ROBOTARM

Feil Charel

JANUARY 30, 2022
BTS-IOT
Lycée des Arts et Métiers
Project CREPO1 2021/2022
1, rue de l'école,
Bissen L-7768
Supervisors: Mr. Bernard, Mr Weiler

Table of contents

1 Description of the task

The task given for this project was to produce a robotic arm in one semester, which at first can seem overwhelming but the project has shown all or most of us that it is feasible.

The goal of this project is to produce the robot arm in a way that it can move freely in a space of at least 15x15 centimetres, in that space it should be able to lift an object at least 10cm high and put it down safely on another place on the same height level.
The object that the robot must grab and move in space has been defined as a general cubic shape of maximum 40 millimetres side-length and a maximum weight of about 25 grams.

How the robotic arm works exactly is freely decided by the students, but the arm must be controlled by a microcontroller that can be chosen freely and must be programmed.
The build-up of the robotic arm is also completely up to the imagination of the students, it just must fulfil the minimum requirements.

Those minimum requirements are that the final product can execute a movement, which is predefined to move that before mentioned cube from one predefined place to another predefined location at the same height. It also must run on cables from a power supply and not on batteries. Another requirement is that the robotic arm is easily maintainable, so not gluing all the parts together in a way that disassembly isn't possible anymore in case of something breaking or not working anymore.

There were also possible improvements given which are that the objects, so the cubes, orientation in space can be defined, which would allow the arm to pick the cube up and put it down while maintaining the same orientation, no rotation.

Another improvement would be that the robotic arm would be controllable with a joystick or gamepad of some kind, which are connected to the microcontroller that controls the movement of the robotic arm.

The last improvement given, which was described as "BEST", was that the robotic arm could be controlled by sending G-code messages through the serial input.

2 Introduction

The beginning of the project was a bit overwhelming at first, but after some time the ideas started flowing, good and bad ones. It certainly wasn't as easy as I anticipated because I had no experience in 3D modelling and 3D printing and at first I couldn't assess if the part, I designed would hold the weight.

In the next few pages, I will explain how I built the robot arm and what I learned, which ideas failed, and which succeeded in the end.

The first thing I looked up was how a robotic arm works, how they move around, what motors can be used, what materials would be the easiest to make for myself, because I only had a rough idea and wasn't so sure.

In the beginning my thought was to give myself enough time to test everything in case something fails or breaks that I have time to correct or replace it. This was easier said than done, because in the end it still has been quite some stress, self-induced and very much avoidable stress.

2.1 How it works

The robot arm can grab a small box from one defined place and safely put it down on another defined place. When the robot arm is turned on it stretches the arm straight up and searches for its middle position.

The robot arm is mostly 3D-printed, the arm itself consists of four main parts; the arm base or "shoulder", where the base motor is placed, the "upper arm" part, no motors are directly mounted to this piece, but it is hollow to allow for better cable management, the "under arm", where the "elbow" and "wrist" motors are mounted and finally the "hand", which consists of a number of parts and houses the grabbing motor.

The base of the arm is mounted on a wooden turn table with most of the electronics, which include the Arduino, which controls everything and the motor driver which allows the Arduino to also control a stepper motor.

The stepper motor can turn endlessly and is controlled by making steps in one direction, the servo motors can only turn about 180° and are controlled by changing the angle.

The wooden turn table is driven by a stepper motor, which is connected to the turn table by a belt, this allows the robot arm to turn. The rest of the robot arm functions with servo motors to control the exact position of the arm in space.

The robot arm searches for the middle position by firstly turning clockwise until the first button hits the definition wall, then it turns counter clockwise until the other button hits the definition wall after it knows how far it is from one end to the other, it turns clockwise until it reaches the exact midpoint of that distance.

Then, the robot arm will be in standby until both buttons are pressed to indicate the start of the movement. Both buttons must be pressed, because this case should never occur, the user must press them to initiate the grabbing movement.

Firstly, the robot arm will turn clockwise to one quarter of the distance measured, so about 45° measured from the midpoint. After that the servo on the base will change its angle, then the one from the “elbow”, after that the one from the “wrist” turns to an angle to pick up an object from that defined position.

Afterwards, the robot arm stretches to its top position and turns half the measured distance counter clockwise, so about 90°.

Then it again adjusts the angles of the servo motors to safely put down the object it’s holding and return to the position it began in the first place.

3 Main part

3.1 Description of the task

The basic task was to produce a robotic arm, which can lift a small cube of 40mm side length and 25g weight maximum.

It must move the cube in a space of minimum 15x15cm and be able to lift it 10cm high.

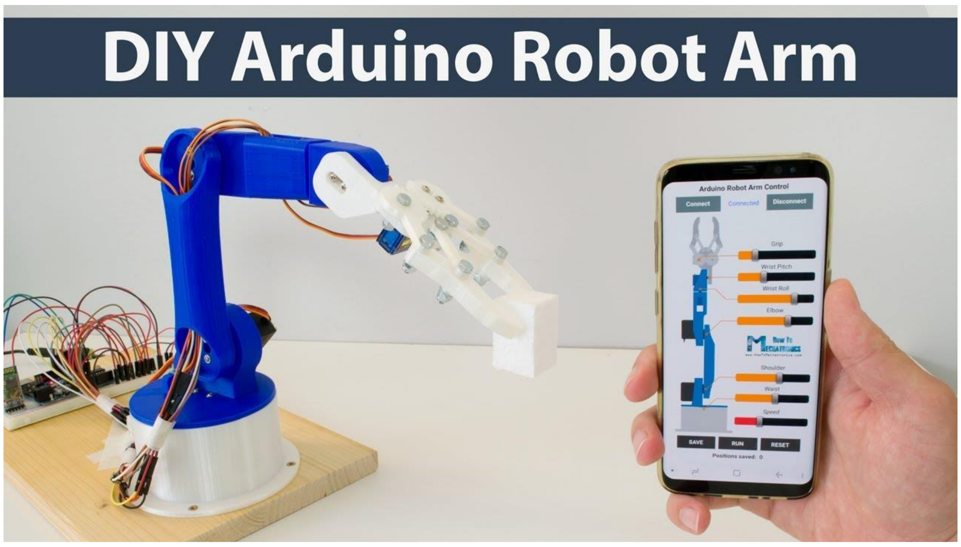
3.2 Step-by-Step description

3.2.1 Beginning ideas

At first my idea was to make the robotic arm functional and that it has a good design. However, I gave that up pretty fast when I came to design the grabbing mechanism, there I just wanted it to work.

At the beginning I planned a lot of time for the design work, because I had no experience in 3D modelling and designing and had to learn it from scratch. Thankfully, Mr. Bernard held a course in which he taught us how to use the 3D modelling program provided by the school.

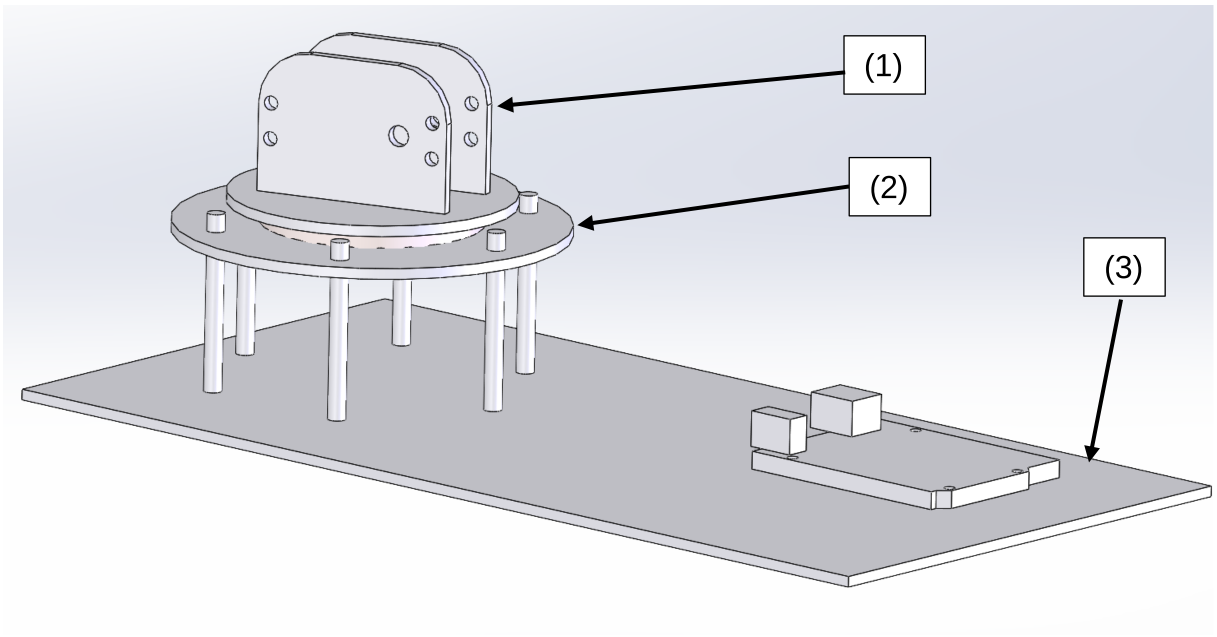
Firstly, I googled some robot arm designs and was mainly inspired by this one, but it changed a bit over the course of the project.



*Link 1

The idea to feed the cables though the arm itself especially pleased me and I wanted to reconstruct it.

My first idea for the base of my robotic arm was to take a steel plate and elevate the arm on a smaller platform, which you can see here:



(1): Base part that holds the base servo motor and rotates
(2): Plate which holds everything up and allows the base part to rotate
(3): Steel plate which also acts as a counterweight, so that the arm can't topple over

I wanted to have full control over the arm, that's why I had the idea to put a servo motor in every axis, this way I can define every angle and how the arm is standing at a given moment.

I had initially only planned 4 motors in my design: the turning motor, the base motor, the shoulder motor and the grabbing motor. This turned out to be a mistake and I had to add a 5th motor, the wrist motor, to my design.

3.2.2 *Designing and 3D-printing*

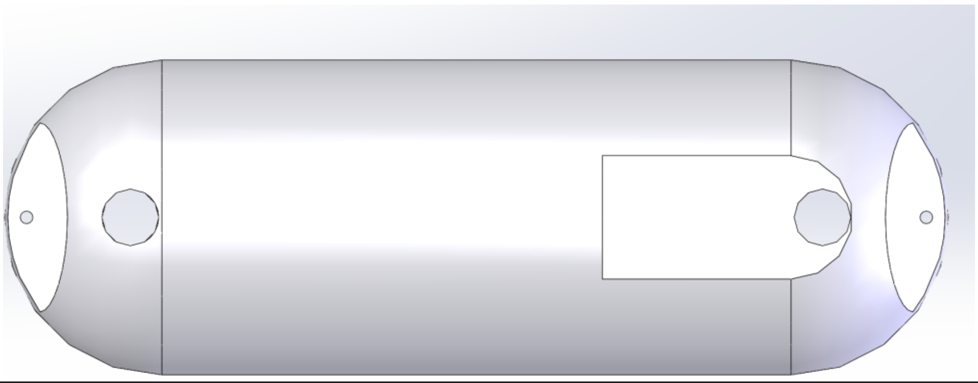
3.2.2.1 Designing

As previously said, the designing and printing took most of the time I had, because I had to learn everything from scratch. To make a 3D model of the idea I had, I first drew a very rough sketch on paper how the silhouette should look like, this wasn't very representative of how I imagined it.

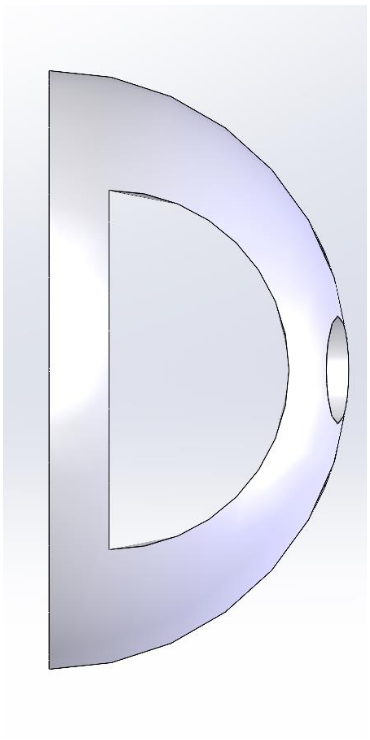
The first thing I designed was the base and how it works, because I was the most certain on that part of my initial idea. You can see the SolidWorks Assembly of these base parts in the point *3.2.1*. .

3.2.2.1.1 Part 1

The first part I designed of the arm itself was the part that connected the base motor with the shoulder motor, the part has a half-round, long shape, it is hollow to feed the cables from the other motors through and it has a flat spot on the round side to prevent interference with the base part:



Here you can see the round and long shape and also the flat spot to prevent interference. The holes on both sides are there to support the part with an axis and that the Servo motor can be easily removed and is accessible.

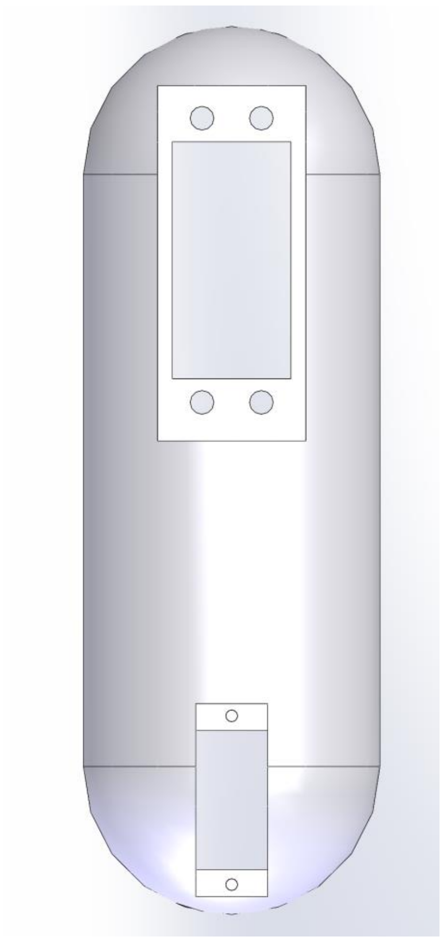


On the left you can see the hollowness of the part.

