École Polytechnique Fédérale de Lausanne

Development of an MRI-compatible closed-loop device for robotic touch stimulation of the animal's back to investigate the neural correlates of presence hallucination.

by Karim Abi Said

Semester Project Report

Dr. Valerio Zerbi
Project Adviser

Dr. Olaf Blanke, Dr. Bruno Herbelin
Project cosupervisor

June 5, 2024

Contents

Chapter 1

Introduction

In this project, we study the design and development of an MRI-compatible closed-loop device for robotic touch stimulation of the animal's back to investigate the neural correlates of presence hallucination. Mainly, we will look into the challenges associated with building the system, testing it, and making it MRI-Compatible. The work is distributed into several objectives, split into scientific and practical objectives. The scientific objectives include:

- Understanding the issue of presence hallucinations in Parkinson's Disease
- Acquiring an in-depth understanding of the literature concerning rodent fMRI, looking mainly into the limitations of awake head-fixed fMRI rat imaging
- Understanding and implementing the a real-time closed-loop stimulation.

The practical objectives include:

- Looking into a feasible action of reach for the animal which is possible while the rat is head-and-body fixated, and the optimal real-time monitoring of this motion
- Developing and constructing a robotic system capable of performing a touch sensation on the animal (poking the animal) in a closed loop fashion
- Identifying pertinent features of rat hallucinations, and how we could monitor these features

Chapter 2

Scientific Objectives

This section is a brief background and introduction to the project. We will look into understanding presence hallucinations in Parkinson's disease, as well as the necessary limitations we have in awake rat fMRI, as well as the importance of the real-time closed-loop stimulation.

2.1 Comprehending Presence Hallucinations in Parkinson's Disease

Firstly, we will look briefly at presence hallucinations in Parkinson's disease, starting with presence hallucinations and Parkinson's disease in general. Parkinson's disease (PD) is a neurodegenerative disease affecting around 3% of the population over 65 years [2]. Generally, structural changes appear in the visual cortex, spanning the lateral and ventral occipito-temporal areas, the fusiform gyrus, and visual parietal areas [1]. Though PD is considered a motor disease in essence, it also has many non-motor symptoms that can begin to appear prior to the motor systems. A presence hallucination (PH) is a sensation that someone is nearby even when there is no one there, and can manifest in many different ways. Hallucinations can range from slight, such as misperceptions or having no form, to moderate and severe hallucinations, where the hallucination can appear as a person or a family member. Hallucinations can manifest in around 40% of patients with Parkinson's disease, and they are also associated with a higher mortality rate. Currently, in the Blanke lab, the team devised a protocol to induce presence hallucinations in both healthy patients as well as PD patients, these robotic-induces presence hallucinations (riPH) were simulated using a closed-loop robotic stimulator, which will be discussed in a further section.

2.2 Setting up rats inside fMRI: goals and challenges

Since we will be working with rats inside the fMRI, we will need to consider the setup of the animal. The rat will be awake, and its head must remain stable during the scan. We also must consider that it is a behavioural task we are studying, and the animal must theoretically be able to perform an action during the study. The size of the apparatus currently being used must be measured, including the outside and inside diameters, as well as the space available after the animal has been placed into the fMRI, in order to know what we can improve and what are the *strict* constraints. At present, the animal is enclosed in a pipe, the arms are tied to each other behind the animal's back, and the legs are also tied to each other and to the tail. The shoulders are pressed down and the head is restrained using a bite-bar system that prevents movement from the front. Our goal is to see what we can change in this case to allow for some movement without interfering with the fMRI signal.

2.3 Real-Time Closed Loop Stimulation

Having a closed-loop system to stimulate the animal is necessary to this experiment. This is because the paradigm goes as follows: The subject has a body part (in the case of a human, a finger) connected to a robotic mechanism. The subject is asked to move their hand around in 3 axes of motion, and the effect is divided into two parts. Firstly, the "control", where the robotic arm behind the patient follows the patient's finger in real time, using a master-slave mechanism [4]. This essentially produces a spatial conflict, where the subject feels as if he is poking in front of him and gets poked in his back, but the subject still knows that this effect is produced by themselves. The closed-loop is required to have the movements with no delay, as there is a positional controller following the motion of the subject in 3 planes of motion. It is important to also note that we will be introducing two more factors into play after this. The first is the bilateral feedback to the user: when they have reached their own back, the robotic finger should also stop so that they do not keep pressing on themselves. The second factor is the delay. After we have created the initial closed-loop follower, we will implement the same effect, but the slave device will follow the motion at certain user-defined delays, where in the study these were between 0 and 500ms of delay.

Chapter 3

Practical Objectives

In this section, we delve into the constraints required for the building of the system, as well as the different iterations that we go through while building the system.

3.1 Constraints and Requirements

The constraints for this experiment are as follows:

- Setup must be small enough to fit into the MRI cradle
- Setup must have fast information processing for real-time stimulation
- Setup must be MRI compatible

The considerations to take while implementing the setup are as follows:

- What actions do we want the animal to perform?
- How will the animal be oriented, and can this orientation be changed?
- How do we expect the animal to behave? Should we add more or fewer restrictions on the animal's movement?
- What kinds of sensors are required?
- Following the above question, can these sensors be changed in the case of being non-MRI compatible

3.2 Ideation Steps

3.2.1 Step 1: Identifying the feasible action for the animal

For this portion, we first get familiar with the system being used in the lab. The dimensions of the system, consisting of a cylinder with width 12.08cm, and a square space to place the animal with width 7.514cm. the maximum allowable height from the floor to the top of the cylinder is 9.16cm, and the width of the "floor" is 6.04cm. As for the tube, the outer diameter is 6.0cm and the inner diameter is 5.17cm.

After taking the measurements, the next step was to think of the placement of the animal. Can the rat be placed in a "sleeping" position, can it be "sitting"? The current setup has the rat placed prone on its stomach, with the bite bar and shoulder restraints as mentioned previously. Due to the nature of the recording and the important placement of the rat's brain, we cannot edit much the orientation of the animal. However, to implement a behaviour, especially one similar to the riPH setup in the Blanke lab, we can edit the current tube that the animal is placed in. After considering several different approaches, such as raising the tube on some legs, changing the tube shape, or scrapping the tube and using simply a snuggle bag as in [3], we decided the most feasible way is to simply drill a small hole into the current cylinder, as seen in Figure 3.1. This is because there is enough space to place the electronics (if they are small enough) onto the side of the tube. The rat snuggle bag, though a feasible idea, does require a nut to be drilled into the rat's head to create a fixation mechanism, which is undesirable in our case.

