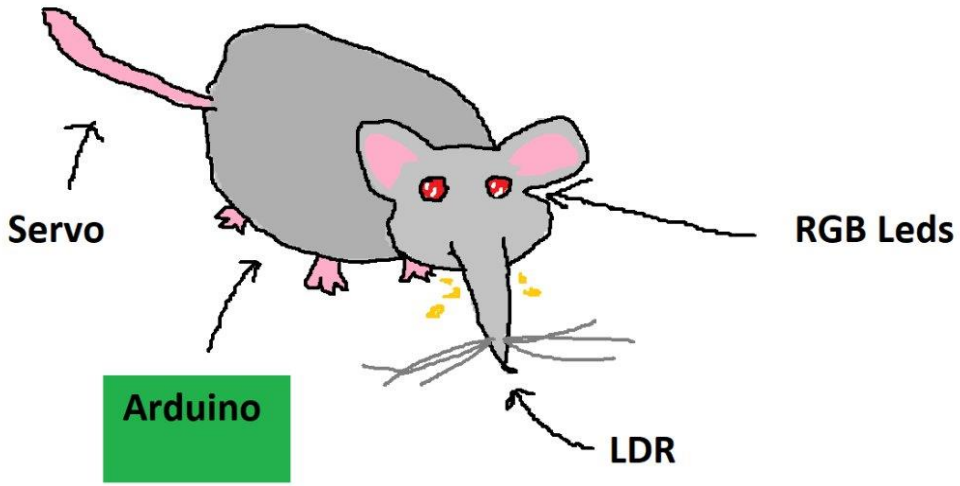
Physical Computing Project Idea



Relevant Links

1

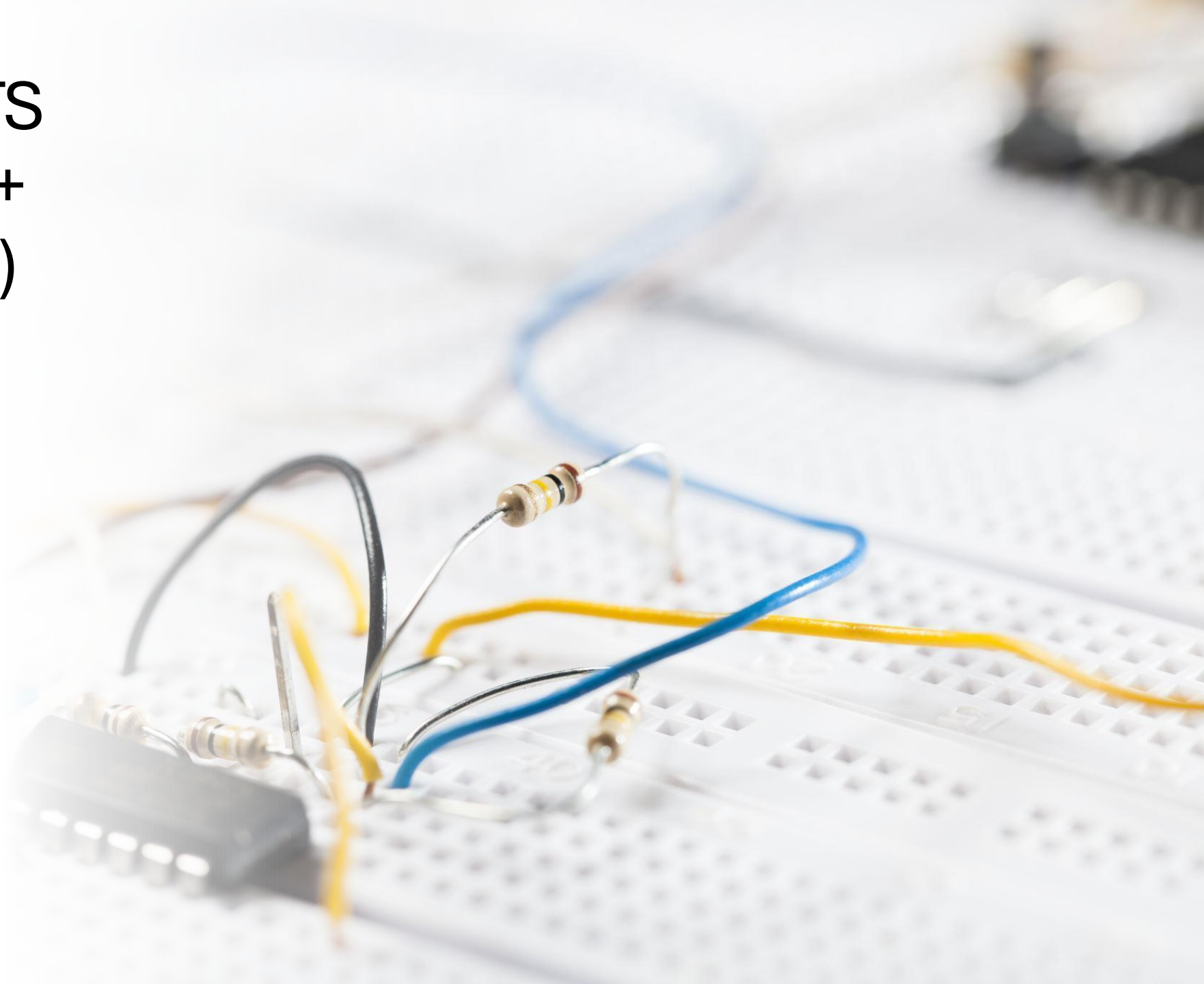
Link to Git Repository: https://github.com/HobbaGobba/23-24-Creative-Making-Experience-and-Physical-Computing/settings/access