On the right you can see the holes for the axis and the notches to insert crosses that can connect to the servo motors of the part.

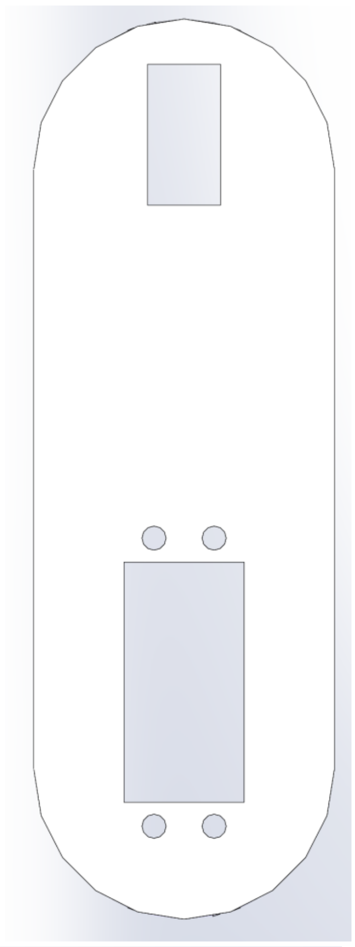
3.2.2.1.2 Part 2

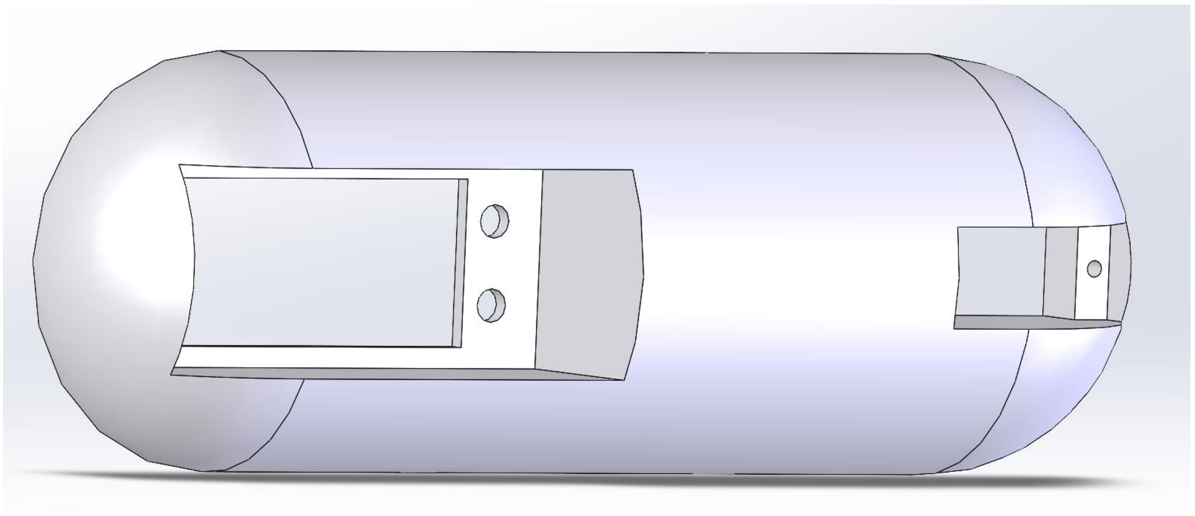
The next part I designed was the part that held the shoulder and wrist motor. This part has a very similar shape than the first one, but it is solid, and it only has notches to house the servo motors and hold them securely in place. This part had to be modified in the end, I will further explain this later.



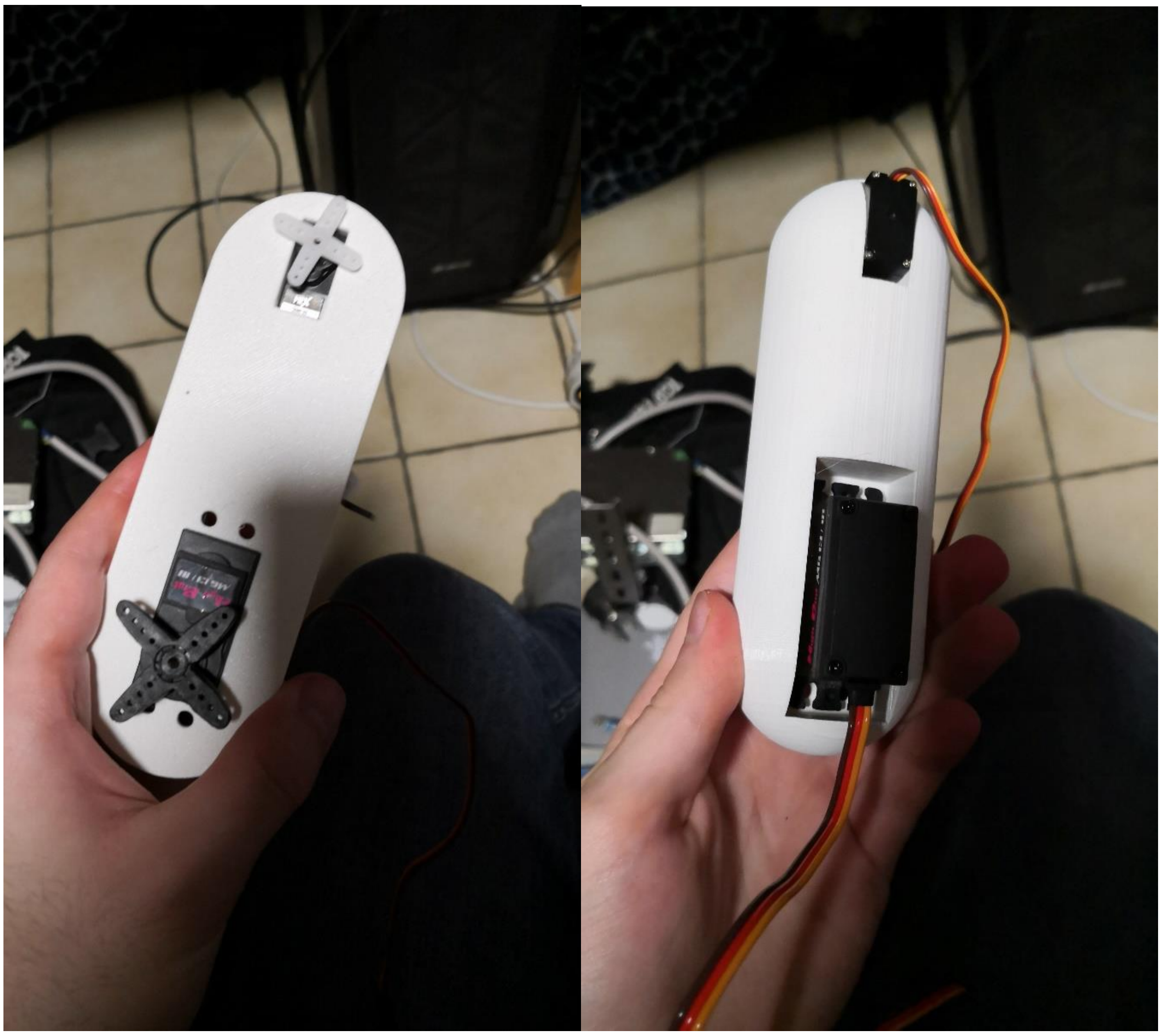
On the left you can see the notches for the servo motors with the screw holes to secure them in place

On the right you can see the underside of the part, which is flat.



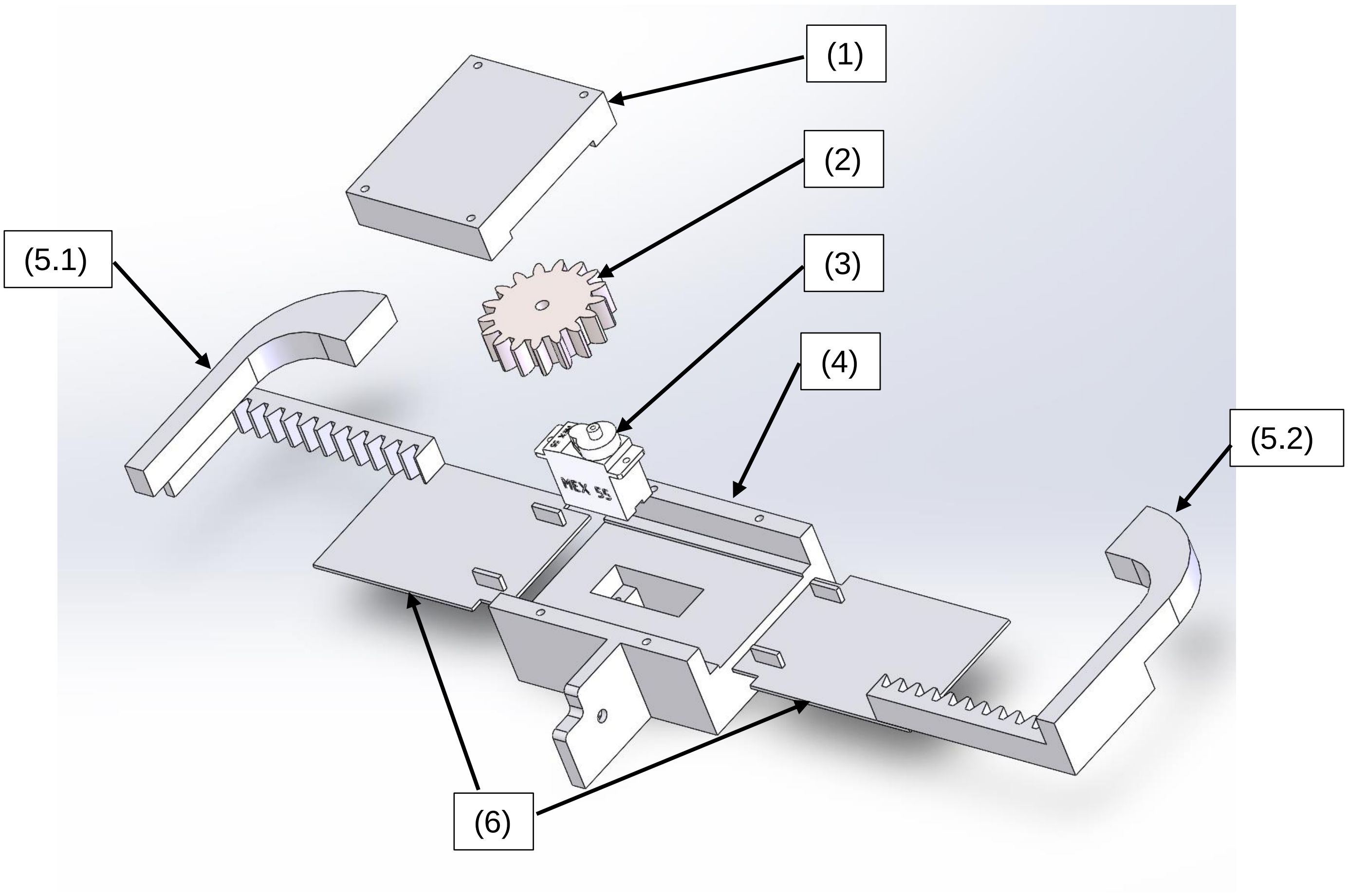


Here you can see the notches more clearly, how the servo motors are supposed to fit inside.



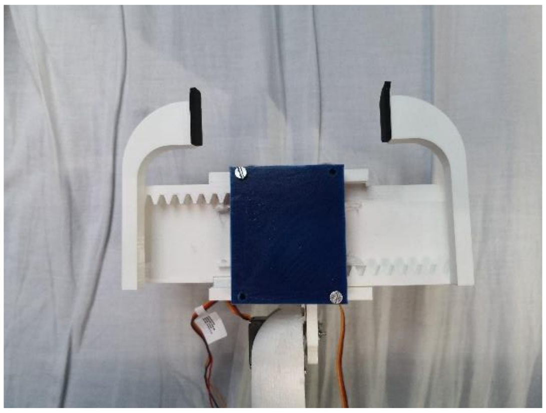
3.2.2.1.3 Hand assembly

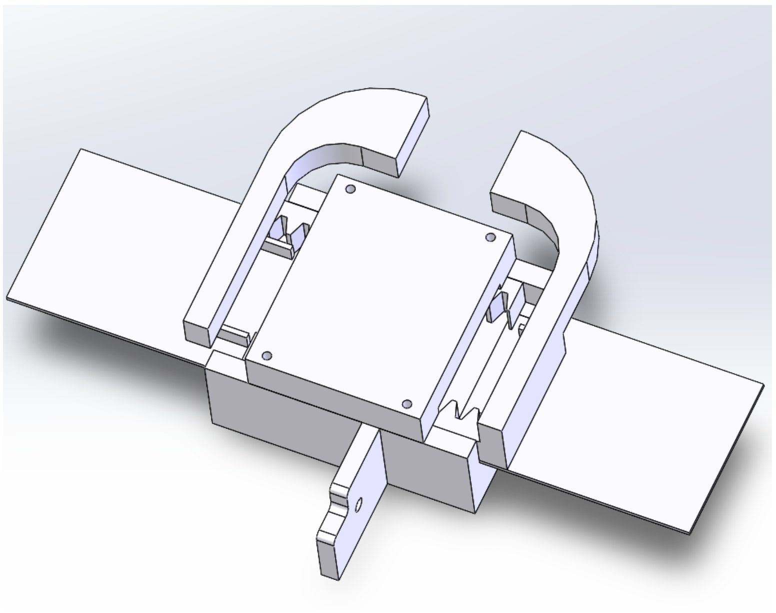
Then I wanted to design the hand/ the grabbing mechanism, which turned out to be a lot more difficult than I anticipated. I had to redo this assembly a few times because most of my ideas have proven to be unreliable, to complicated or just very badly designed. However, I'm proud of the final design of this assembly because this mechanism works very well.