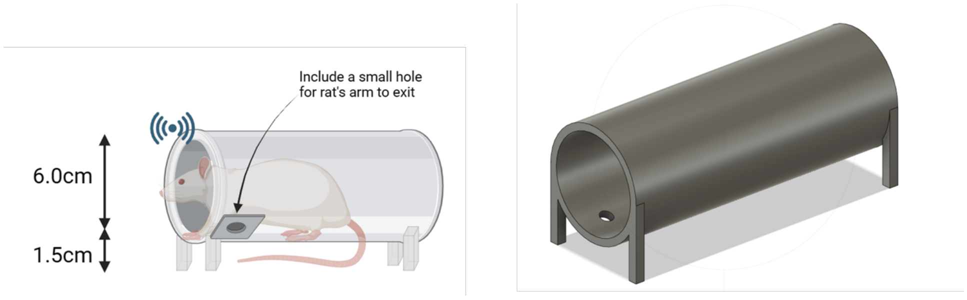


Figure 3.1: Example of modified Rat Tube as a 3-D model

Next, we consider several movements and actions that the animal might take. In this case, we want to consider the degrees of freedom of the animal, how it will perform the tactile stimulation, and the freedom of the animal's movement. To this end, we decided to go with a simple 1-DOF motion for simplicity (back and forth), and the stimulation will **prod** the animal at the butt area, to create the least amount of stress (since the back can be considered a predatory influence and cause the rat to panic. For the motion, after considering several possible movements (back and forth, unrestricted movements) we decided to have the animal's arm tied to a lever mechanism and pushing back and forth, and have a mechanism measuring the amount of movement.

3.2.2 Identifying mechanisms for the tactile rat stimulation

For the tactile stimulation mechanism, we look again in the same way, do we want to stimulate several portions of the animal's butt, one or two hemispheres, and how will we create this motion. We have decided on a linear motion (also 1-DOF) to poke the animal, following the motion of the arm connected to the lever.

The last portion is the sensory tactile feedback for the animal. In the protocol used for humans, the subject is asked to imagine an imaginary "wall" while exploring different locations on their back. Indeed, this is an issue since we cannot ask the animal to stop pushing forward when the butt is in contact with the robotic arm, so we must devise a feedback protocol as well as a "stopping mechanism" of the system. to not keep pushing the animal as the arm goes forward.

3.2.3 Identifying mechanisms to detect whether the rat is experiencing a hallucincation or some form of change

In order to know whether the rat is experiencing any form of change, or any difference between stimuli, we would need to assess its behaviour inside the limited space of the MRI cradle. We consider several possible mechanisms for this:

- Imaging techniques, such as using a camera inside the scanner to process whether the animal is shifting in synchrony with the movements.
- Using pupillometry to study whether the rats pupils dilate or shrink as a response to the tactile stimulus.
- Translating the current respiration module to both sides of the rat's cheeks, to detect movements that would otherwise be unnoticeable.

Chapter 4

Implementation

The following section will discuss the iterations of the system, looking at both the mechanical and electrical components and design choices for the system.

4.1 Iteration 1

In the first iteration, the goal is to get a working system that follows the constraints that we have placed, however the constraint of MRI-compatibility is flexible for this iteration, since we would like to have a system that functions in a closed-loop manner, and then fine-tune it with more dedicated hardware, as well as increase the functionality step by step.

4.1.1 Mechanical Design

We have created an initial model using Fusion 360 to house our mechanisms. The model consists of a rack-and-pinion mechanism connected to a servo motor to produce linear motion back and forth behind the animal; on the tactile stimulation side.

As for the rat's arm's side, there will also be a rack and pinion mechanism. The animal's arm will be connected

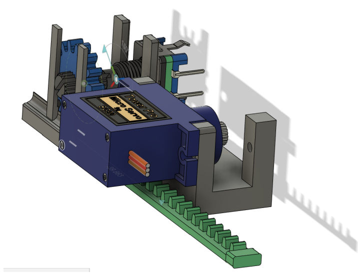


Figure 4.1: Encoder setup for the portion connected to the rat's arm

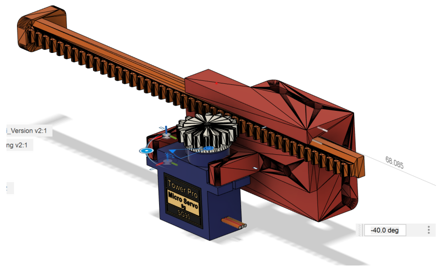


Figure 4.2: Setup for the animal robotic pusher

to the rack, and the pinion will be coupled to a rotary encoder which will allow us to measure how much the animal has moved its arm. Finally, we also model a braking system that we will want to test, that is inspired from typical caster brakes on chairs, but on the pinion. The 3-D models are shown in Figures 4.2 and 4.1.

4.1.2 Electrical Components

For this, we have decided to use an Arduino as the main controller for the system. We were able to program it to follow the encoder's position in real time. To do this, we read the values from the encoder, and map them to angles to input into the servo, and these angles are presented to the servo in real time where it is able to follow the movement almost instantaneously (the delay is negligible). Then, to add the delays we implement a circular buffer of size 200 elements (similar to a queue), which contains the "history" of the encoder for around 500ms, and based on the delay we require - which is set initially to 5ms delay every loop, we input the information from a certain index of the buffer to follow the actions of the encoder with a user-set delay.

Next, since we do not want the animal to push itself over the limit, we set a requirement that states that when the rat has touched itself, the system can no longer go forward, only backwards, automatically calibrating its own limits. This is implemented through a contact sensor at the tip of the robotic finger. The flowchart for the process is shown below in Figure 4.3.

4.1.3 Iteration 1 Comments and Possible Fixes:

The initial problems we had were firstly in the rotary encoder. The encoder has a limited resolution, and in our particular case, we could only read $16 \times 2 = 32$ portions of the circle, meaning our resolution was $\frac{360}{32} = 11.25$ degrees. This will be an issue concerning any small movements the animal might make, as we will want the motion to be relatively fluid.

The second problem is also related to the rotary encoder. The next issue is that the rotary encoder suffers from the issue of "debouncing". Since it has no external capacitors and resistors, and since it is a mechanical switching on and off of a switch inside the encoder as it clicks to the next position, it suffers from inaccuracies, leading the system to sometimes read the forward and backward motion incorrectly, especially when the movement is extremely quick. Also, when actually implementing the "bidirectional braking mechanism", the brake was too flimsy, and did not hold up due to its small size.

