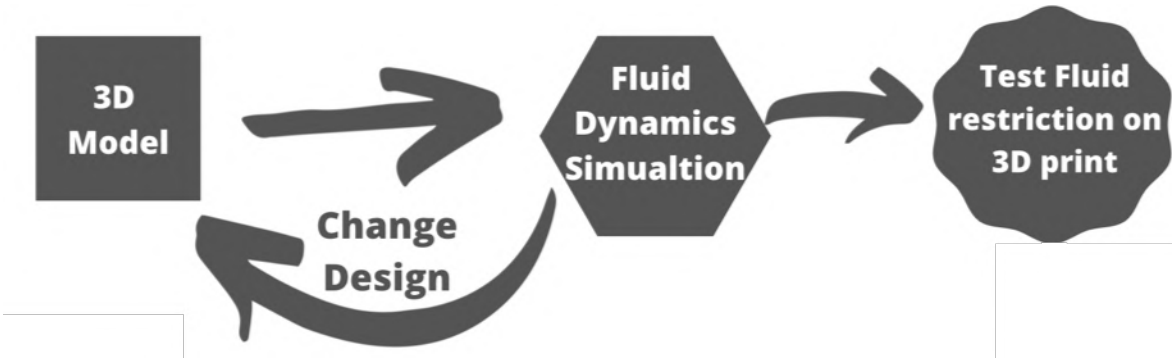


Figure 6: design plan for the chassis

3.3 Fluid Simulations

The software used to perform the simulations is Ansys Discovery Live. This software is very simple to use and is sufficient in our case. The 3D models that are tested are simplified versions of the final model that do not have protection grids/nets and do not have motor mounts. This saves valuable time and is not necessary to compare different designs since the air resistance of the grid/net will remain quite constant through all the designs.

3.3.1 First iteration

The first design that satisfied all the physical requirements and was tested for air flow is the following:

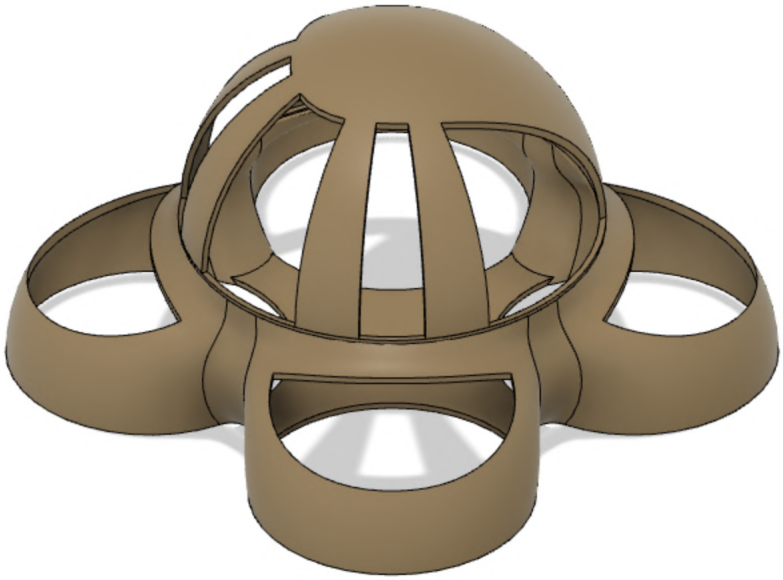
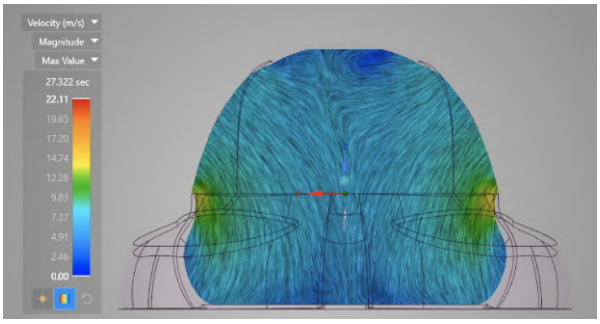
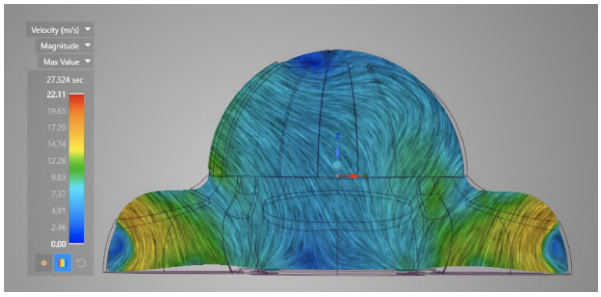


Figure 7: first design iteration

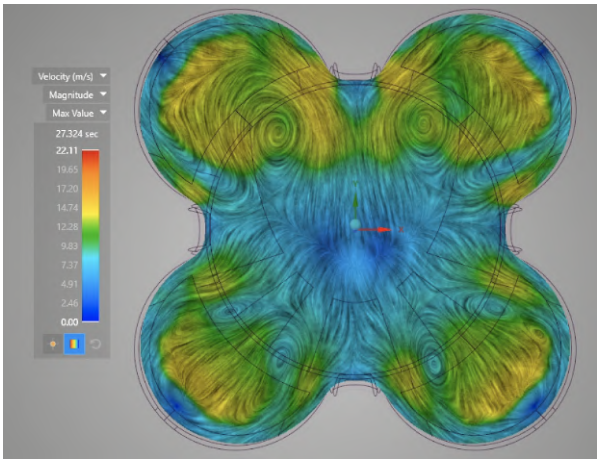
This design delivered the following air velocity field:



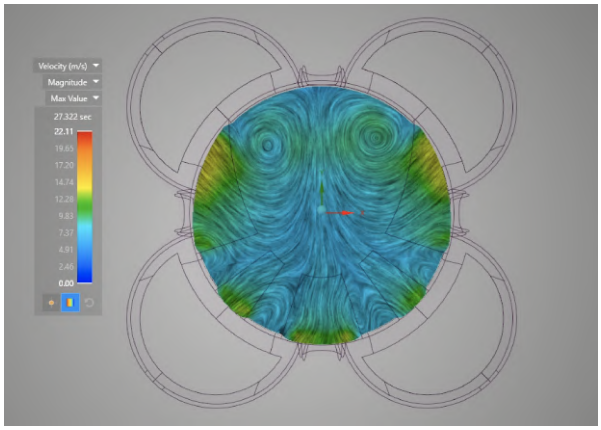
(a) side view of the velocity field



(b) side view of the velocity field



(c) downward view of the velocity field



(d) downward view of the velocity field

As can be seen in (a) the airstream has to corner around the upper opening and thus creates a faster stream closer to the rotation point. This could be resolved by reducing the angle of air intake. (b) shows that the airflow coming from the upper part of the chassis is small compared to the flow coming directly from the fan pockets. To balance the two contributions, we should increase the size of the upper pockets and bring them closer to the fans. (c) and (d) are showing a lot of turbulence inside the core. To avoid those, the lower part of the hand support should get slimmer.

3.3.2 Last iteration

After 4 different iteration and simulations of different design parameters we settled for the following:

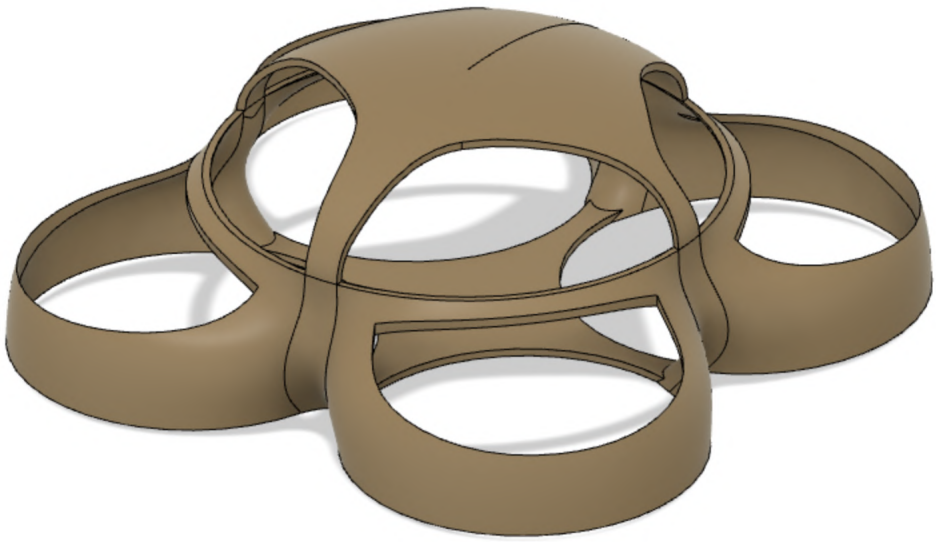
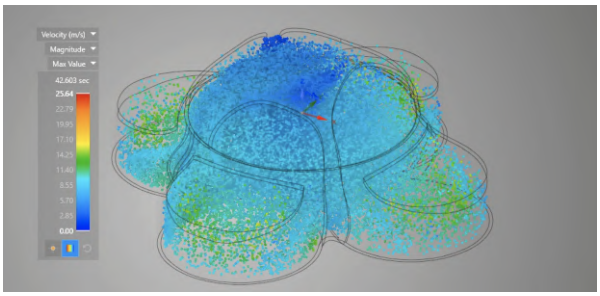
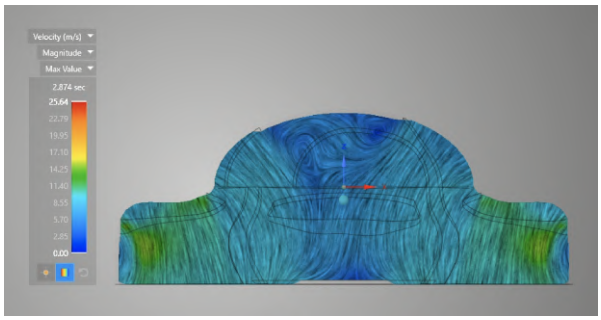


Figure 9: last design iteration

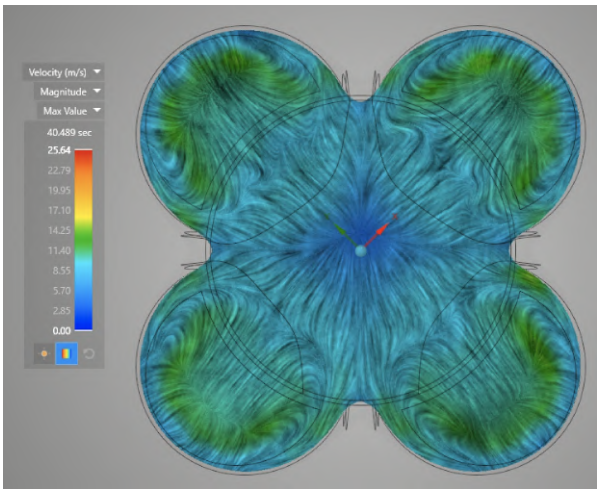
The upper pockets got widened and lowered to increase the airflow coming from the upper part of the chassis. Since the upper pockets are now directed downwards, the airstream does no longer need to corner. Furthermore, the fan pockets got bigger to increase the airflow coming directly from above the fan. These two major changes allow us to get a much smoother/even velocity profile and less turbulence compared to the first iteration.