(1): The lid that holds everything in place
(2): 17 tooth module 2 gear
(3): Servo Motor: Modster MEX 55
(4): Hand housing
(5): 5.1 is the left finger with the teeth for the gear on the top and 5.2 is the right finger with the teeth on the bottom
(6): Support plates for the fingers

Both fingers are being moved by the gear in the middle, which is driven by the servo motor. The supports have been added to the design after the first prototype, because it proved to be unstable.
I will go into further detail about every part.
The whole assembly put together looks like this in the final product:

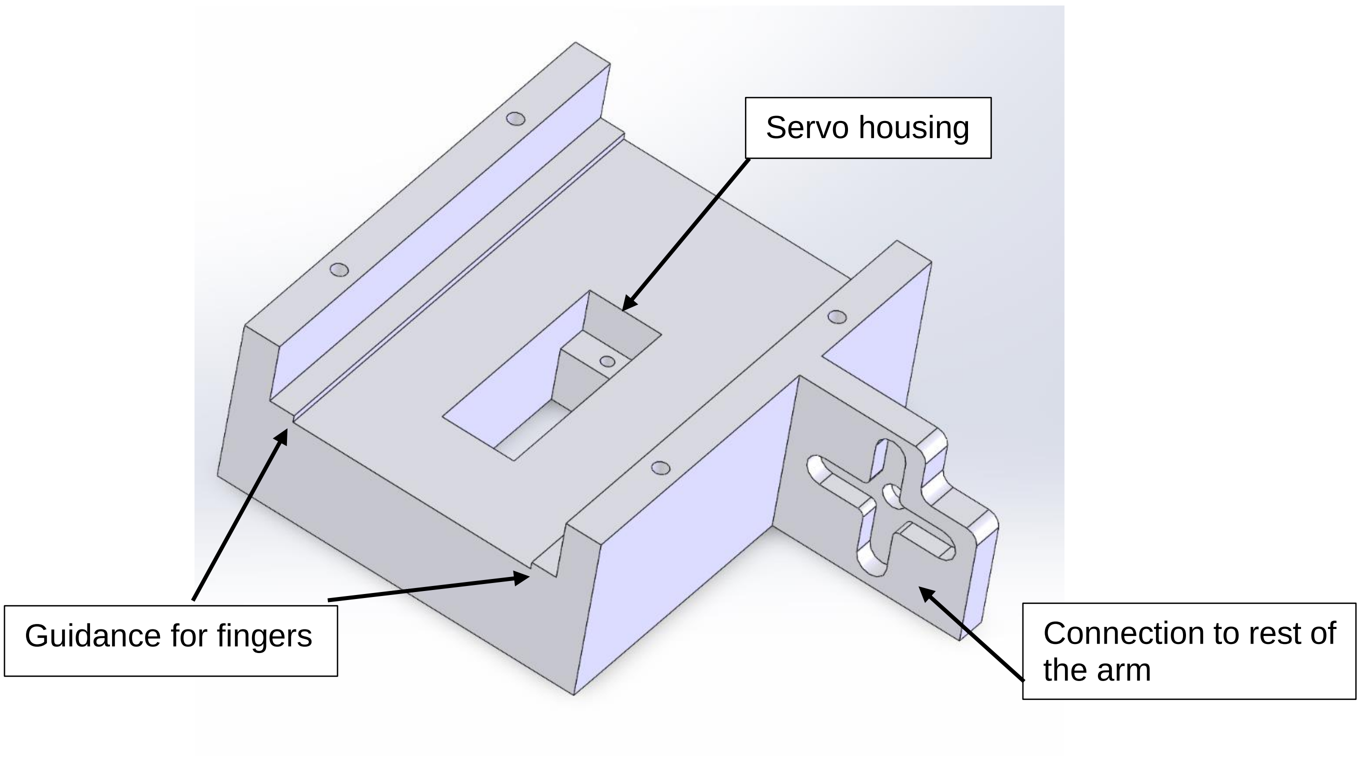


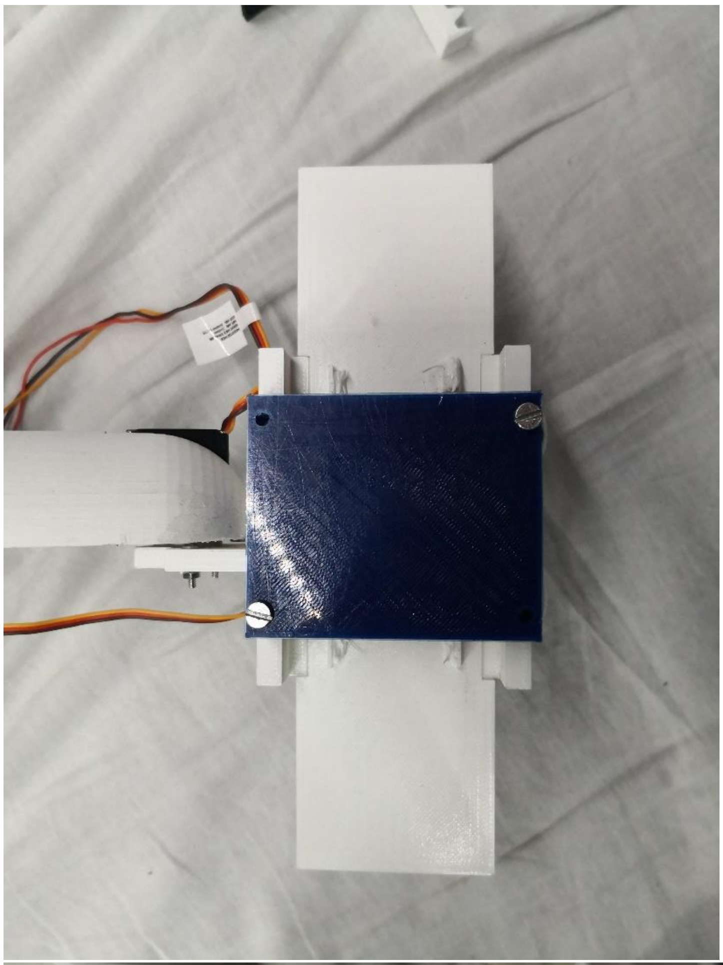


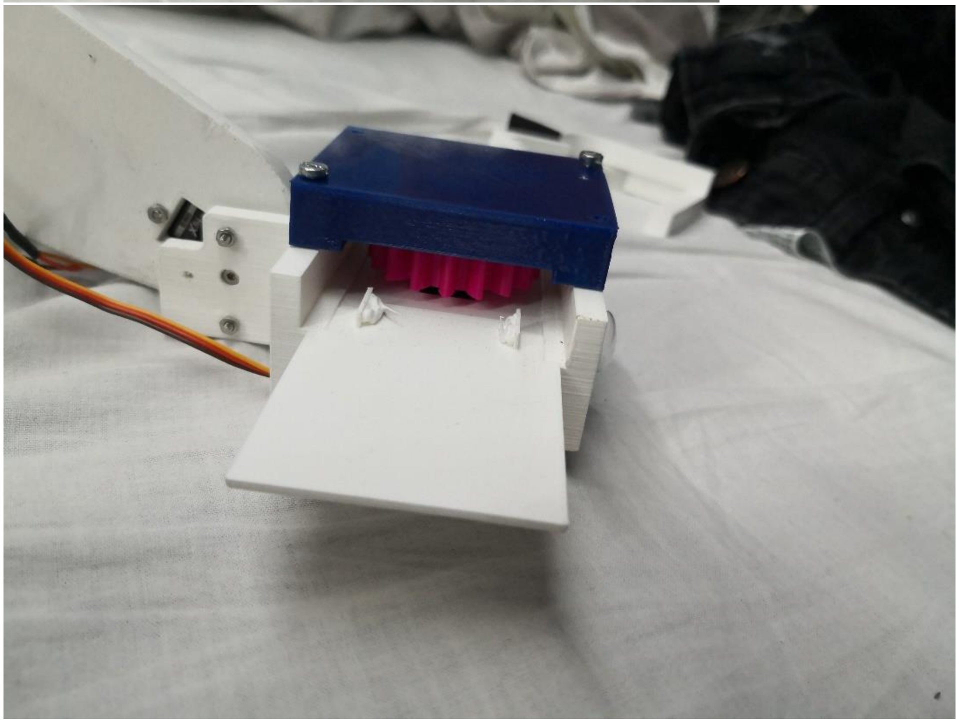
Here you can see the hand fully assembled

3.2.2.1.3.1 Hand housing

The hand housing is designed to house the servo motor, have a connection to connect to the rest of the arm, to hold all the parts and guide the fingers.





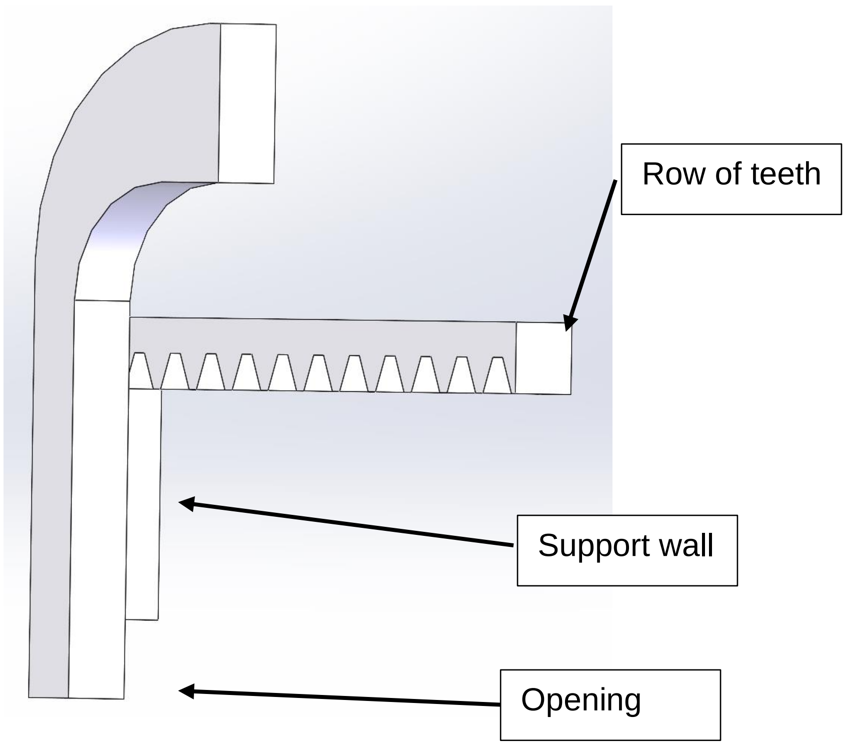


3.2.2.1.3.2 The fingers

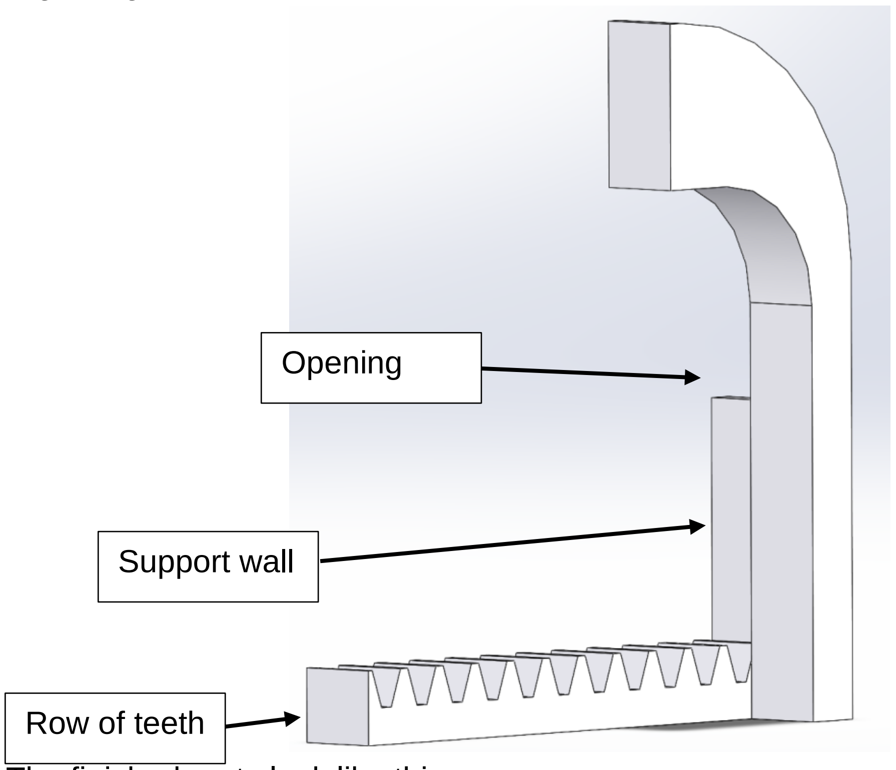
The fingers have a row of teeth to interact with the gear in the middle and they have a "wall to further support them when sliding on the support.