The final problem is the fact that the servo (for precision position control) can only move 180 degrees, and not

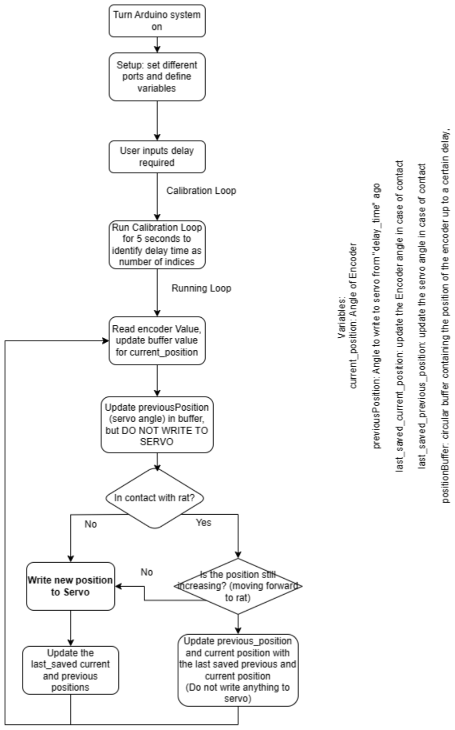


Figure 4.3: Simple Block Diagram of the Master-Slave Mechanism with Delay

more. This is due to the servo's mechanism using a potentiometer to accurately encode its location at a high resolution (we can map it from -90 degrees to 90 degrees down to single degree resolution). This might give us a limited range of motion, but it should be tested after implementing the entire initial setup.

After testing, since we are using a rack and pinion mechanism for the pusher, the range of motion of a 180 degree servo seemed to provide more than enough range, and in the case that it was unable to in an actual setup, then the gear can be increased in diameter, leading to more distance covered with the rotation.

Concerning the encoder, there are two possible fixes for this. Firstly, is to use a simple potentiometer. The potentiometer can measure distance much more accurately and down to a much smaller resolution than the rotary encoder, but the issue arises is that it uses resistive elements and might not be MRI-compatible for later on. The second solution is to use another higher-resolution MRI-compatible rotary/linear encoder. This would require the setup and housing of the system to change, but that is necessarily part of the end design. One technique is to use an optical rotary/linear encoder such as the system in MICRONOR (MR303 linear or MR302 rotary) which are both MRI-compatible, but require extra circuitry and another controller, which might complicate the setup further and prevent us from interfacing with the Arduino. However, we will keep the idea of using an optical linear or rotary encoder, but we also mention the fact that the pre-made rotary encoders are somewhat bulkier than the linear encoders.

Next, concerning the servo issue, as a solution, we note the possibility of using a different mechanism which is operated pneumatically, and can be operated from afar since there is only the pneumatic device (which should not have any metal parts) inside the scanner.

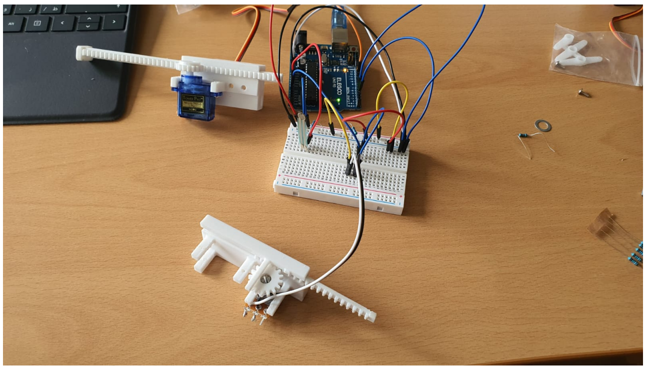


Figure 4.4: Physical setup of the master-slave mechanism

4.2 Iteration 2

4.2.1 Mechanical Components

For this iteration, we kept the same mechanical setup but replaced the rotary encoder with a potentiometer. As expected, the potentiometer gives us an "infinite" resolution (physically reduced to 1 degree due to the software encoding of the servo). However, it is much more responsive and sensitive than a basic rotary encoder. Figure 4.4 displays the physical setup of the 3-D printed components placed all together.

4.2.2 Electrical Components

Since we changed from a digital device (the rotary encoder has a clock that requires debouncing and "counting") to an analog one, where the potentiometer is simply having values read off of it, the flow of the arduino program changed slightly. This is since we are constantly reading an absolute position instead of a relative position which we can reset as in the encoder. Below in figure 4.5 is the flowchart for the code where a potentiometer is used.

4.3 Iteration 3

4.3.1 Mechanical Components

Moving forward, we would like to build a system that can go into the cradle all together, and would be possible to test in vivo. We would also like a system that follows the constraint of being simple and easy to move by the animal, and so we are looking at different mechanisms of measurement than mechanical coupling to an encoder or a potentiometer. One mechanism we have considered for this case is an optical time of flight (TOF) sensor, which uses the time that a laser takes to travel forward and be reflected by a surface. After initial testing with a VL53L0X time of flight sensor, the measurements had some amount of noise, but the precision of ± 1 mm, which is acceptable for our case.

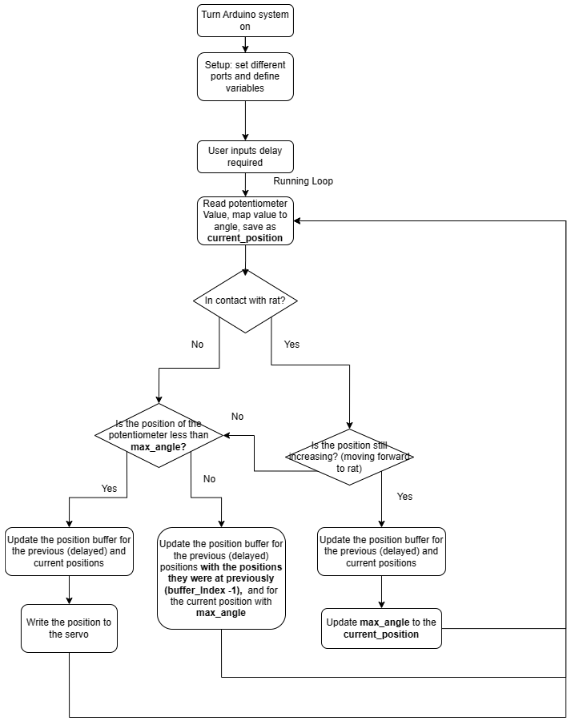


Figure 4.5: Block diagram of code with potentiometer

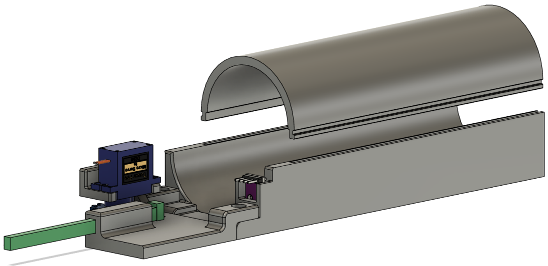
Next we designed the 3D model of the whole system, with a stick that the animal holds on to and implementing the previous servo motor rack and pinion mechanism. The model of the setup is shown in Figure 4.6a. Finally, the entire setup is shown in Figures 4.6b and 4.6c.