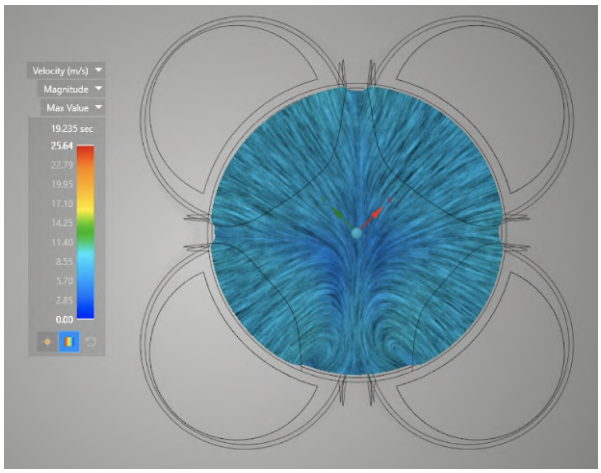
(a) isometric view of the velocity field



(b) side view of the velocity field



(c) downward view of the velocity field

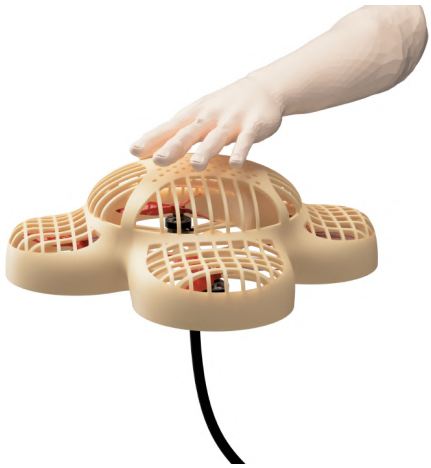


(d) downward view of the velocity field

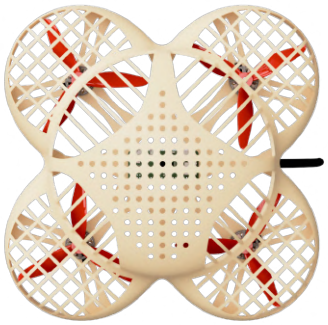
3.4 Finalization of the 3D model

Now that the main shapes of the design are validated, the final part of the 3D design consists of adding safety grids/nets and the motor mounts. The choice between grids or nets can at this stage not be determined since the drone is not flying yet and the impact on user-friendliness can not be determined. This will have to be tested once the drone is used and both solutions can be tested. Thus, grids will be used for the finalization of the design but can always be replaced later by nets. Concerning the propellers, the only parameter that can be changed is the number of blades and the pitch. The theory says that the more blades a propeller has, the more thrust it can produce but the less controllable it is. The optimal number of blades will have to be determined through empirical testing, once the drone is flying. For the moment we settled for 3 blades. The final design including safety grids, motor mounts, propellers, a Raspberry Pi and a whole for the supply cable looks like this:

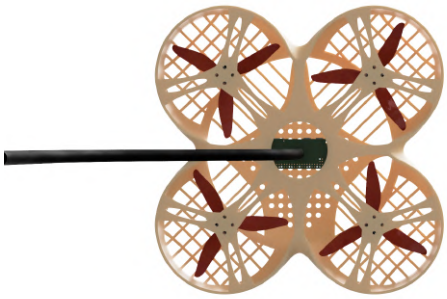
(a) isometric view from behind



(b) isometric view from the front



(c) view from above



(d) view from beneath

The volume of the final chassis is $2.97 \cdot 10^5 mm^2$. If the chassis is 3D printed in ABS, which is mechanically the most robust material classical FDM printer can print and which has the lowest density, the print would weigh 326g. Finally, we can also extract the moment of inertia since it will be necessary for the dynamics of the drone. We use the following axis:

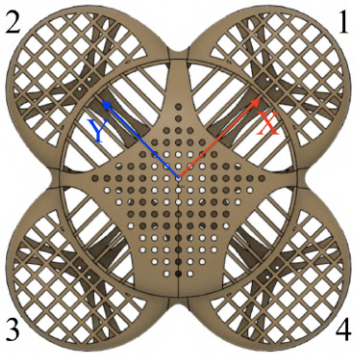


Figure 12: choice of X and Y axis and numbered motors

Using these axis, Fusion 360 gives us the following matrix:

$$\mathbf{I} = \begin{bmatrix} I_{11} = 2.66 & I_{12} = -3.53 \cdot 10^{-2} & I_{13} = 2.51 \cdot 10^{-2} \\ I_{21} = -3.53 \cdot 10^{-2} & I_{22} = 2.66 & I_{23} = 2.43 \cdot 10^{-2} \\ I_{31} = 2.51 \cdot 10^{-2} & I_{32} = 2.43 \cdot 10^{-2} & I_{33} = 4.64 \end{bmatrix} \cdot 10^{-3}[kgm^2] \quad (1)$$

To greatly simplify the dynamics equations we can approximate the small moment of inertia by zero. This is the matrix that will be used:

$$\mathbf{I} = \begin{bmatrix} 2.66 & 0 & 0 \\ 0 & 2.66 & 0 \\ 0 & 0 & 4.64 \end{bmatrix} \cdot 10^{-3}[kgm^2] \quad (2)$$

3.5 Cost

After having designed the chassis and knowing its volume we can finally do a cost estimation for the first prototype. This cost estimation does not take into account the power supply on the ground since the price can fluctuate a lot depending on quality.