Both fingers have their row of teeth on the opposite sides, so that they don't interfere with each other. Both fingers could potentially collide with the other one's wall, so they have openings to prevent that.

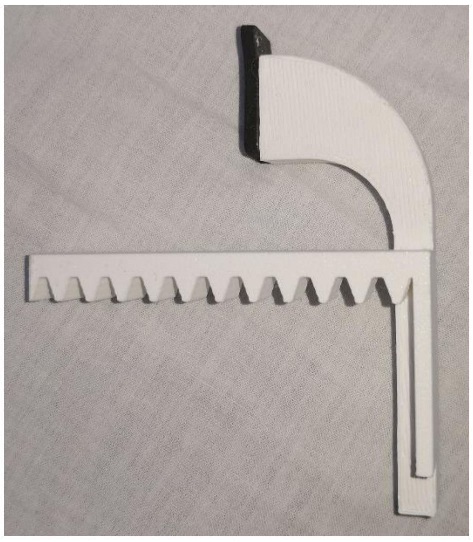
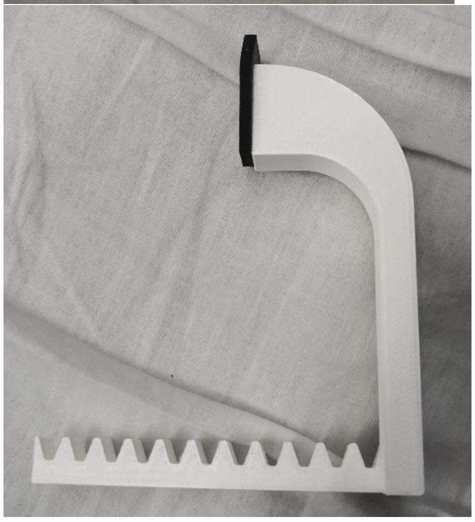
Left finger:



Right finger:

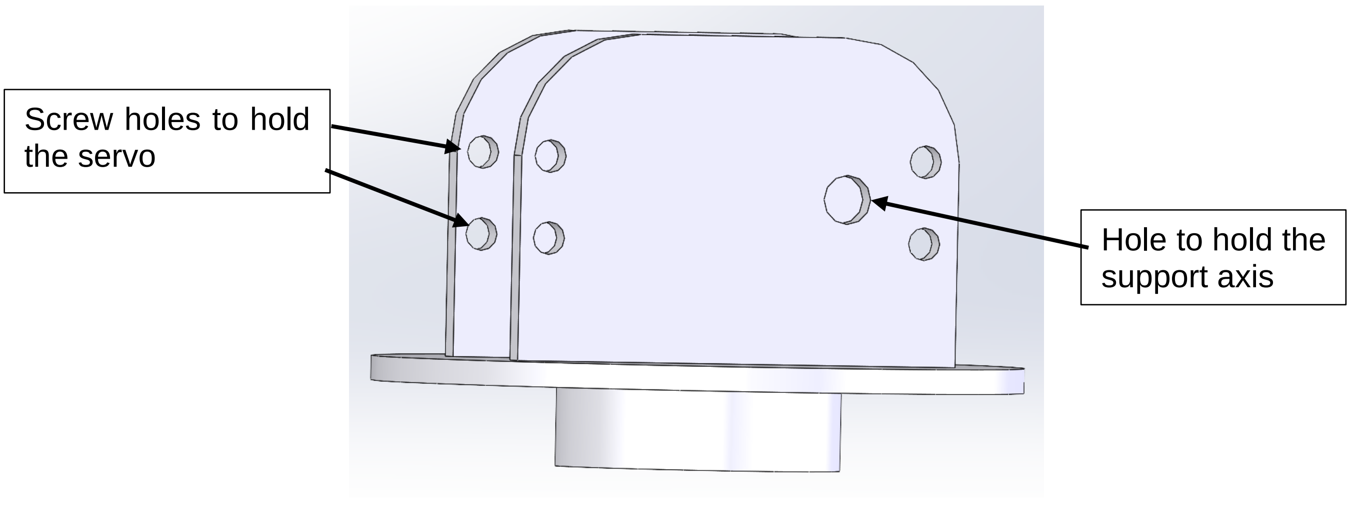


The finished parts look like this:

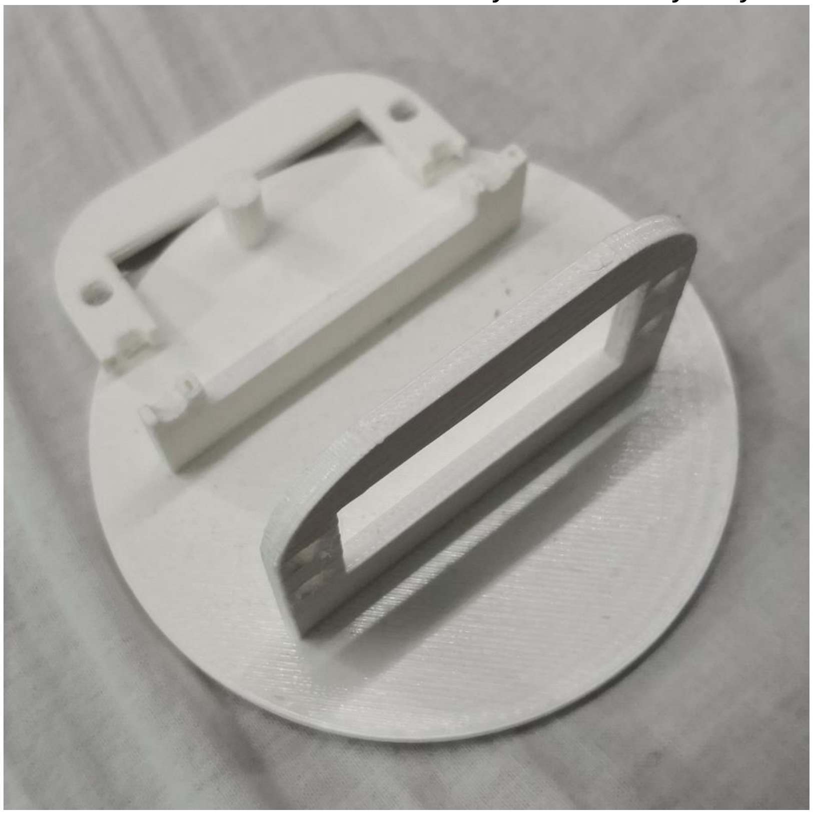



3.2.2.1.4 Base

I designed the first base part to be compatible with my idea of the steel base plate. It should be turned by a stepper motor and hold the base servo motor and the rest of the arm.



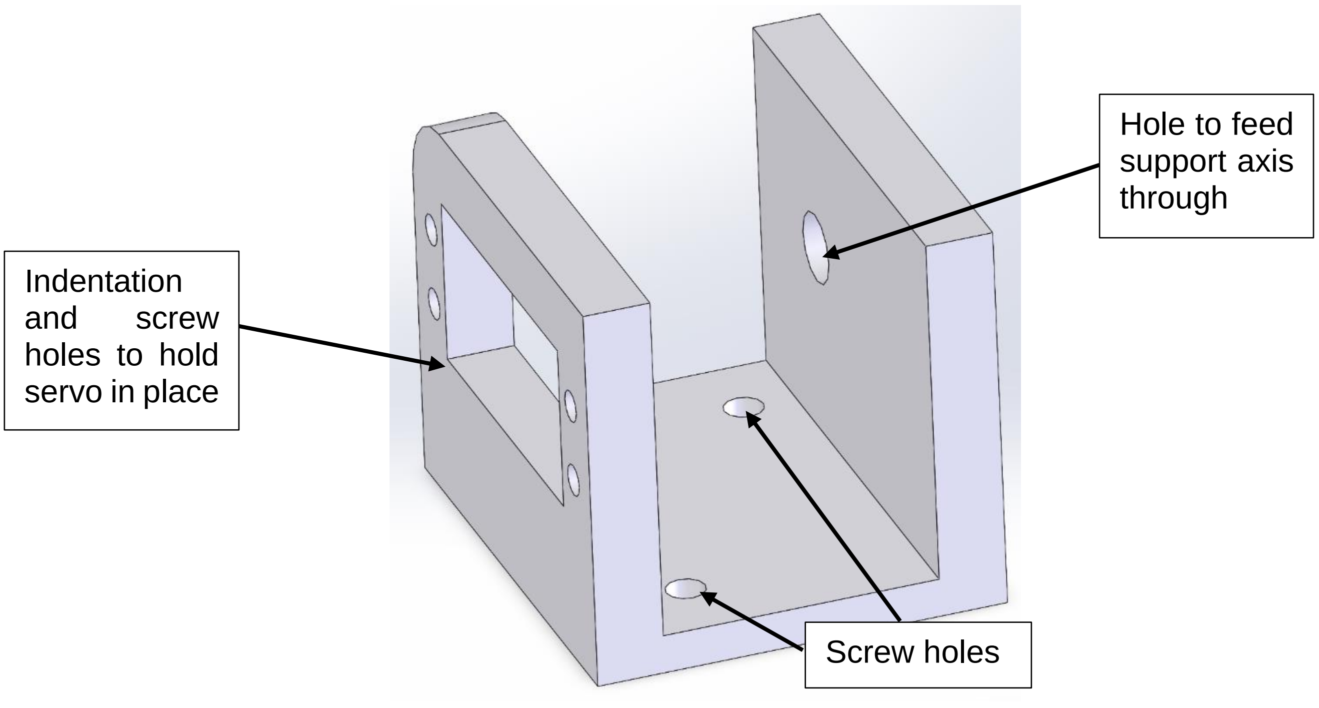
This idea turned out to break nearly immediately as you can see here:

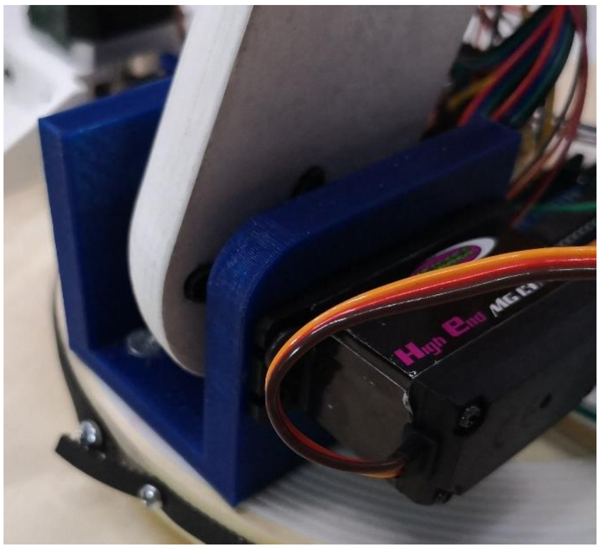


3.2.2.1.5 New Base

My initial plan of using a steel sheet as a base have changed over time and now, I'm using a turntable that you can buy. This changes how I attach the base and how I turn the whole thing.

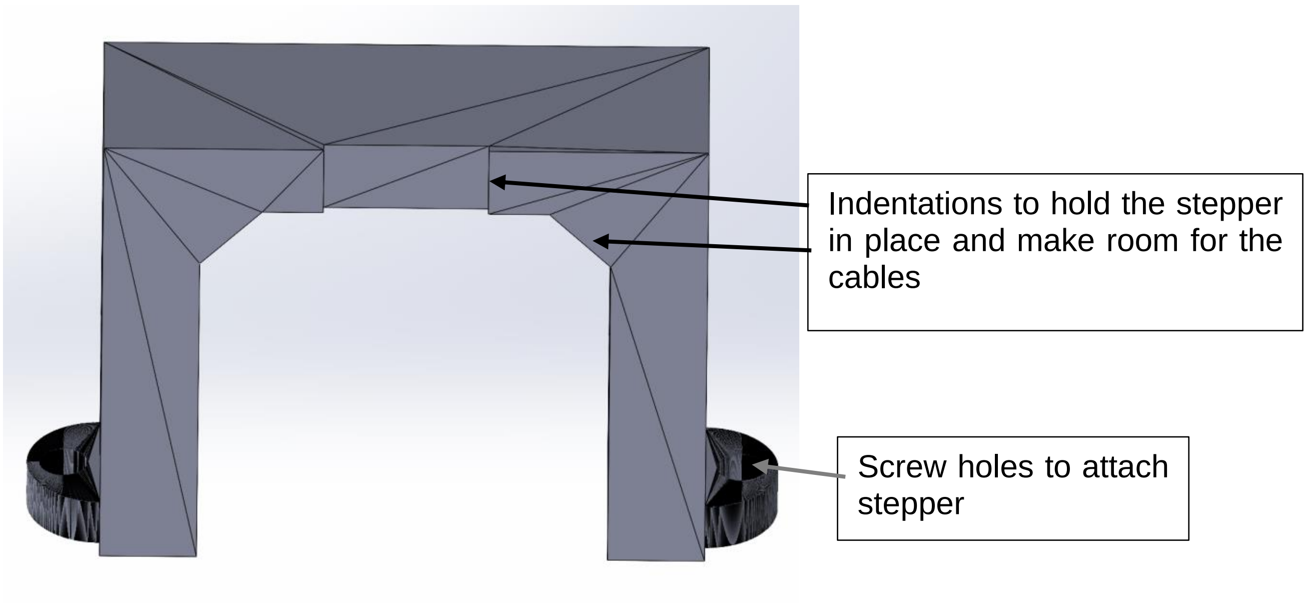
The new base is simply screwed in place on the small part of the turntable and holds the base servo and the rest of the arm.