4.3.2 Electrical Components

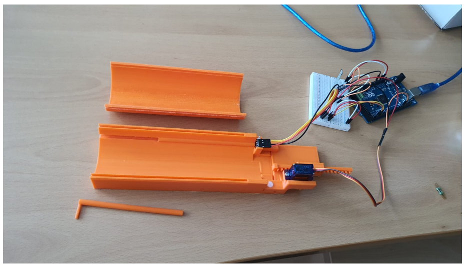
Since the TOF sensor gives us an absolute position, we can simply use the code with the potentiometer from the previous system, taking into account the calibration of the sensor to translate the distance in mm to the angle of the servo. Furthermore, the TOF sensor seems to require a different source of power, so we might resort to using a different arduino to power the servo. One note is that since the servo is placed upside-down, the angle written to the servo should map 0-179 degress into 179-0 degrees, effectively reversing its position to follow the location of the handle. However, we have had to make several changes to the code, including a calibration loop since the delay of the sensor is indeed substantial, with the next iteration of the code found in the figure 4.12.

4.3.3 Rat Reaction to the Stimulation

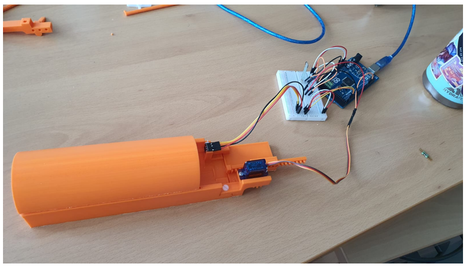
The simplest method we can employ to this end is to use two pressure sensors on either side of the animal's cheek to simply graph the movement of the animal's head, and be able to sense small jitters of the animal. To this end, we modified the neck and shoulder bars to have a wider (0.5cm from each side) space for the rat's face,



(a) 3D model of Iteration 3 of the system



(b) Setup of Iteration 3 Separated



(c) Setup of Iteration 3 Assembled

Figure 4.6: Iteration 3 Setup

and hence to be able to place the pressure sensors on the sides effectively. The modified shoulder and neck bars are shown below, in Figure 4.7.

4.3.4 Iteration 3 comments and possible fixes

One issue in this iteration is the use of the VL53L0X time of flight sensor. Since this is a cheap and relatively easily obtainable sensor, the accuracy of the measurements is not guaranteed. Especially with many concurrent and repeated measurements, the sensor cannot be used effectively as a "digital ruler". Especially with the quick repeated measurements, the system reads many measurements very quickly, and causes the servo to be very jittery back and forth. However, there are several fixes that we employed to target this issue in software:

1. Rounding to the nearest 5mm, where the location only changes if there was a difference of more than 5mm, which is theoretically small enough to still have the servo be able to follow in real time, and large enough to not cause any jittering issues. However, that led to larger "jumps" when the distance was hovering around a multiple of 5mm. For example between 32 and 33mm, will be detected as jumping between 30 and 35mm, which is not ideal.

2. Increasing the timing budget of the vision sensor. What this allows is for more time to measure the light bouncing off the object between measurements, effectively allowing for the system to be more accurate, at the cost of a bit more time per measurement. However, the amount of delay induced per

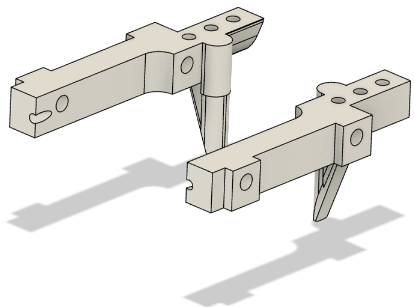


Figure 4.7: Modified Shoulder and Neck 3D model

measurement is in the order of a couple of milliseconds, and does not affect the measurement much. This does help significantly, but the jitter remains existent.

3. Finally, we could apply a low pass moving average filter, by considering the last 5 measurements and averaging them out instead of only the current measurement. This does incur a delay however, since each loop is taking 10-15ms to run, and so averaging the last 5 loops will give us a substantial delay of around 30-40ms, calculated in the appendix A.1. It is able to further remove the jitter substantially though. However, we can also fix this by taking less time induced in the delay of the loop incurred in the arduino.

4. Finally, we could also consider getting a different version of the sensor such as the VL53L1X sensor, or any of the similar ones in that line which prioritize accuracy with a small measurement distance. This can be found in Table 4.1 below.

Vision Sensor Model	**Range**	**Best Linear Response Range**	**Resolution (approx. mm)**	**Viewing Angle**	**Cost**	**Sampling Frequency**
VL53L0X	30mm - 1200mm	50mm - 1200mm	Varies; down to 1mm	-	$14.95	33Hz
VL53L1X	Up to 4000mm	Up to 4000mm	Varies; down to 1mm	27°	$19.95	Up to 50 Hz
VL53L4CX	1mm to 6000mm	Up to 6000mm	Varies; down to 10mm	18°	$14.95	Not specified
VL53L4CD	1mm - 1300mm	0mm - 1200mm	Varies; down to 1mm	18°	$14.95	Up to 100 Hz
VL6180X	5mm - 100mm	5mm - 100mm	Varies; possible down to sub-mm	-	$13.95	Up to 100 Hz

Table 4.1: Comparison of Adafruit Time of Flight Sensors

4.3.5 In Vivo testing

Next, to truly understand our positives and limitations, we moved to our first in-vivo experiment. We began by working with an anaesthetized rat, to test the fit of the setup and the working of the entire system.

At once after placing the animal inside the housing and cradle, we noticed several points.

- The stick connected to the rat's arm must be longer, due to the limited range of the sensor, especially when measuring through the small hole. The reason for this is explained in the appendix A.2
- The rat must have its arm/hand taped to the handle, *before* placing it into the housing. This is since the handle is quite long (longer than the housing) and cannot be placed after the rat has been set in the housing.
- The VL53L0X sensor is very jittery, and indeed the servo motor will move very frequently due to the high noise of the sensor. This re-emphasizes the necessity of acquiring a better sensor.
- The initial shoulder bar restraints on the rat that were placed in the original cradle had to be removed on the side that the animal is moving its hand.
- When the rat is awake, all the electronics must be protected from the organic waste of the animal, which can be especially damaging to exposed electronics.

Hence, we must target all these issues, and work while keeping these in mind to improve our design. However, overall, the design was functional and the test on the rat, proved to be able to follow the rat's arm movement, albeit with a small delay.

4.4 Iteration 4

4.4.1 Electrical Components

We replaced the VLX53L0X sensor with the VL6810X TOF sensor. This effectively allowed us to reach a 10ms sample time per distance measurement, and through testing, every loop takes a delay of 11ms, with the moving average delay reaching 27ms. This should be fast enough to be effectively unnoticeable by the rat. We must also note the physical speed of the servo: it is provided with a position, and moves to the position as fast as it can, meaning we are also limited by the physical speed of the servo. However it must also be noted that the animal's movement is indeed continuous, so the servo location will not be quickly moving towards extremely far locations. However, the motion of the animal's arm can be faster than the servo, especially with it "yanking" the stick. This can be shown through the data in the plot in figure 4.10, where we can see that the animal moves its hands to the two extremes in less than 100 ms. The final connections of the components can be found in figure A.2.