Motor: 4x *F40PRO IV KV1950* (24V) from *T-motor* **4 x 27 CHF**
Speed controller: *F55A PRO II F3 6S 4IN1* from *T-motor* **104 CHF**
Propellers : 4x *T5150* from *T-motor* **4 x 0.7 CHF**
Rasberry Pi **50 CHF**
326g of ABS **18CHF/kg x 0.326kg**
2x 2.5m cable of $6mm^2$ **2 x 2.5m x 1.8 CHF/m**

Total **280 CHF**

Adding to this a power supply, which costs at least 100 CHF for correct quality, brings the whole project to a pretty high price tag. The majority of the price comes from the motors and their controller which are very powerful. If after the first tests, the level of power initially estimated seems to be unnecessary, a smaller controller and motors can be mounted instead which will drastically drop the price.

4 Dynamics

In this section, we will derive the dynamics of the drone. [5] helped at this task. We chose a 12-state description for the cellulo:

$$\mathbf{x} = [\alpha, \beta, \gamma, \dot{\alpha}, \dot{\beta}, \dot{\gamma}, x, y, z, \dot{x}, \dot{y}, \dot{z}] \tag{3}$$

We will write $p = [x, y, z]$ as the position and $\theta = [\alpha, \beta, \gamma]$ the roll, pitch and yaw. The input are the four motor thrusts $\mathbf{u} = [u_1, u_2, u_3, u_4]$.

Each motor produces a force and a torque proportional to its thurst:

$$F_i = k_F u_i$$
$$M_i = k_M u_i$$

The total force and torque can be expressed as follows:

$$\begin{bmatrix} F \\ M_\alpha \\ M_\beta \\ M_\gamma \end{bmatrix} = \begin{bmatrix} k_F & k_F & k_F & k_F \\ 0 & k_F a & 0 & -k_F a \\ -k_F a & 0 & k_F a & 0 \\ k_M & -k_M & k_M & -k_M \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \tag{4}$$

The angular dynamics of the system are given by:

$$\ddot{\theta} = I^{-1}(-\dot{\theta} \times I\dot{\theta} + \begin{bmatrix} M_\alpha \\ M_\beta \\ M_\gamma \end{bmatrix} + \begin{bmatrix} M_{Xext} \\ M_{Yext} \\ M_{Zext} \end{bmatrix}) \tag{5}$$

$$\ddot{p} = \frac{1}{m}\Sigma\vec{F} = -\vec{g} + \frac{F}{m}\vec{z_c} + \vec{F_{ext}} = \begin{bmatrix} \frac{F}{m}sin(\beta) + F_{Xext} \\ -\frac{F}{m}cos(\beta)sin(\alpha) + F_{Yext} \\ -g + \frac{F}{m}cos(\alpha)cos(\beta) + F_{Zext} \end{bmatrix} \tag{6}$$

where $\vec{F_{ext}}$ and $\vec{M_{ext}}$ are the external force and torque applied by the patient.
Using equation (4), (3), (5) and (6) we can now express the state-space of the system:

$$\dot{\mathbf{x}} = f(\mathbf{x},\mathbf{u}) = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ I_1^{-1}(x_6x_5(I_2 - I_3) + k_F a(u_2 - u_4) + M_{Xext} \\ I_2^{-1}(x_4x_6(I_3 - I_1) + k_F a(u_3 - u_1) + M_{Yext} \\ I_3^{-1}(x_4x_5(I_1 - I_2) + k_M(u_1 - u_2 + u_3 - u_4) + M_{Zext} \\ x_{10} \\ x_{11} \\ x_{12} \\ \frac{k_F}{m}(u_1 + u_2 + u_3 + u_4)sin(x_2) + \frac{F_{Xext}}{m} \\ -\frac{k_F}{m}(u_1 + u_2 + u_3 + u_4)cos(x_2)sin(x_1) + \frac{F_{Yext}}{m} \\ -g + \frac{k_F}{m}(u_1 + u_2 + u_3 + u_4)cos(x_1)cos(x_2) + \frac{F_{Zext}}{m} \end{bmatrix} \tag{7}$$

5 Control

Now that all the dynamical equations have been derived, control algorithms can be written and compared. We will analyze two algorithms which differ a lot in the sense that one will be able to predict the patient's movements and thus better prepare for it and another one will just react to the current state but will therefore be a lot less resource-intensive and can thus perform at higher frequencies.

To lighten the control computations we need to linearize (7). The linearization around a nominal point can be expressed using the Taylor expansion:

$$\begin{aligned}
\dot{x} &= f(x,u) \\
&= f_i(\bar{x},\bar{u}) + \frac{\delta f_i(\bar{x},\bar{u})}{\delta x_1}\tilde{x_1} + ... + \frac{\delta f_i(\bar{x},\bar{u})}{\delta x_n}\tilde{x_n} + \frac{\delta f_i(\bar{x},\bar{u})}{\delta u_1}\tilde{u_1} + ... + \frac{\delta f_i(\bar{x},\bar{u})}{\delta u_n}\tilde{u_n} + p(x,u) \\
&\approx \frac{\partial f}{\partial x}x + \frac{\partial f}{\partial u}u \\
&= Ax + Bu
\end{aligned}$$

Now we can discretise the state-space:

$$\begin{aligned}
& x(k) = Ax(k) + Bu(k) \\
\iff & \frac{x(k+1) - x(k)}{h} = Ax(k) + Bu(k) \\
\iff & x(k+1) = (hA + I)x(k) + hBu(k) \\
\iff & x(k+1) = \Phi x(k) + \Gamma u(k)
\end{aligned}$$

5.1 LQR Control

The first controller is an LQR controller which stands for Linear-Quadratic regulator. This controller is minimizing a cost function composed of a first quadratic term representing the deviation from the reference state and a second quadratic term representing the magnitude of the input:

$$J = \sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k$$

where the matrices Q and R have to be tuned by the designer.