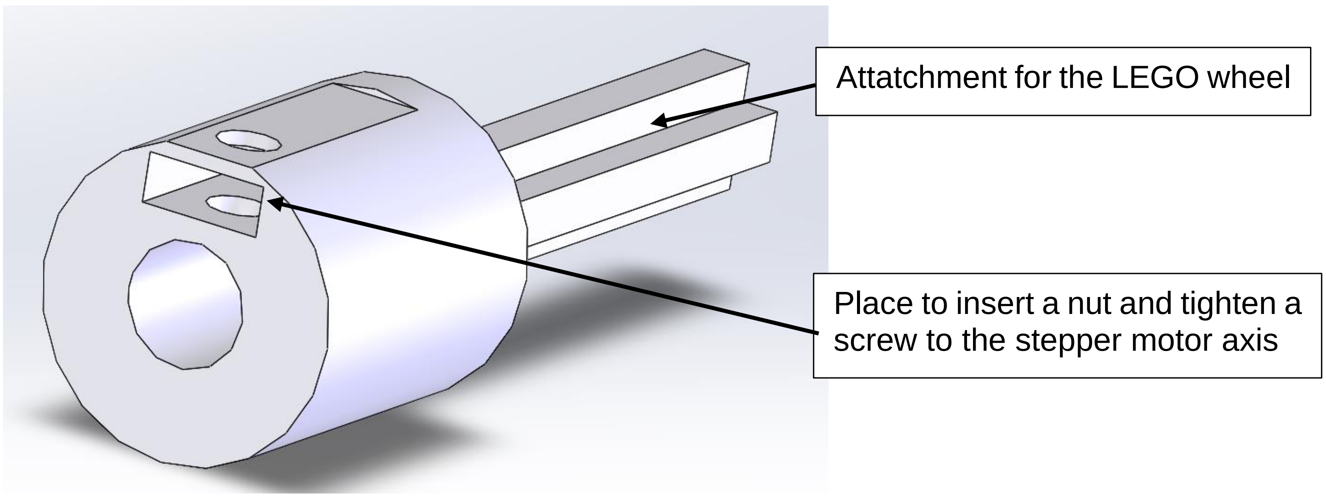

Hole to feed support axis through
Indentation and screw holes to hold servo in place
Screw holes



3.2.2.1.6 Old Stepper motor holder and drivetrain

My first idea to turn my turntable was to attach the stepper motor to the turning part of the turntable and attach a LEGO wheel to it. I thought this would turn the arm reliably and precise enough to fulfil this assignment. This wasn't the case, here you can see my first prototype to attach the stepper motor to the table and attach a LEGO extension for the wheel.


Indentations to hold the stepper in place and make room for the cables
Screw holes to attach stepper

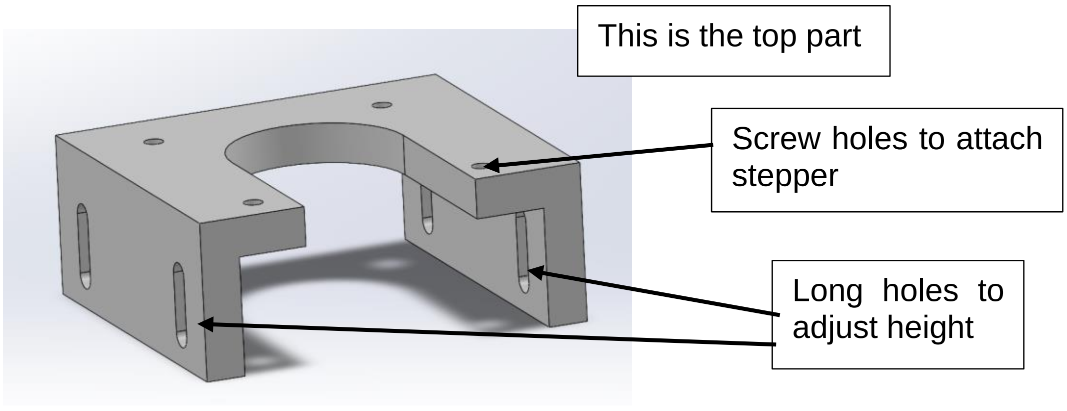

Attatchment for the LEGO wheel
Place to insert a nut and tighten a screw to the stepper motor axis

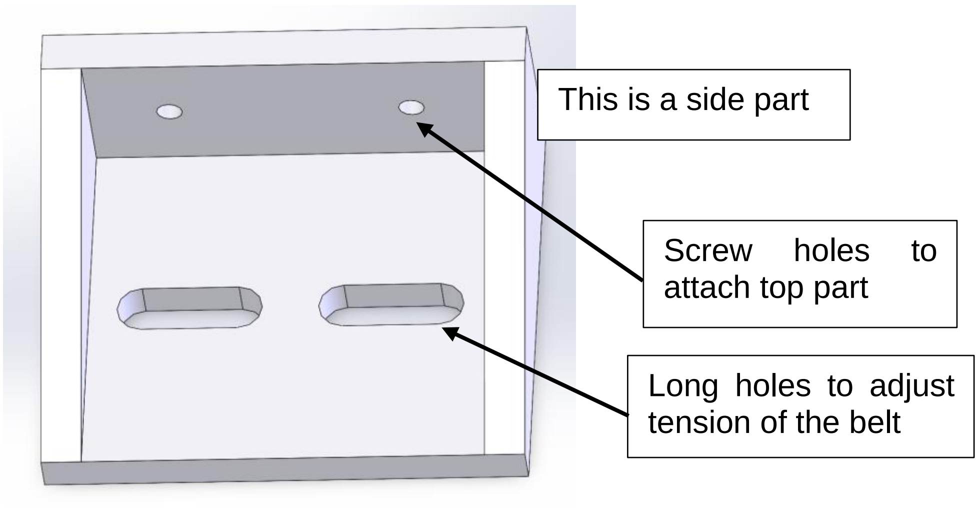
3.2.2.1.7 New Stepper motor holder

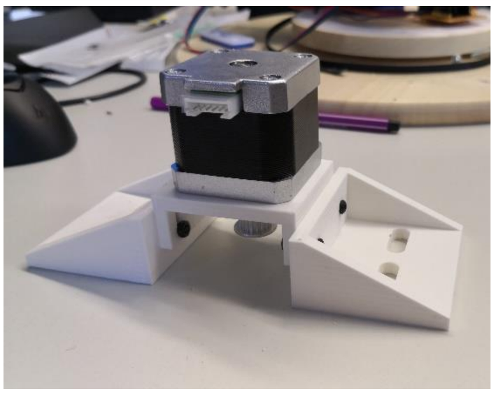
I had a few ideas to turn the turntable with the stepper motor, one of which was by attaching the stepper motor to the same part as the robotic arm and attaching a big LEGO wheel which turned everything by friction. I printed and tested a prototype, but it turned out to be unreliable and unprecise.

That's why I changed the design completely and attached the stepper motor to the stationary part of the turntable and turn/drive the turntable with a belt. The new stepper holder had to be adjustable in height and adjustable to tighten the belt to ensure reliability.

The design is split into three parts, the side parts, which allow for the adjustment of the tightness of the belt and the top part, which holds the stepper and allows for height adjustment.

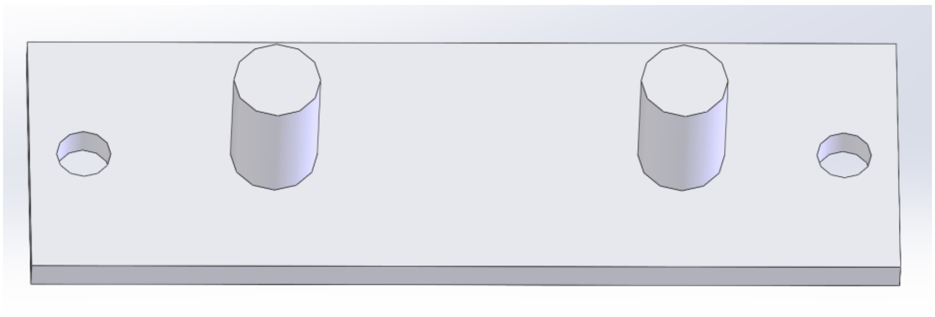



This is a side part
Screw holes to attach top part
Long holes to adjust tension of the belt



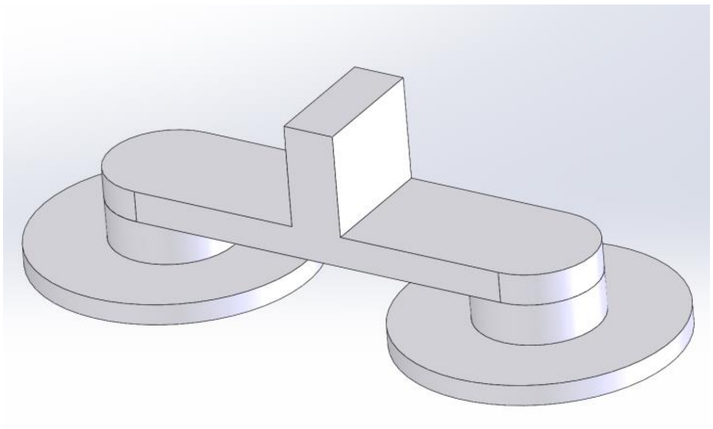
3.2.2.1.8 Ball bearing guide

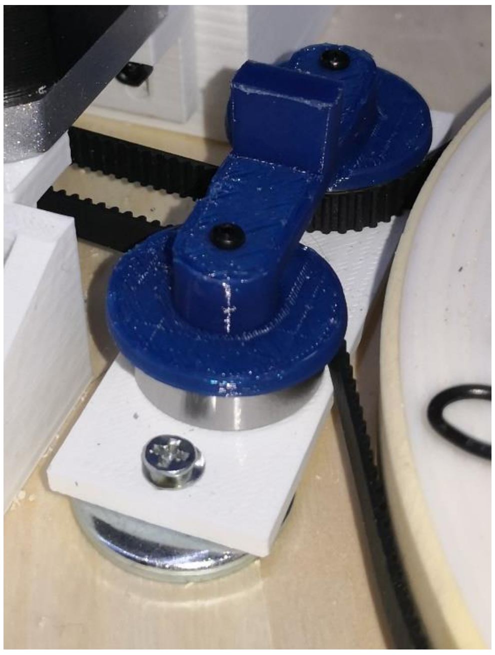
I had to design a guide for the belt, because it was rubbing against the side parts of the stepper motor holder. It holds 2 ball bearings that securely guide and further tension the belt.



3.2.2.1.9 Cap to find zero position

I have also designed a cap for the ball bearing guide so that the belt can’t slide off the bearings. I interconnected them and put a wall in the middle of it, so that the limit switches can push against it to find the zero position of the robot arm.





3.2.2.2 3D Printing

3D printing was completely new to me, and I was very uncertain to print anything on the school's 3D printers. However, after I printed a few prototype parts with help I felt more at ease with 3D printing and know now a bit more about what certain settings do and how to level the bed.

Most of my prints were successful or at least acceptable, but some failed, which caused some delay in the overall progress of my project.

3.2.3 Assembly

The assembling process brought up a few problems which I will explain in these following paragraphs.

3.2.3.1 Problems

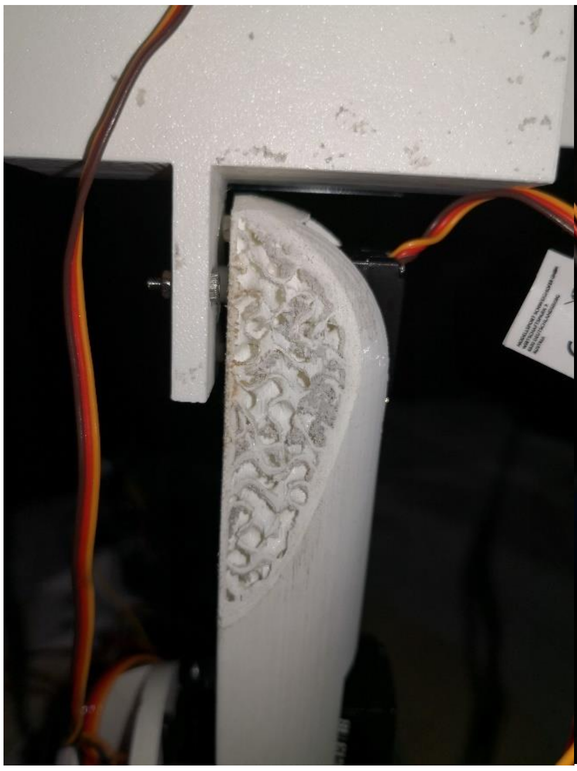
I encountered a few problems while trying to assemble the robot arm, especially with the crosses I designed into the parts to hold a cross from the servo motors. My measurements must have been wrong by a bit, because after the print was done and I wanted to put the crosses inside the notch, they wouldn't fit. This was especially frustrating with the *3.2.2.1.1. Part 1* because it has 2 of these crosses that wouldn't fit. To still be able to fit the crosses, I grinded the notch away until the cross would fit snuggly and then I glued the cross to the part, so that it couldn't move anymore.