4.4.2 Mechanical Components

We have also scaled the size of the housing to 68:61, to be able to accommodate larger animals. We have also changed the size of the sensor housing on the right and back of the housing, to fit the VL6810X to size. Two more changes were allowing the sliding top of the house to fit from the back, rather the front, in order to be able to fit the rat while it is under anaesthetic. Finally, we made the connection to the pinion higher, as the current system pushes the animal very close to the tail, and we need to push it on its butt, hence we allow for a higher push.

4.4.3 Iteration 4 Comments

The current system has much better sensitivity and less delay, allowing for a smoother transition of the servo. It is true that the jitter is still existent, but to a much smaller amount. The issues for this iteration were few, and we were able to extract data in the form of videos, images, and data recording off of arduino.

However, several issues remain:

- The contact sensor does sense when the rat is being touched, but must have its sensitivity and coverage increased heavily, due to the possibility of the rat moving or shifting around in the housing. A possible fix for this could be to restrain the animal even from the back, preventing it from moving.
- The issue concerning the limited range of the rat's motion is still present, and should be tackled. This could be possibly done in several ways. One way is to extend the stick even further and simply measure from the outside, but that would require a longer stick, and hence a more difficult process to setup the rat inside. Another way could be to increase further the diameter of the hole, allowing for a wider range of vision of the sensor, and consequently a larger possible range.

4.4.4 In Vivo Testing

For the tests in vivo, and in this setup in particular, the setup was done as follows:

1. Put the animal under anaesthesia and restrain one of the arms and both feet.
2. Tie the stick to the animal's hand
3. Place the animal into the bottom part of the housing and the stick into the hole
4. Place the housing onto the cradle and fix the animal's mouth onto the bite bar
5. Place the VL6180X sensor into its slot
6. Close the housing from the back
7. Fix the shoulder bars.

(a) Angle 1 (b) Angle 2 (c) Angle 3

Figure 4.8: Different Angles of the animal's placement

The figures showing the different angles of the rat is shown below, in figure 4.8

Furthermore, the next two images show the animal with its hand (and hence also the servo) in the two different positions, contracted and stretched out, in figures 4.9a and 4.9b. However, for a clearer view of the images and the movement with and without delay, it is recommended to refer to the videos in the github link in A.4. It must be noted that all the videos are in slow motion (8x slowed).

4.4.5 Data Collection and Recording

Finally, to record our data, we have created two python scripts:

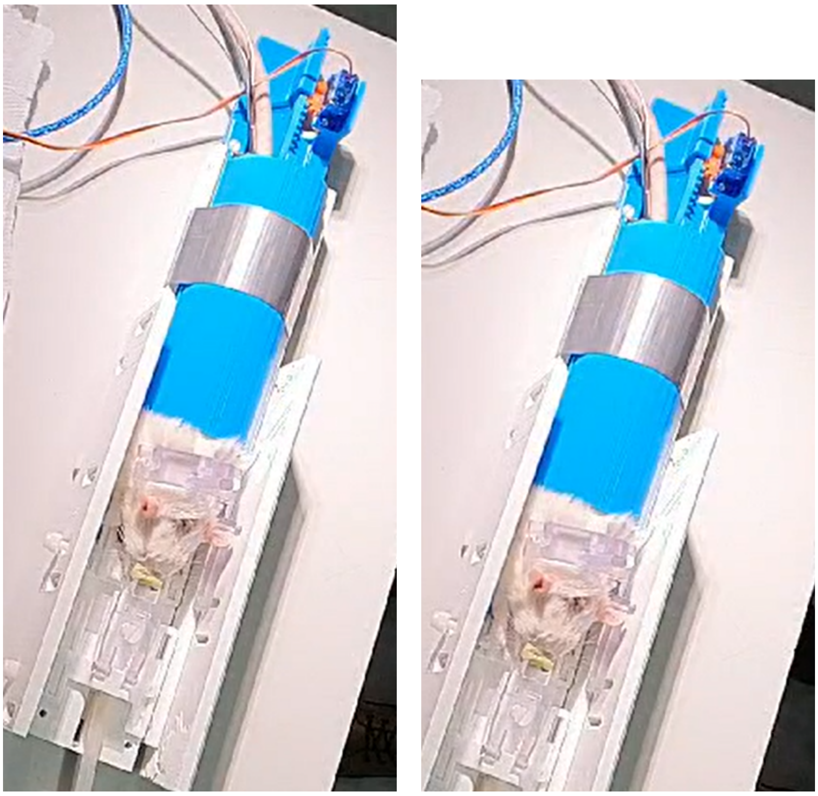
- `Serial_to_CSV.py`, which the user can edit the delay given to the servo, as well as change the name of the CSV file they are writing to, before running the script.

  To run this code, the user must first attach the arduino, and input which "COM" serial port the arduino is attached to, since it differs from device to device. Then, simply run the code to begin recording.

  To stop or edit the code, simply apply a KeyboardInterrupt (Ctrl + C), and restart the code.

  The code fills in currently 6 columns of data:

  - Time Elapsed
  - The maximum angle the servo can move to (in degrees)



(a) Hand Pulled back (b) Hand Pushed to the front

Figure 4.9: Animal's hand stretched and contracted

- Contact with the animal (Arbitrary Units)
- The delay time of the servo motor (in ms)
- The raw data from the VL6180X sensor (in mm)
- The filtered and delayed data sent to the Servo Motor (in degrees)

- `Plotting_CSV.py`, which plots the data from the CSV onto a python plot, as shown in the example in figure 4.11.