2

Link to Tinkercad: https://www.tinkercad.com/things/9cda5ENWTuI-copy-of-ratcircuit-

COMPONENTS (list + images + specifications)

- Arduino Leonardo
- Breadboard (Small)
- 2x Red LEDs
- 2x 290 Ohm resistors
- LDR (Photoresistor)
- 1k Ohm resistor
- Battery Power Bank
- 12x Wires
- Rat 3D print to house my components



How will it work?

The LDR will be used as a more subtle alternative to a proximity sensor. This LDR will be the input creating values which I can use to map the movement of the servo and the brightness of the LEDs.

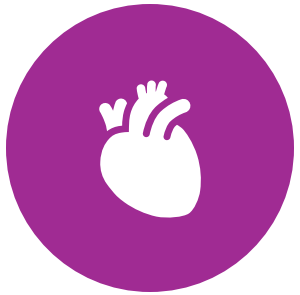
As a person approaches the rat, the LDR value will decrease. To begin with the rat will wag its tail slowly. If a person gets closer, the tail(servo) will begin to wag faster and the brightness of the eyes(LEDs) increases. When a person is extremely close, the tail will wag more aggressively and the brightness of the eyes will be at their maximum.

Why am I doing this project, future applications (Animatronic, applied from lessons)

I was inspired to do this project when learning about LDRs and mapping the movement of a servo respectively to the LDR value.

It provoked an idea of using more discrete components to make rudimentary animatronics, with the hopes of making more complex animatronics in the future.