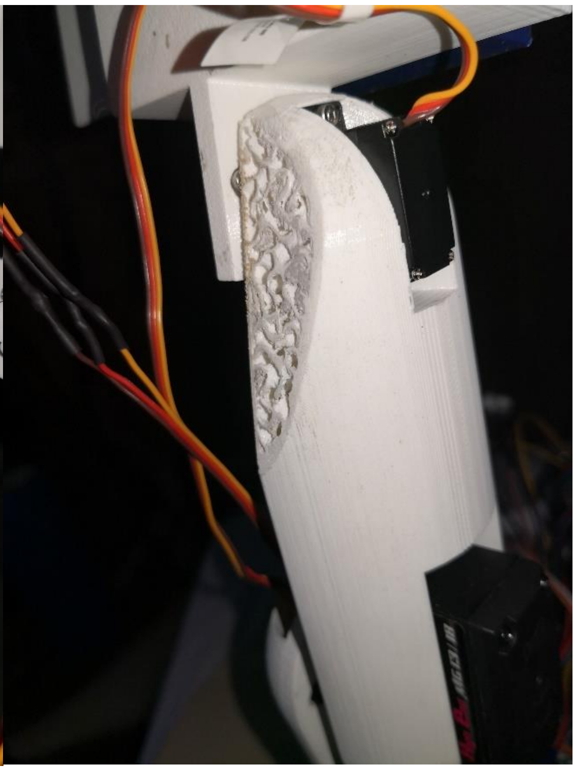
Another problem while assembling was that after I had printed the first base and tried to put it together with the servo motor and the *Part 1* it just broke because the tolerances were to tight and I had applied to much force on the thin material. Later on my idea for the base changed and I fixed these previously had issues with the design.

I came across another problem while trying to mount the LEGO wheel onto the stepper motor in its first mounting bracket. The problem was that the axile of the wheel was a bit higher that the one from the stepper motor, so the stepper motor was always at an angle and the wheel forced the axile of the stepper up, which is never good.
I later changed this drivetrain completely as previously mentioned.

When putting together the new base and the rest of the arm, I wanted to insert the support axile and I noticed that the axile was misaligned. After some inspection of the parts, I became aware that the base servo has elevations on the mounting brackets, and I forgot to make indentations for those in the base. I grinded away some material to allow the servo to sit straight in the part. This greatly fixed the issue, after I drilled the holes a bit wider open and fit a new axile through it fits perfectly.

During testing I noticed that the angle at which the grabbing mechanism could interact with the cube wasn’t 90° to the second part, as I planned. I had to buy a new servo motor for the “wrist” because I apparently forced it against this wall for too long, which I didn’t notice until it was too late. I resolved this problem by cutting the second part off to a certain point, so the grabbing mechanism could be turned around 90° to the second part.





3.2.3.2 How to build it

Firstly, I took a turntable and flipped it upside down, so that the part that normally sits on the table is on top. Then I attached the base part to the turning part of the turntable by predrilling holes in the wood and then putting screws in through screw holes in the part.

Then I attached the base servo motor to the base and added the first part of the arm to the base servo and secured it with a screw.

The next part needs to be prepared by inserting both the big and small servo motor into their foreseen mould and securing them with screws in their respective positions.

Then the big servo from the second part can be attached to the first part by putting the cross on the servo motor and securing it again with a screw. The cables of the 2 added servo motors can conveniently be fed through the hollow first part.

Coming to the hand assembly, I took a small servo motor and put it in the foreseen place in the hand part and attached the gear to it and secured it with a screw. Then I glued the support parts onto the hand part because they were added after the first testing. After that, I screwed on the lid and put the servo into its fully open position, then held the fingers in a way that they would interact with the gear, if it would turn again. Then I made the servo go to its closed position to fully insert the fingers into their supposed positions. The whole assembly could then be attached to the small servo on the second part with the inserted cross and secured by a screw. The cables from the servo motor can be fed through the hollow first part.

Then I put together the stepper holder, inserted the stepper motor and attached a belt to the turning part of the turntable. I proceeded to predrill holes for the stepper holder and the ball bearing guide for the belt. I inserted the ball bearings on their supposed position and screwed on the cap to secure everything in place. The guide is being screwed in position, the belt fed through their path and the stepper holder is put loosely in place to be able to attach the belt and tighten it.

The buttons should be attached to the base in a way that they are activated by turning the arm to a certain degree and the buttons touch the wall on the ball bearing guide cap. This allows the robot arm to find its 0 position.

After all these parts are assembled, the microcontroller and the stepper motor driver can be screwed to the turning part of the table and be connected to the motors and buttons as the electronic circuit schema is showing it.

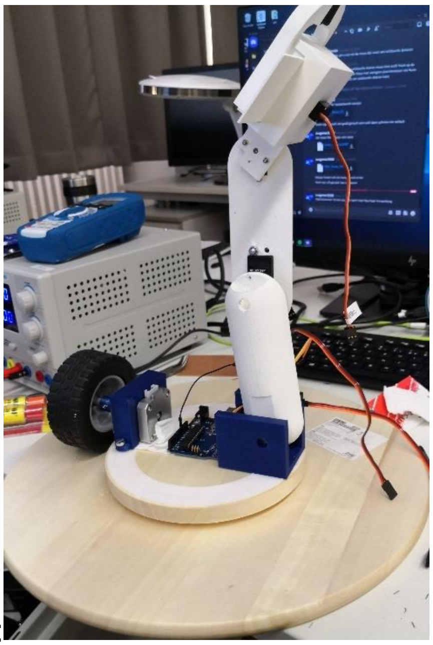
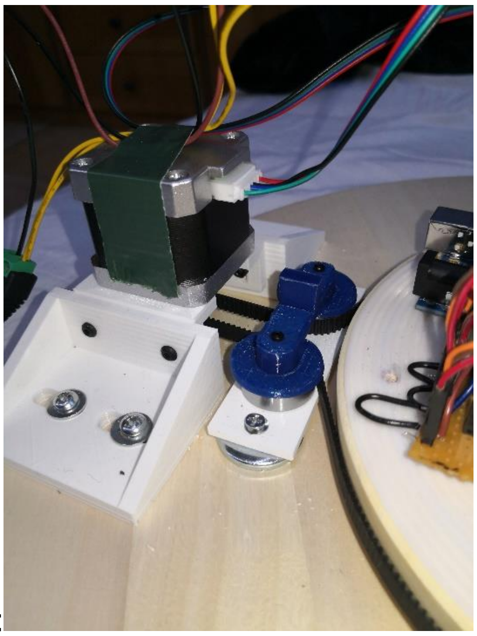
3.2.4 *Coding*

The coding part was at first a little exhausting at first, because I had no idea how to turn the stepper motor and how to control it properly. However, when I understood some examples of code snippets from other people, I could recreate my ideas to turn the stepper motor the way I wanted. The servo motors are easily controllable, and their code is also very easy. I will explain everything in detail here.

3.2.5 *Testing*

The testing process is one of the most important parts of the whole building and programming. This is the part where you can see your ideas succeed or terribly fail.

As I previously stated, I had tested my idea with the LEGO wheel as a drive train for the robot arm but after some testing it proved to be unreliable and not precise enough to power my robotic arm. This was due to multiple reasons. One being that the wheel wasn't very securely attached, another that it was misaligned. However, the main problem was that this system relied on friction, which wasn't always present, therefor it sometimes skipped steps and had no way of controlling if and how many. This is the reason I switched to the belt drive, which turned out to be very reliable and precise in its movements.

Before:  After: 

As mentioned previously, I also had issues with the grabbing mechanism. This was easily fixed by cutting the part.

The testing also allowed me to control the angles of the servo motors correctly and tune them in a way that the arm can now pick up and put down a cube safely and reliably.

3.3 Detailed List of parts

3.3.1 Motors

3.3.1.1 Servo Motors

Servo motors are a type of motor that can only rotate around 180° +-, at least the ones we're using. The movement of these motors can be programmed on a microcontroller where you define the angle the motor is supposed to go to. For example, if the motor is currently at the angle 43° you can tell it in the code to go to the angle 120° and it turns immediately to that angle.

3.3.1.1.1 Jamara MG-13-18

I have used 2 of these servo motors because of their metal gears, which ensure strength and reliability and because of their strength, which plays a big factor in my design. These are the big base and elbow motors.

3.3.1.1.2 Modster MEX-55

I have also used 2 of these servo motors for the same reasons mentioned in the previous point. These are the small motors for the wrist and grabbing mechanism.

3.3.1.2 Stepper Motors

Stepper motors are motors which are controlled by "making steps" in one or another direction. They can also turn an indefinite number of revolutions, unlike the servo motors.

3.3.1.2.1 Nema 17

I have used one stepper motor of this kind to turn my turntable.

3.3.2 Microcontrollers and boards

3.3.2.1 Arduino uno

I used one Arduino Uno to control every movement of the robotic arm.

3.3.2.2 A4988

This is a stepper motor driver and I used it to control/drive the Nema 17 Stepper motor that I used

3.3.3 Additional parts

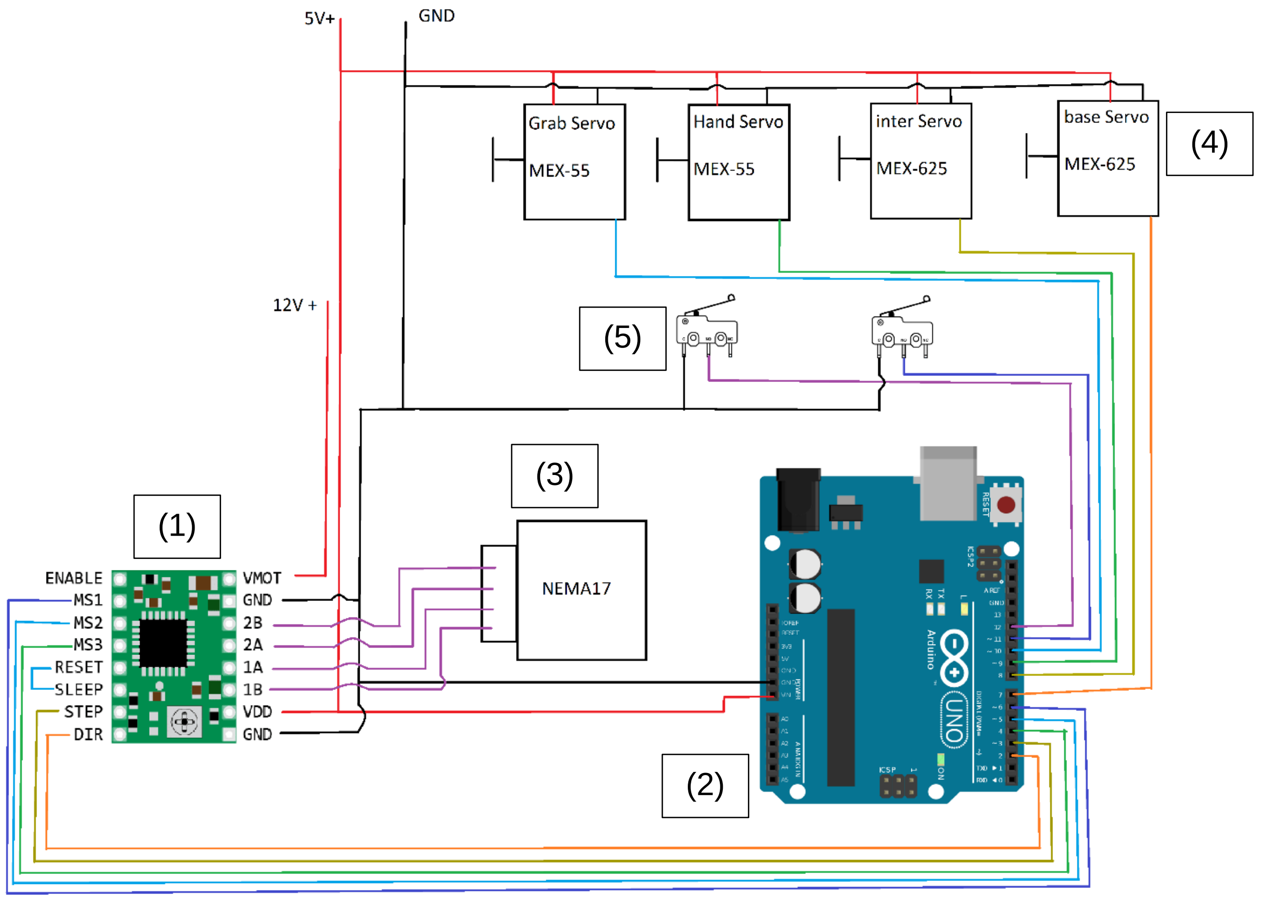
3.3.3.1 Limit/leaf switch

I used 2 limit switches to be able to find the zero position of my robot arm.

3.3.3.2 Turntable

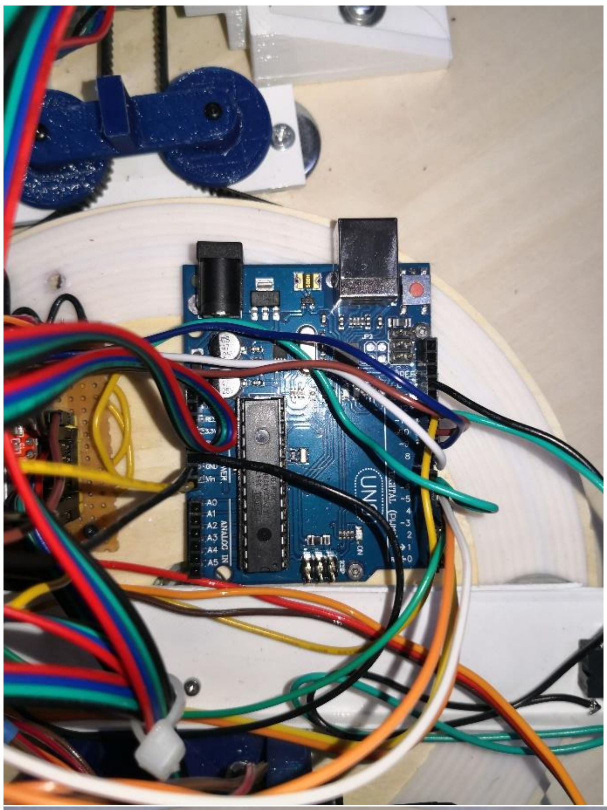
I used a turntable as a base and turning point of my robot arm.