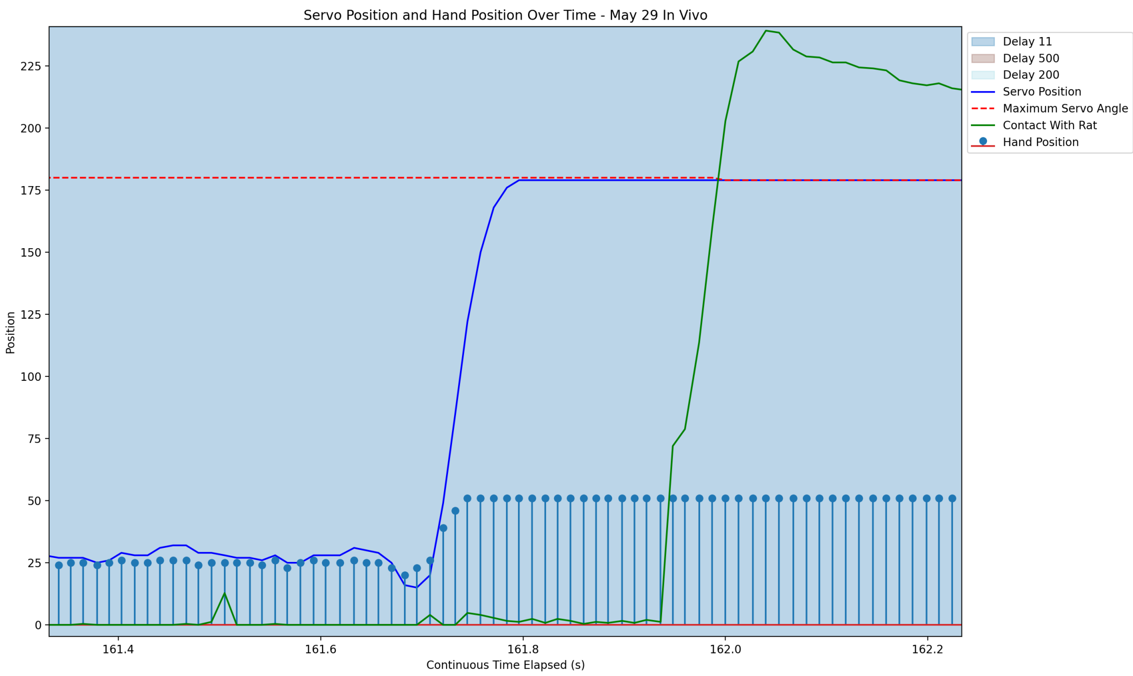


Figure 4.10: Plot showing the motion of the rat's arm (in stem points) and the motion of the servo after