The classical LQR optimal input is given by:

$$u_k = -Kx_k \qquad \text{where } K = (R + \Gamma^T P \Gamma)^{-1}(\Gamma^T P \Phi + N^T)$$

and P is the solution of the Riccati equation. Fortunately, MATLAB already has an integrated function that computes K. We could also have used a Proportional-Derivative controller, but unfortunately they are quite difficult to tune for a system with as many states. With the LQR, we directly get the optimal K with respect to Q and R. Since R tunes the cost of input, the higher we set R the more economic the drone will fly. However,

we are supplied from the ground and thus do not need to be economic. Hence to get the best reactiveness we set R to zero. Now that R is set to zero, the value of Q doesn't matter for the solution of the cost function. In the algorithm, Q was set to the identity matrix.

In our case, we also need an integrator because otherwise the drone would not come back to its initial position after a hand is put on it. The integrator accumulates all the errors overtime to produce a supplementary input of weight α. Furthermore, we add the nominal input which corresponds to a quarter of the drone's weight to the input to avoid an initial drop in altitude. Thus our input is given by:

$$u_k = -Kx_k - \alpha K \sum_{i=0}^{k} (x_k - x_{ref}) + u_{nominal}$$

where $\alpha = 0.05$ rendered satisfactory results.
Using this controller and the dynamical equation, we can now simulate the quadcopter's behavior when subject to various external forces and torques.

To compare the different controllers we will apply the same external forces and torques, which are:

$$F_{ext} = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \\ -14 \end{bmatrix} N \qquad M_{ext} = \begin{bmatrix} M_\alpha \\ M_\beta \\ M_\gamma \end{bmatrix} = \begin{bmatrix} 0.05 \\ -0.05 \\ 0 \end{bmatrix} Nm$$

They should represent the forces applied by a hand of approximately 1.5kg. We add 5N force to the x and y-axis since it is impossible for the patient to apply a perfectly vertical force. Furthermore, the force will be applied as a ramp because the patient will not drop his whole hand instantly. This is the force profile that will be applied:

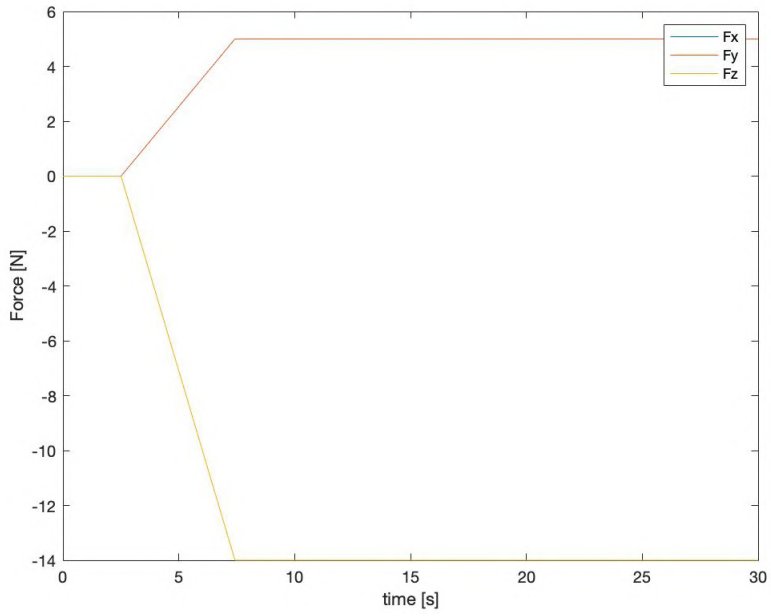
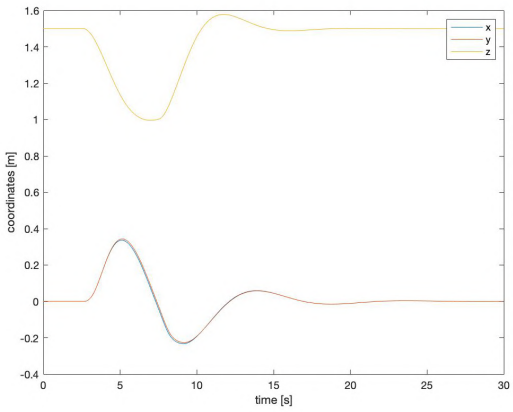
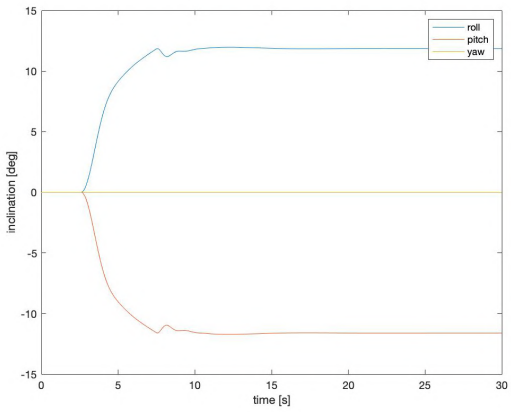