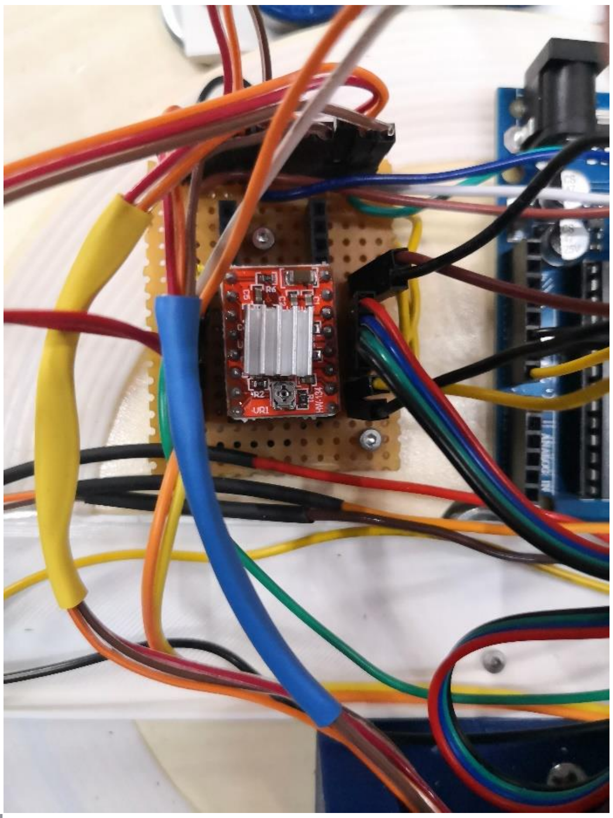
3.4 Electronic circuit schema



(1): stepper motor driver A4988
(2): Arduino uno
(3): Stepper motor
(4): Servo motors
(5): limit/leaf switches

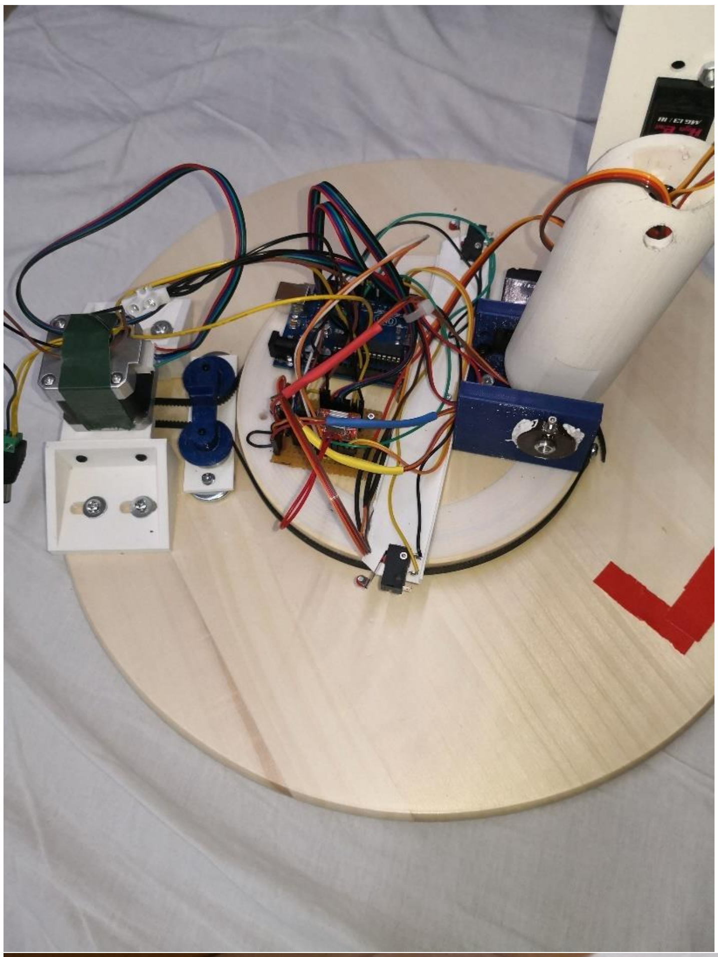
The stepper motor driver's MSx pins are to be able to control the stepper in different precise levels. For example: if none of the pins are powered, the motor takes whole steps and if all of those pins are powered, the motor only makes 1/16 steps.
The STEP and DIR pins are connected to the Arduino uno, these allow the Arduino to control the steps and direction of the movement from the stepper motor.
The driver is also connected to 5V and 12V DC. The ground connection is shared among all devices. The 5V connection is also there to power the Arduino and the servo motors.
The servo motors are all connected to the Arduino with their signal pins. Their 5V and ground connections are shared.
The switches are connected to the Arduino and ground, they are defined as internal pullup buttons on the Arduino.