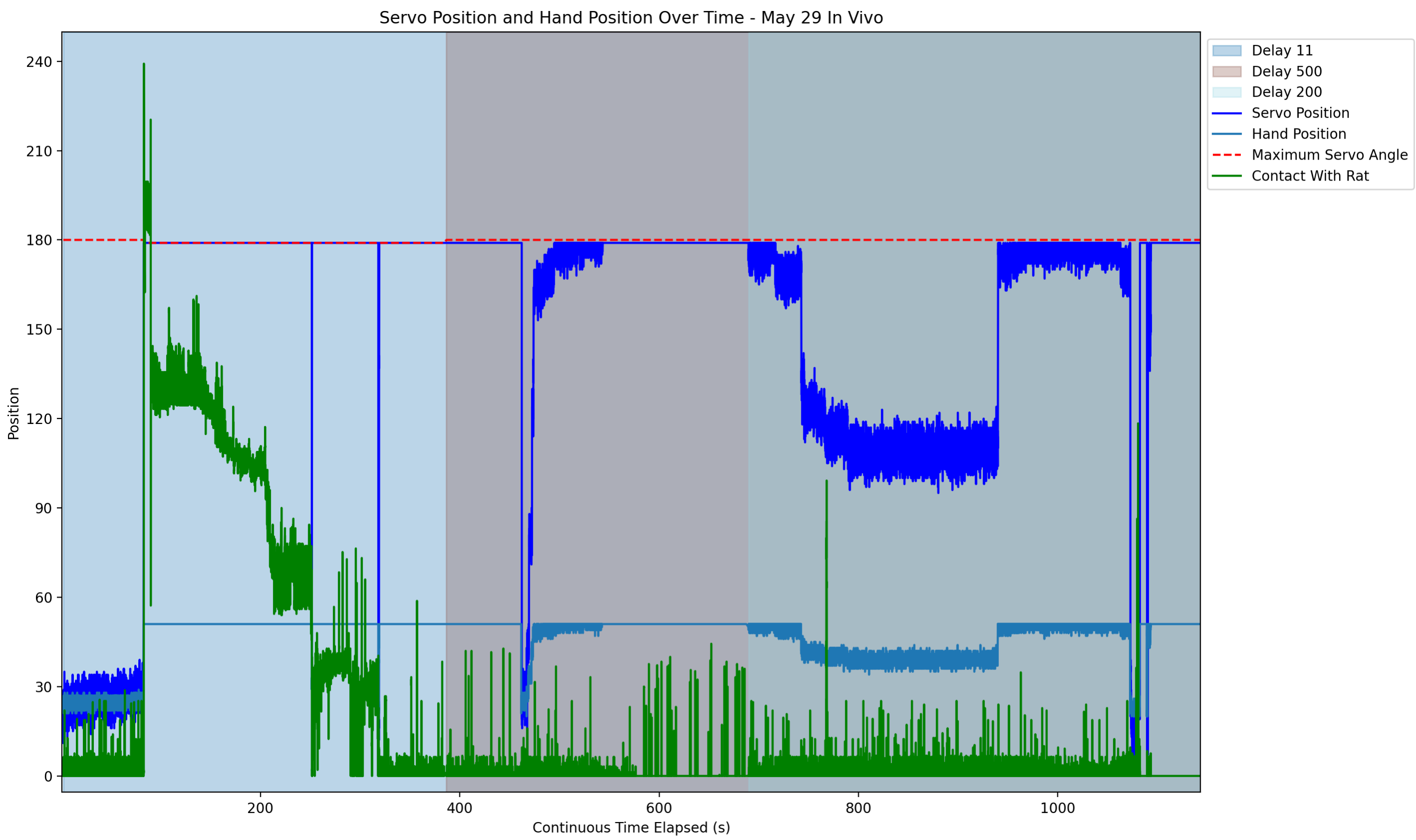


Figure 4.11: Example of a plot from data taken in in vivo experiment

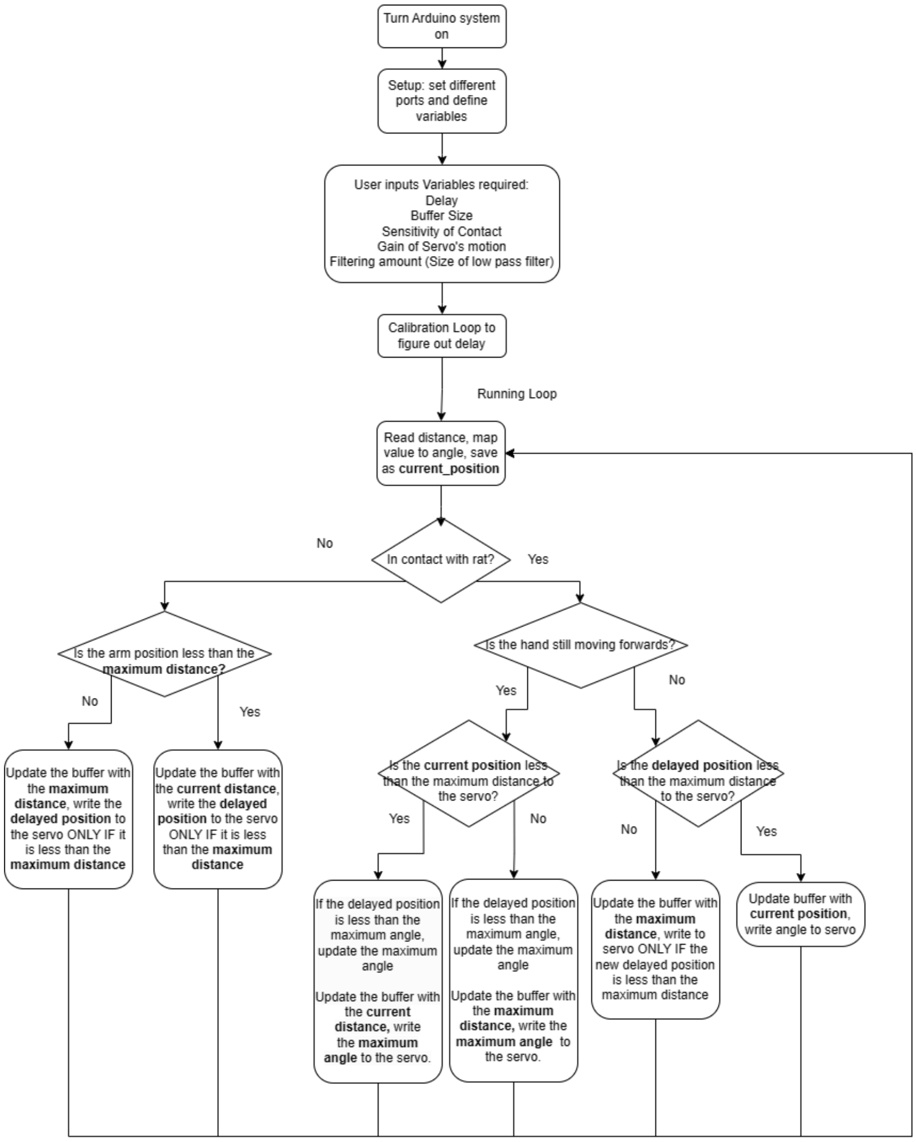


Figure 4.12: Code with the Vision sensor and Calibration

Chapter 5

Conclusion, Limitations and Future Outlook

In conclusion, we have created a responsive system capable of following an animal's motion almost in real time, with a delay that is easily changeable between experiments. However, there were several limitations to the study that must be addressed, and future considerations to look into.

Limitations:

- The current animals being used are very well trained to remain calm and static inside the MRI-scanner. This means there is no real incentive to move forward and backward, even in an attempt to escape.
- The setup is inexpensive, which is both a positive and a limitation, since the vision sensor, contact sensor, and servo can relatively easily malfunction or break.
- It is quite difficult to quantify the understanding of whether the animal is experiencing hallucinations, and we have not been able to test this due to the limited tome to test the awake version of the entire setup.

Future Improvements:

- To target the idea of the animal not having an incentive to move, we could implement a reward system for an untrained animal to receive.
- Concerning the issue of MRI compatibility and inaccuracy of recordings, there are several possible replacements for the servo and the distance sensor. For the servo, an ultrasonic motor from xeryon, namely the **XLA-3-125-50MU Linear Actuator**, which should be ordered with the pieces separate, and have the prodding stick 3D printed, as it is made of steel while the rest of the housing is non-magnetic. As for the time of flight sensor, a baumer laser time of flight sensor can also be used.
- An improvement on the shoulder restraints can be made, to hold the animal's neck in place since the arms are now able to move back and forth.

However, a base to build upon is now proposed, and shown that indeed an animal can move while being relatively constrained, and can take actions and be stimulated with a novel sensory cue (tactile) inside an imaging system such as an MRI.

Bibliography

[1] Fosco Bernasconi, Eva Blondiaux, Jevita Potheegadoo, Giedre Stripeikyte, Javier Pagonabarraga, Helena Bejr-Kasem, Michela Bassolino, Michel Akselrod, Saul Martinez-Horta, Frederic Sampedro, Masayuki Hara, Judit Horvath, Matteo Franza, Stéphanie Konik, Matthieu Bereau, Joseph-André Ghika, Pierre R Burkhard, Dimitri Van De Ville, Nathan Faivre, Giulio Rognini, Paul Krack, Jaime Kulisevsky, and Olaf Blanke. "Robot-induced hallucinations in Parkinson's disease depend on altered sensorimotor processing in fronto-temporal network". en. In: *Sci. Transl. Med.* 13.591 (Apr. 2021).

[2] Fosco Bernasconi, Javier Md, Helena Bejr-kasem, Saúl Martinez-Horta, Jaime Kulisevsky, and Olaf Blanke. "Theta oscillations and minor hallucinations in Parkinson's disease reveal decrease in frontal lobe functions and later cognitive decline". In: (Nov. 2022). DOI: `10.1101/2022.11.24.517668`.

[3] Pei-Ching Chang, Daniel Procissi, Qiyuan Bao, Maria Virginia Centeno, Alex Baria, and A. Vania Apkarian. "Novel method for functional brain imaging in awake minimally restrained rats". In: *Journal of Neurophysiology* 116.1 (2016). PMID: 27052584, pp. 61–80. DOI: `10.1152/jn.01078.2015`. eprint: `https://doi.org/10.1152/jn.01078.2015`. URL: `https://doi.org/10.1152/jn.01078.2015`.

[4] M Hara, G Rognini, N Evans, O Blanke, A Yamamoto, H Bleuler, and T Higuchi. "A novel approach to the manipulation of body-parts ownership using a bilateral master-slave system". In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. San Francisco, CA: IEEE, Sept. 2011.

Appendix A

A.1 Calculation of delay incurred by the moving average filter

In order to minimize jitter while still keeping the system relatively real time, we have to employ a smoothing filter with enough filtering to stop high frequency components, but not causing the system to suffer from too much delay. Hence, after several iterations, we are using a moving average filter with a transfer function of $h[n] = \delta[n] + 0.75\delta[n-1] + 0.5\delta[n-2] + 0.5\delta[n-3] + 0.25\delta[n-4]$, where n is the discrete time point for every loop. To find the running time of the loop, we also run the calibration loop for 2 seconds, count the number of loops, and then divide by the time elapsed. In the high accuracy mode of the vision sensor, our loop (and hence our sampling time) is consistently 38ms. Hence, we are able to calculate the delay incurred by the sensor through simple calculation:

Delay $= \frac{\sum_{i=0}^{N-1} w_i \times i}{\sum_{i=0}^{N-1} w_i} \times$ Sampling Time, where w_i represents the i'th weight and i represents the delay in terms of number of samples n.

Plugging into this equation,

$$\text{Delay} = \frac{(1\times 0) + (0.75\times 1) + (0.5\times 2) + (0.5\times 3) + (0.25\times 4)}{1 + 0.75 + 0.5 + 0.5 + 0.25} \times \text{Sampling Time}$$

$$\text{Delay} = \frac{0 + 0.75 + 1 + 1.5 + 1}{3.75} \times \text{Sampling Time}$$

$$\text{Delay} = \frac{4.25}{3.75} \times \text{Sampling Time}$$

$$\text{Delay} = 1.133 \times 0.038$$

$$\text{Delay} \approx 0.043 \text{ seconds}$$

So, the delay introduced by this weighted moving average filter would be approximately 43 milliseconds.