Figure 13: force profile through time

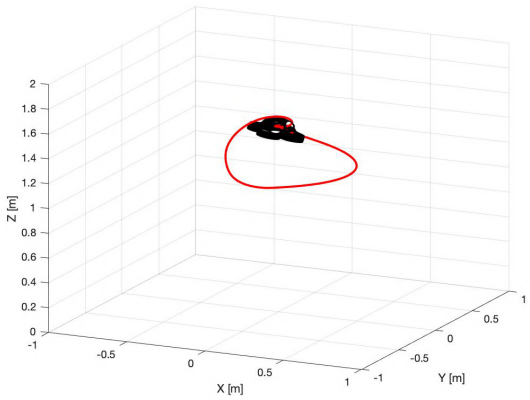
We apply the same ramp up for the torques.
If these external perturbations are applied to the quadcopter we get the following results:



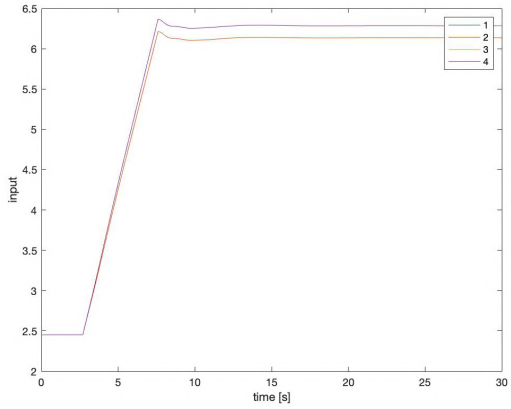
(a) coordinates of the drone through time



(b) inclination of the drone through time



(c) trajectory of the drone



(d) amplitude of input for each motor

As can be seen in the figures above, the drone manages to stabilize after the disturbances have been applied. However, the drone suffers from a few problematic behaviors. Firstly we can observe a huge drop in altitude (0.5m drop) when the hand is placed on it. Secondly, due to the small forward force of 5N we applied to the drone, the drone moves too much forward (0.4m). This misbehavior is due to the fact that the controller does not know when the patient will put his hand on it and thus cannot predict to increase the thrust beforehand. This could be resolved by a Model Predictive Controller.

5.2 Model Predictive Control

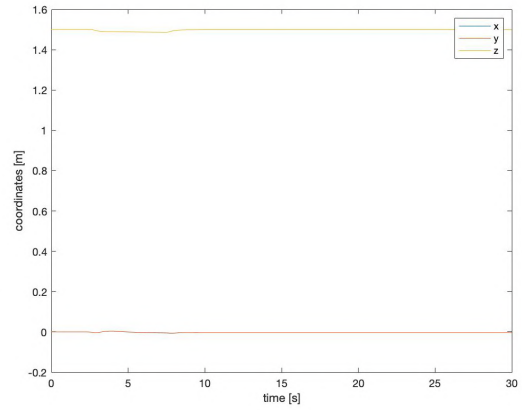
MPC is an optimal control computed over a finite horizon. The advantage of MPC is first of all that we can add as many constraints to our problem as we want. Furthermore, it will be able to foresee the wanted trajectory and thus better behave. However, the drawback is that this controller needs substantially more computational power.

The framework used to write the algorithm on MATLAB is CasADi. CasADi is an open-source tool for nonlinear optimization and algorithmic differentiation. [6] has been very helpful to write clear code with CasADi.

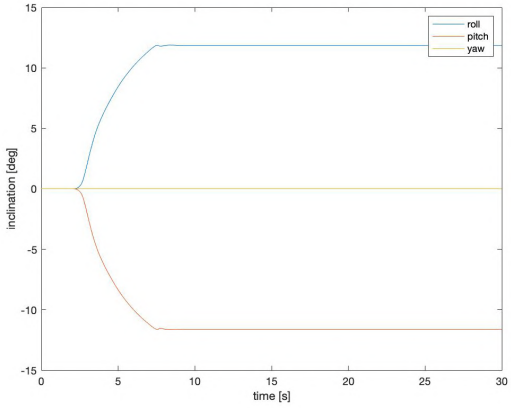
The MPC optimization problem can be expressed as follows:

$$\begin{aligned} \min_{\vec{u}} \quad & \sum_{k=0}^{N-1} \Delta\vec{x}_k^T Q \Delta\vec{x}_k + \vec{u}_k^T R \vec{u}_k \\ \text{s.t.} \quad & \Delta\vec{x}_0 = \vec{x}_0 - \vec{x}_{ref} \\ & \Delta\vec{x}_{k+1} = \Phi\Delta\vec{x}_k + \Gamma\vec{u}_k \end{aligned} \tag{8}$$

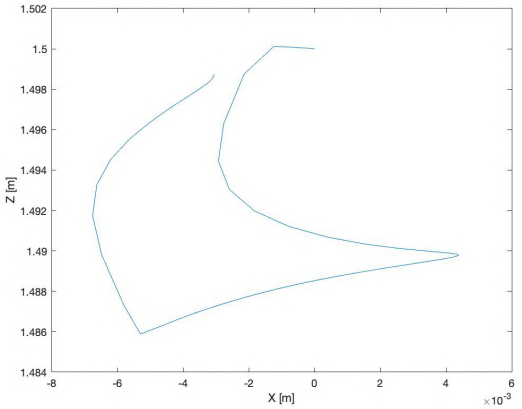
After all the inputs have been optimized for the next N steps, only u_0 is fed to the system. In our case, to take advantage of the predictive capacity of MPC, we will feed the force profile into the state-space so that the controller can foresee and counteract the patient's touch. Simulating the drone with the same external disturbances as seen in the LQR controller we get the following results:



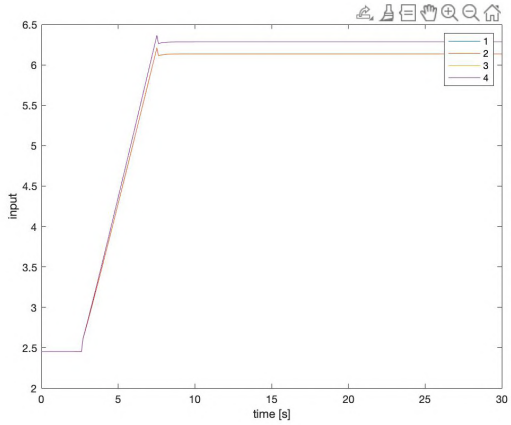
(a) coordinates of the drone through time



(b) inclination of the drone through time



(c) trajectory of the drone



(d) amplitude of input for each motor

As can be seen in the plots above, the controller perfectly counteracts the forces and torques. At first glance, it looks like the computed inputs look perfectly identical to the LQR's. However, if we plot both on the same graph and zoom in on the moment the hand touches the drone, we realize that the MPC controller is doing a jump in intensity before the ramp. This shows that it could predict the hand's weight and increases the throttle before it is too late.

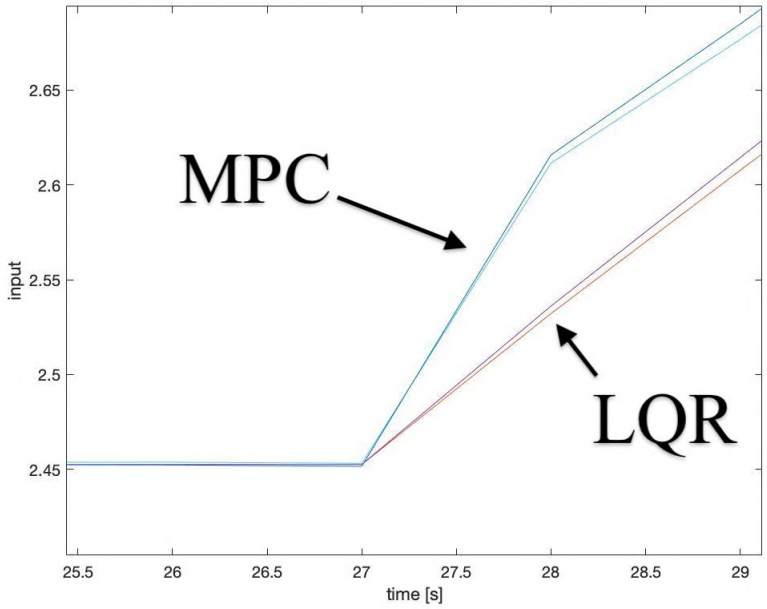


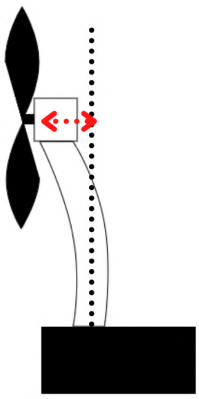
Figure 16: difference in input intensity between MPC and LQR

Unfortunately, MPC is a very resource-intensive controller and thus needs a lot of computational power. If this controller,as it stands, is able to run on a Raspberry Pi Zero 2W is currently unknown but unlikely. However, there are some parameters that can be tuned like N the number of iterations per optimization and the time horizon which determine how far into the future the controller is seeing. A smaller time horizon would make the controller much lighter in computation but still a heavy task for a Raspberry Pi Zero. Furthermore, future work will have to be done on the prediction of the moment of touch. This can be done using proximity sensors or cameras for example. The complexity of this additional work is another drawback for MPC. However, the performance difference between LQR and MPC are huge and thus continuing to inspect this solution seems important.

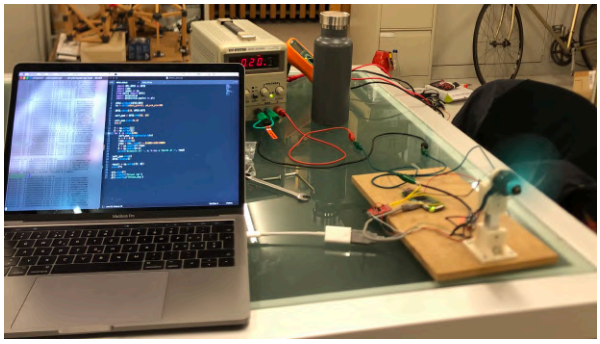
6 Testing

6.1 Relationship between Thrust and Duty Cycle

Now that the controllers have been defined, it is time to test the motors and find the relationship between the input and the actual thrust. Since the motors that are used for such applications are brushless DC motors, the command will be given in duty cycles. To measure the thrust of a motor we fix the motor on a load cell, which emits a voltage proportional to its bending.