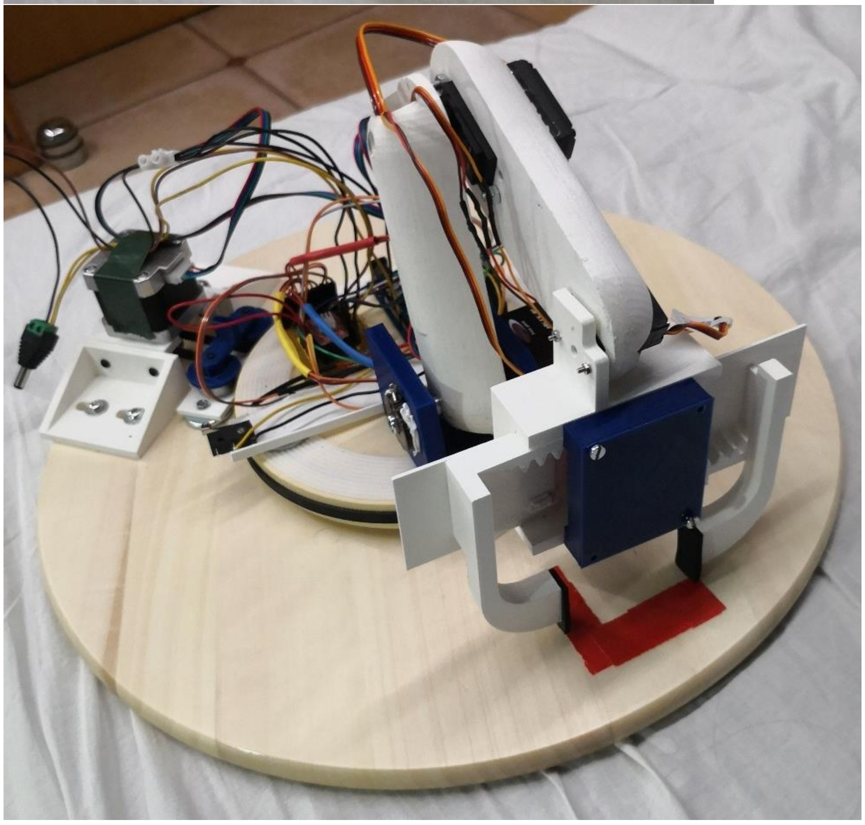


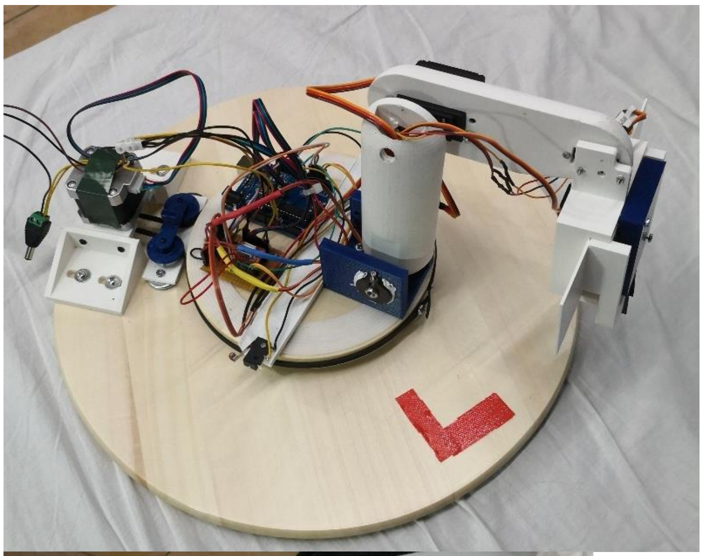


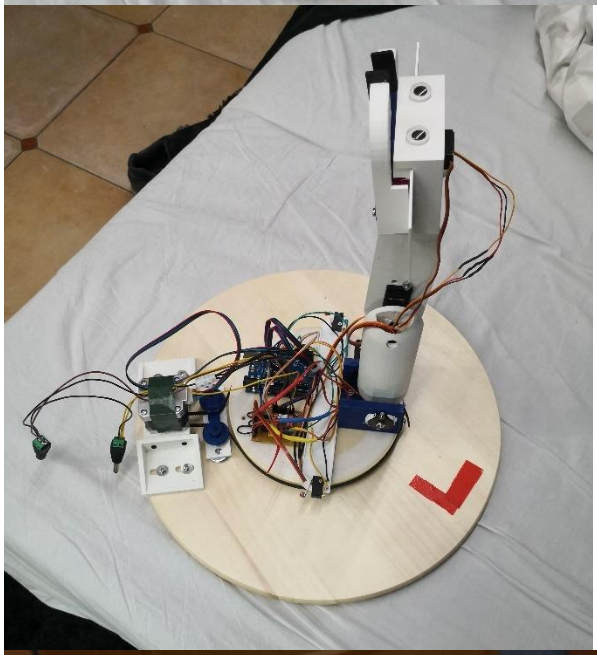
MG 13 / 18
Just play
JAMARA

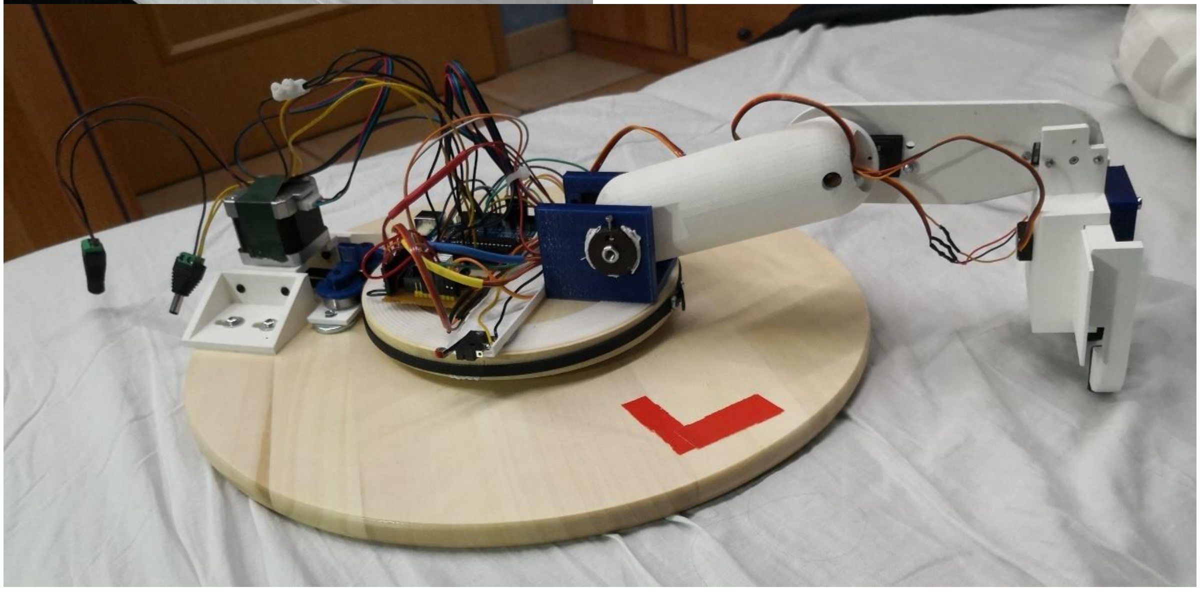
3.5 Pictures of the final product









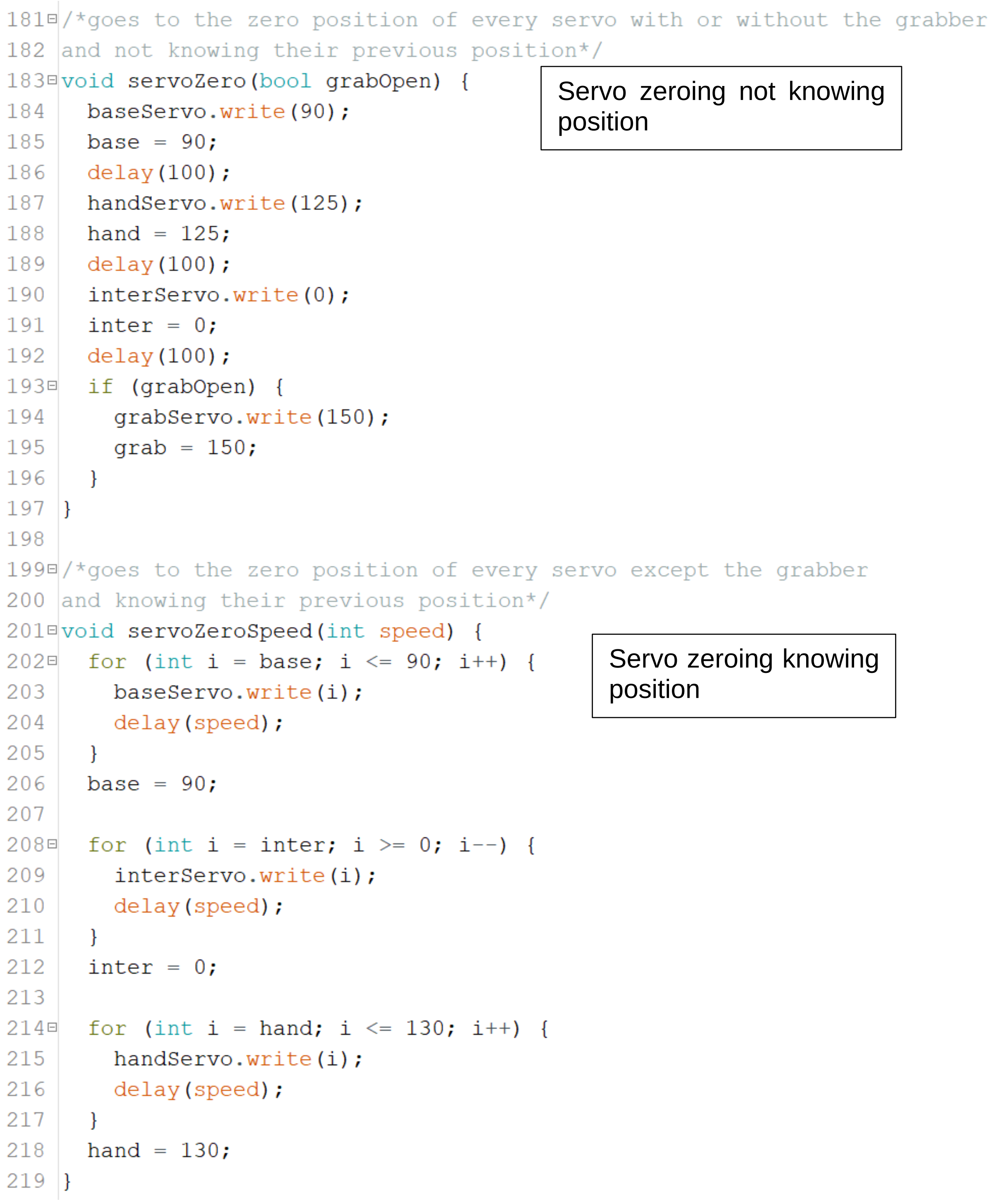


3.6 Description of firmware

My setup function only declares all the pins and moves the servo motors to their zero position that the robot arm is standing upright.

```
void setup()
{
  // Declare pins as Outputs
  pinMode(stepPin, OUTPUT);
  pinMode(dirPin, OUTPUT);

  pinMode(MS1, OUTPUT);
  pinMode(MS2, OUTPUT);
  pinMode(MS3, OUTPUT);

  //declare buttons as internal pullup
  pinMode(LEFT, INPUT_PULLUP);
  pinMode(RIGHT, INPUT_PULLUP);

  //attach the servos to their according pins
  baseServo.attach(baseServoPin);
  grabServo.attach(grabServoPin);
  handServo.attach(handServoPin);
  interServo.attach(interServoPin);

  //go to the servos zero position without knowing their previous position
  servoZero(true);

  //set the accuracy to maximum: 1/16 steps
  digitalWrite(MS1, HIGH);
  digitalWrite(MS2, HIGH);
  digitalWrite(MS3, HIGH);

  //begin serial monitor for testing
  Serial.begin(9600);
}
```

I have 2 functions to go to the 0 position of the servos, one is when it knows the previous position of the servos and the other one is at the start up, when it doesn't know all the positions.



```
/*goes to the zero position of every servo with or without the grabber
and not knowing their previous position*/
void servoZero(bool grabOpen) {
  baseServo.write(90);
  base = 90;
  delay(100);
  handServo.write(125);
  hand = 125;
  delay(100);
  interServo.write(0);
  inter = 0;
  delay(100);
  if (grabOpen) {
    grabServo.write(150);
    grab = 150;
  }
}

/*goes to the zero position of every servo except the grabber
and knowing their previous position*/
void servoZeroSpeed(int speed) {
  for (int i = base; i <= 90; i++) {
    baseServo.write(i);
    delay(speed);
  }
  base = 90;

  for (int i = inter; i >= 0; i--) {
    interServo.write(i);
    delay(speed);
  }
  inter = 0;

  for (int i = hand; i <= 130; i++) {
    handServo.write(i);
    delay(speed);
  }
  hand = 130;
}
```

Servo zeroing not knowing position

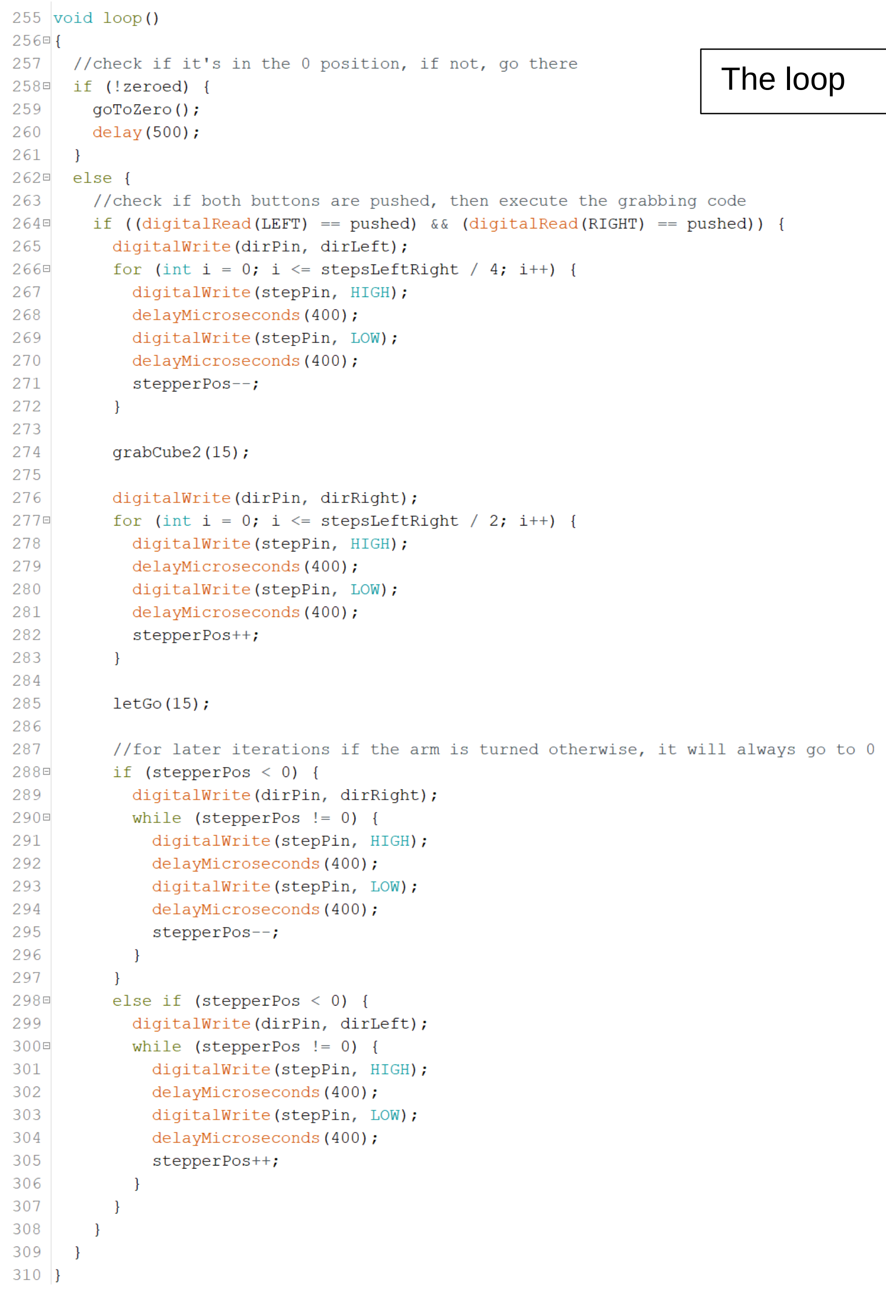
Servo zeroing knowing position

My grabbing method is simple, but unfortunately with hardcoded positions. First, I check if both buttons are pressed, if that’s the case, the stepper motor will turn ¼ turn clockwise and then execute the “grabCube2(int speed)” method. This method turns every servo motor to the point where it can grab the cube from the marked spot on the turntable and grab it.

The function is called that way, because I also have a excluded function in my code “grabCube()”, which is functional, but the reliability is not as easily verifiable as in my second iteration.

Then the loop continues and turns the stepper motor again ½ turn counter clockwise and activates the “letGo()” function which safely puts down and lets go of the cube and at the end goes to all the servos zero position. Lastly, the stepper turns the arm to the zero position again.

The code is further commented as you can see in the ArmProject.ino inside the added .ZIP file.



```
void loop()
{
  //check if it's in the 0 position, if not, go there
  if (!zeroed) {
    goToZero();
    delay(500);
  }
  else {
    //check if both buttons are pushed, then execute the grabbing code
    if ((digitalRead(LEFT) == pushed) && (digitalRead(RIGHT) == pushed)) {
      digitalWrite(dirPin, dirLeft);
      for (int i = 0; i <= stepsLeftRight / 4; i++) {
        digitalWrite(stepPin, HIGH);
        delayMicroseconds(400);
        digitalWrite(stepPin, LOW);
        delayMicroseconds(400);
        stepperPos--;
      }

      grabCube2(15);

      digitalWrite(dirPin, dirRight);
      for (int i = 0; i <= stepsLeftRight / 2; i++) {
        digitalWrite(stepPin, HIGH);
        delayMicroseconds(400);
        digitalWrite(stepPin, LOW);
        delayMicroseconds(400);
        stepperPos++;
      }

      letGo(15);

      //for later iterations if the arm is turned otherwise, it will always go to 0
      if (stepperPos < 0) {
        digitalWrite(dirPin, dirRight);
        while (stepperPos != 0) {
          digitalWrite(stepPin, HIGH);
          delayMicroseconds(400);
          digitalWrite(stepPin, LOW);
          delayMicroseconds(400);
          stepperPos--;
        }
      }
      else if (stepperPos < 0) {
        digitalWrite(dirPin, dirLeft);
        while (stepperPos != 0) {
          digitalWrite(stepPin, HIGH);
          delayMicroseconds(400);
          digitalWrite(stepPin, LOW);
          delayMicroseconds(400);
          stepperPos++;
        }
      }
    }
  }
}
```

The loop

grabCube2 function

```
//grabs the cube from the marked position
void grabCube2(int speed) {

  for (int i = base; i <= 120; i++) {
    baseServo.write(i);
    delay(speed);
  }
  base = 120;
  for (int i = hand; i >= handMin; i--) {
    handServo.write(i);
    delay(speed);
  }
  hand = handMin;
  for (int i = inter; i <= 130; i++) {
    interServo.write(i);
    delay(speed);
  }
  inter = 130;

  for (int i = grab; i >= 50; i--) {
    grabServo.write(i);
    delay(speed);
  }
  grab = 50;

  servoZeroSpeed(speed);
}

```

LetGo function

```
//puts the cube down safely
void letGo(int speed) {
  for (int i = hand; i >= handMin; i--) {
    handServo.write(i);
    delay(speed);
  }
  hand = handMin;
  for (int i = inter; i <= 30; i++) {
    interServo.write(i);
    delay(speed);
  }
  inter = 30;
  for (int i = base; i >= 35; i--) {
    baseServo.write(i);
    delay(speed);
  }
  base = 35;
  for (int i = grab; i <= 150; i++) {
    grabServo.write(i);
    delay(speed);
  }
  grab = 150;

  servoZeroSpeed(speed);
}
```

4 Executive summary

My iteration of the robot arm can lift a cubic shape of 40mm side length and 25g of weight at least 30cm high and the arm can potentially move 180°.

It is powered by a power supply and needs 5V and 12V input, which power the 4 servo motors; which control the movement of the arm itself, the stepper motor; which makes the rotation of the arm possible, the Arduino; which controls everything and the stepper motor driver; which makes communication between the Arduino and the stepper possible.

My initial idea of the steel base was discarded over the course of time and replaced by a turntable, which also created some necessary design changes. This changement occurred at the second half of the project and induced quite some stress, because some of my designs were therefore unusable.

Furthermore, I had planned for another grabbing mechanism, which I tried to make work for a long time and in the end, I had to change to my final design to have a working prototype.

This project gave me new insights on how to 3D model and print and on myself, especially that my time management needs to be much better organized and structured. This allows me to be more prepared for future projects and endeavours.

5 Bibliography and list of sources

1. https://www.youtube.com/watch?v=_B3gWd3A_SI
2. Pololu - A4988 Stepper Motor Driver Carrier

6 Acknowledgements

Special thanks to Mr. Bernard for giving a course on 3D modelling and helping with issues and questions on the mechanical side of things.

Special thanks to Mr. Weiler for explaining everything from the electronical side and for explaining how the school’s 3D-Printers work.

Special thanks to my father who also helped me with mechanical advice for my designs.

Special thanks to Yves Fischer for helping me all around the project.

7 Declaration of autonomy

I certify herby that I have written the thesis and report independently without the help of other aids than those explicitly stated.

Name:	Charel Feil
Place and Date:	Bissen, Luxembourg 30.01.2022
Declaration:	I certify herby that I have written the thesis and report independently without the help of other aids than those explicitly stated.
Signature:	[illegible]