A.2 Reason for limited range of sensing

As we can see from the datasheet in figure A.1, the visual angle of the VL53L0X - and all the sensors of this series, (VL6180X included), have a sensing angle of around 25 degrees. When looking through a hole, at some distance the measurement recorded by the sensor will no longer be the animal's arm movement, but rather it will simply be sensing the lateral distance of the walls, which would be around 60mm away. This is for the for the further distance, whereas the closer distance will be *recorded* at around 20mm close to the sensor, simply due to the limitations of the hardware.

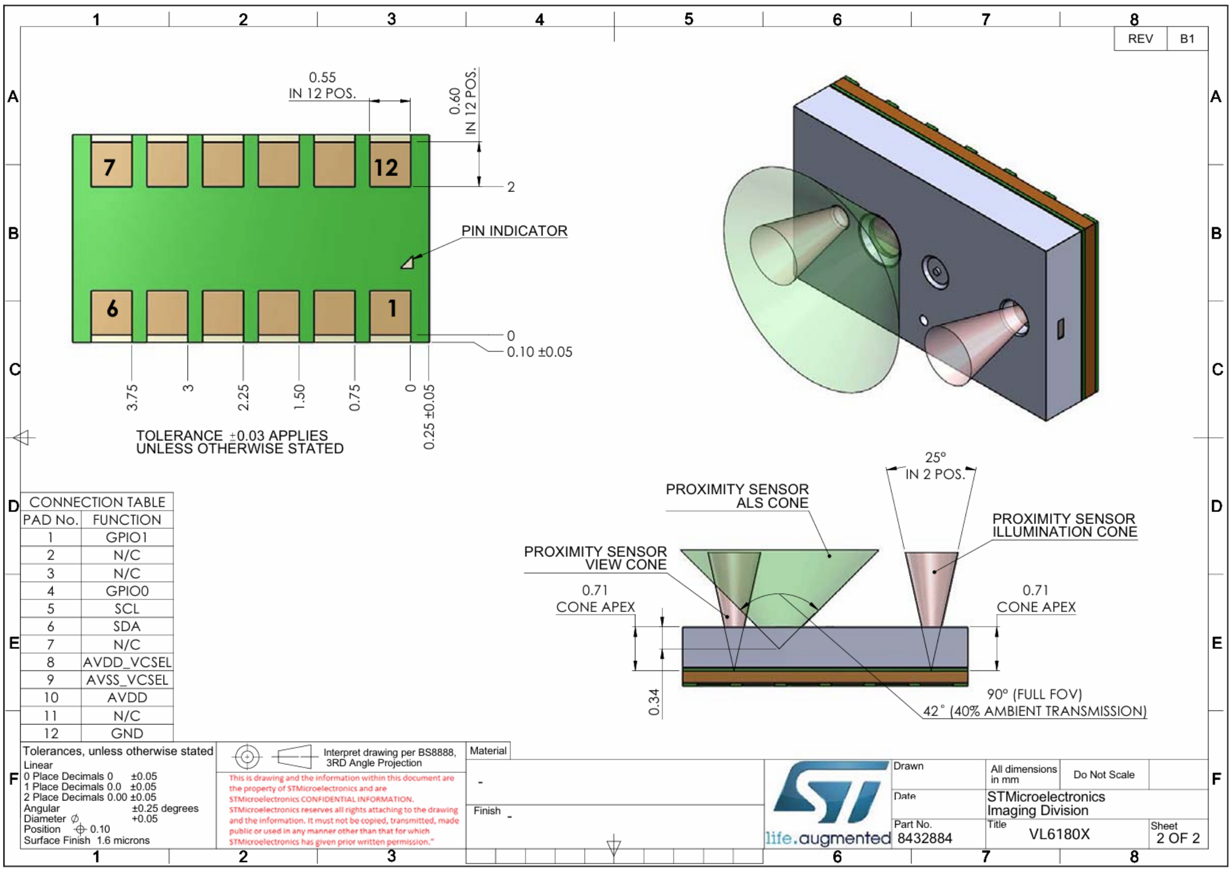


Figure A.1: VL8180X angle of Sight

A.3 Arduino Electrical Connections

Below is an image displaying the connections of the arduino mega.

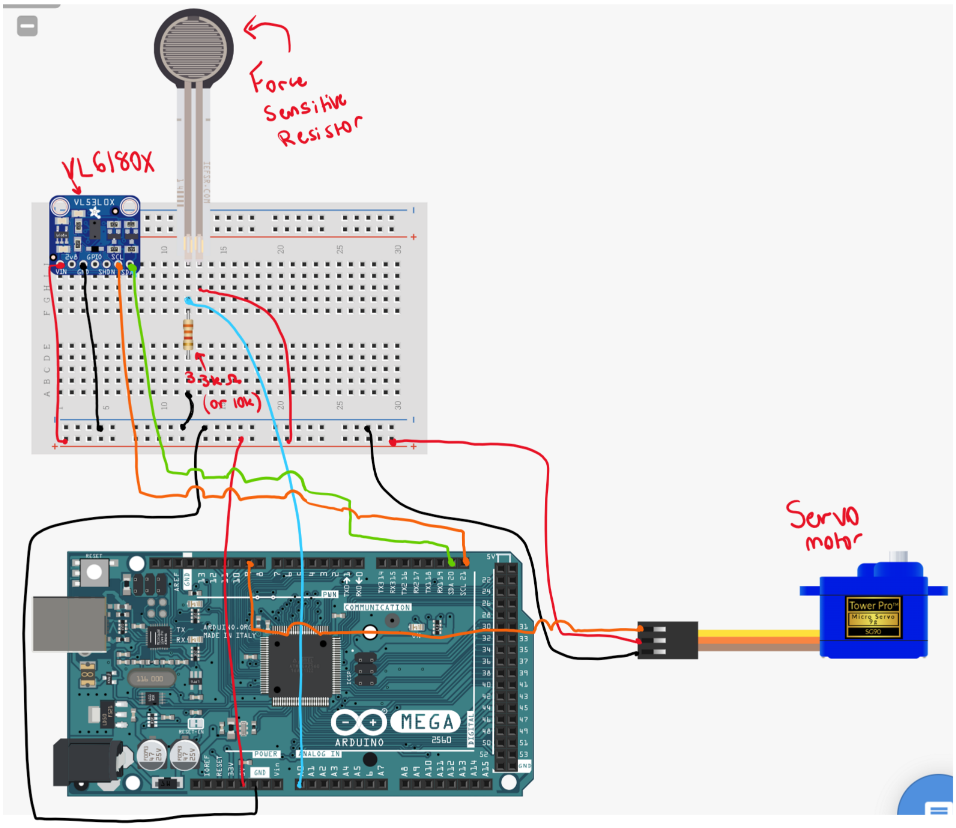


Figure A.2: Electrical connections for arudino circuit

A.4 Videos and Images, and access to code

All supplementary videos and code can be found on the github link [Click Here]