(a) bending of the load cell through thrust



(b) picture of the setup

The components used for the build of this testbench are:

Motor: *F40PRO IV KV1950* from *T-motors* : **27 CHF**
Controller: *F45A* from *T-motors* : **29 CHF**
Load cell: *SEN-13329-TAL220* from *SparkFun Electronics* **10 CHF**
Load cell amplifier: *SEN-13879-HX711* from *SparkFun Electronics* **12 CHF**

Thus, we can increase the duty cycle, read off the power drawn from the power supply, measure the thrust and represent it in the following graph:

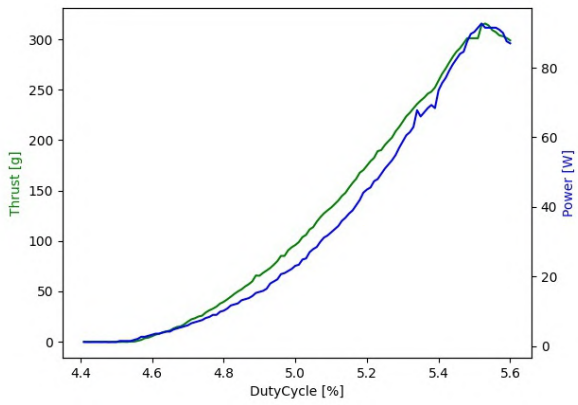


Figure 18: Thrust and power in function of duty cycle

Unfortunately, the power supply of the CHILI Lab is limited to 90W and thus we could not test the complete range of power. Nevertheless, this graph gives valuable information about the relationship between the duty cycle and the thrust. After plotting some functions on top of this graph we found that a good function that approximates the relationship is the following:

$$Thrust = 255 \cdot (DC - 4.4)^2$$
$$\iff DC = \sqrt{\frac{Thrust}{255}} + 4.4$$

This relationship is valid for a thrust under 300g and will have to be validated for a higher thrust once a bigger power source is found. This function enables us to find the proper DC command once the control algorithm has computed the thrust needed to stabilize the quadcopter.

6.2 Inclination control

In the last week of the project, we decided to build a small testbench to test the controller on the inclination. Namely, to fix the position of the drone in space and to let it rotate on a ball joint. Unfortunately, we had some issues with the delivery of the parts and the whole test fell through. If the project is taken over, this would be an interesting point to start with since it greatly simplifies the testing of the controller by only focusing on the inclination. Thereby, the controller would be able to perfectly stabilize before starting to control the space coordinates.

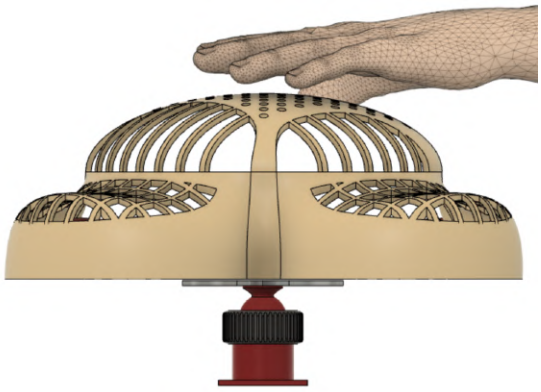


Figure 19: Cellulo attached to a ball joint

7 Conclusion

In this project we did the initial research to analyze the feasibility of a flying drone as a rehabilitation solution. However, a lot of work still has to be done to confirm if the product is really commercially viable since user-friendliness is impossible to measure at this point of development.

The biggest advantage of our solution compared to a 3-axis arm is price. Furthermore, it is a much more compact product that can be brought home without a problem. Nevertheless, this comes with much lower force feedback. A second drawback is a high sound and air blow that will come from the drone compared to the arm. The latter will have to be accessed once the drone is flying.

Regarding the 3D design, it will have to be checked for safety once the drone is up and flying. Even if it is an extremely important requirement, it seems to be the most accessible. If some flaws in the safety are detected by future developers after putting the drone in the air, the rectification should not be that challenging. As for the manufacturing of the chassis, it will have to be 3D printed for the duration of development which will come as an additional challenge considering that most common 3D printers do not support this size.

Concerning the controller, we showed that the MPC can successfully reject the external disturbances coming from the patient's hand. The next challenge would be first of all trying to run the controller on an on-board computer such as the Raspberry Pi 2W and secondly design a way to predict the moment of the patient's touch. This could for example be done using proximity sensors on the drone. In addition, the force profile of a typical touch should be studied to best predict the input needed.

As for the testing, it is unfortunate that we did not get our hands on a more powerful power source to test the total range of duty cycles. A test was scheduled and performed with the «Laboratoire des systèmes électriques distribués» but was unsuccessful due to a mechanical flaw (play in the propeller). Future developers will have to either purchase a more powerful power supply or recontact another lab.

Regarding the cost of the drone, we could reduce the price by using motors and a controller that are less powerful. The price given in this report of 280CHF was for the prototype which is quite certainly over-dimensioned to be able to validate the first proof of concept. Once the first prototype has been built, the size of the motors will be reduced depending on the performance of the prototype and the personal opinion on the user-friendliness of the developers. All in all, we estimate the final production costs at 50% of the prototype's cost.

All the initial goals of the Gantt chart were achieved, and additional challenges were introduced, however, not all of them were fulfilled due to time constraints. Future developers can start with the building of a testbench, demonstrating the working of the controller on the rotation of the Cellulo. All the files and codes can be found in the project's GitHub repository: `https://github.com/VBorjaDD/FlyingCellulo`
Special thanks to Victor Borja who supervised this project.

References

[1] M. Costandi, "Rehabilitation: Machine recovery," Nature.

[2] "Armeo®power." https://www.hocoma.com/solutions/armeo-power/.

[3] "Armeo®spring." https://www.hocoma.com/solutions/armeo-spring/.

[4] "Armeo®senso." https://www.hocoma.com/solutions/armeo-senso/.

[5] P. Wang, "Dynamics modelling and linear control of quadcopter," IEEE.

[6] "Casadi documentation." https://web.casadi.org/docs/.