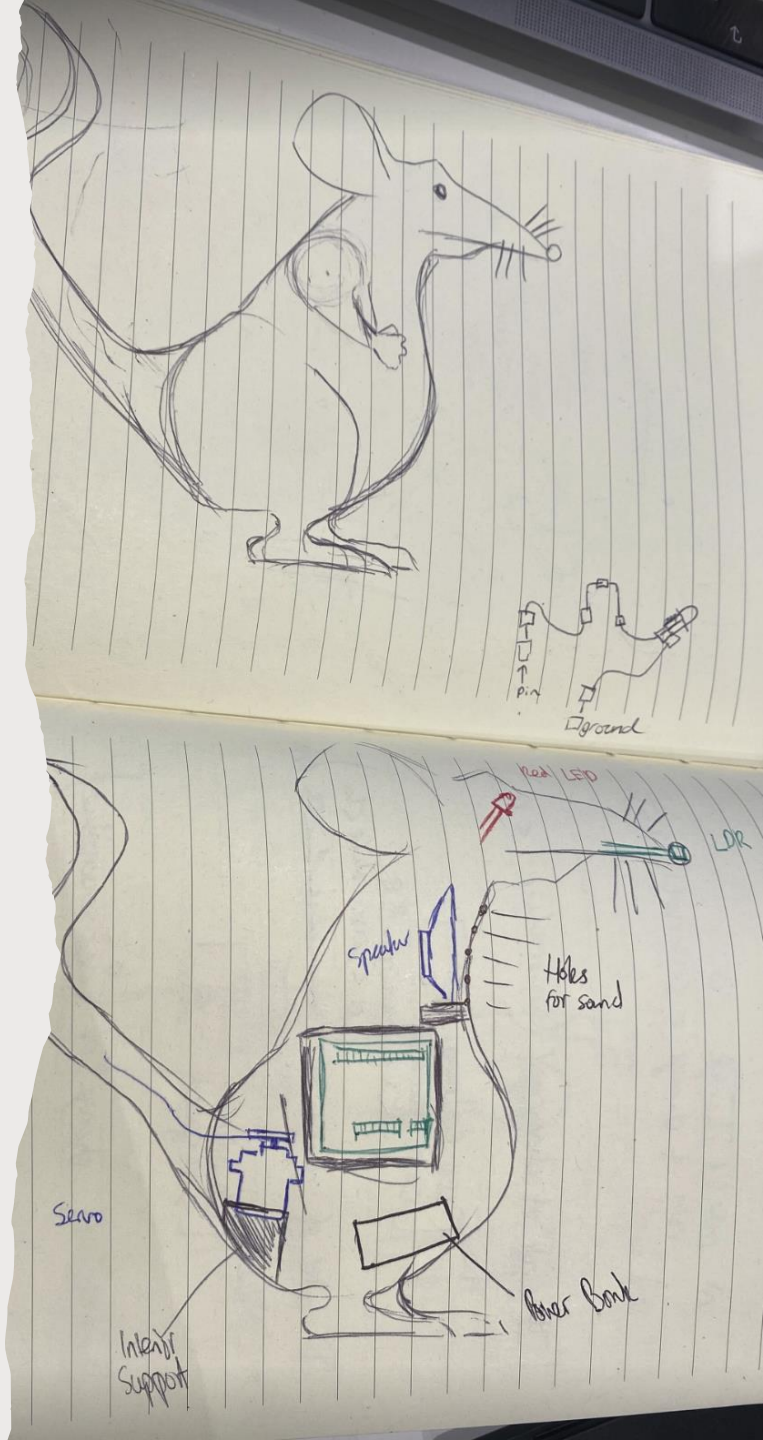
My fondness of animatronics is due to their use in the film industries as a practical approach to visual effects, often creating a tactile and more immersive, especially due to the fact actors have a reference point to act with as opposed to CGI.



I'd love to include the movements of arms and legs in the future.

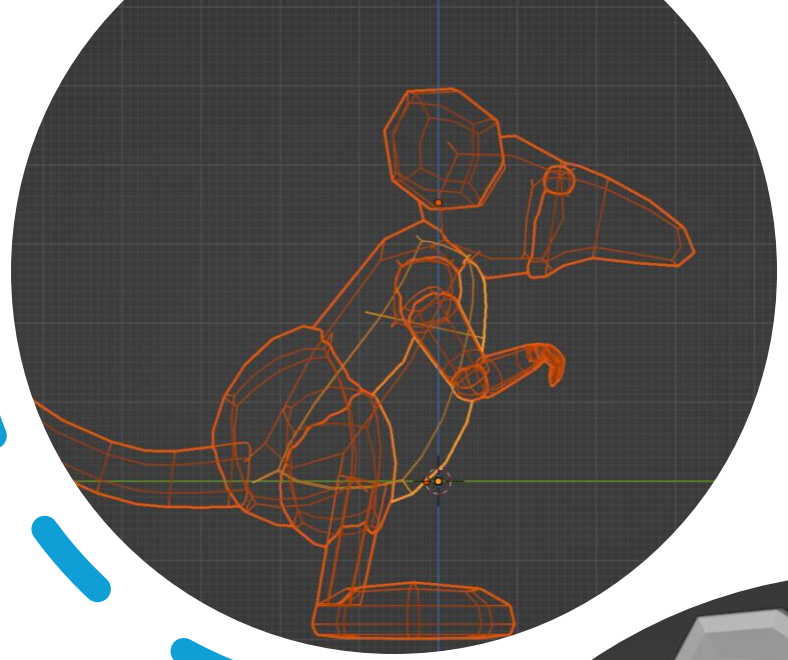
Sketching the 3D model

The first challenge was using my 3D skills in Blender and changing my workflow to something that suited 3D printing. I first quickly sketched out ideas for the rat on my notebook, I went through many iterations to achieve a cute aesthetic whilst ensuring there would be space to house my components inside.



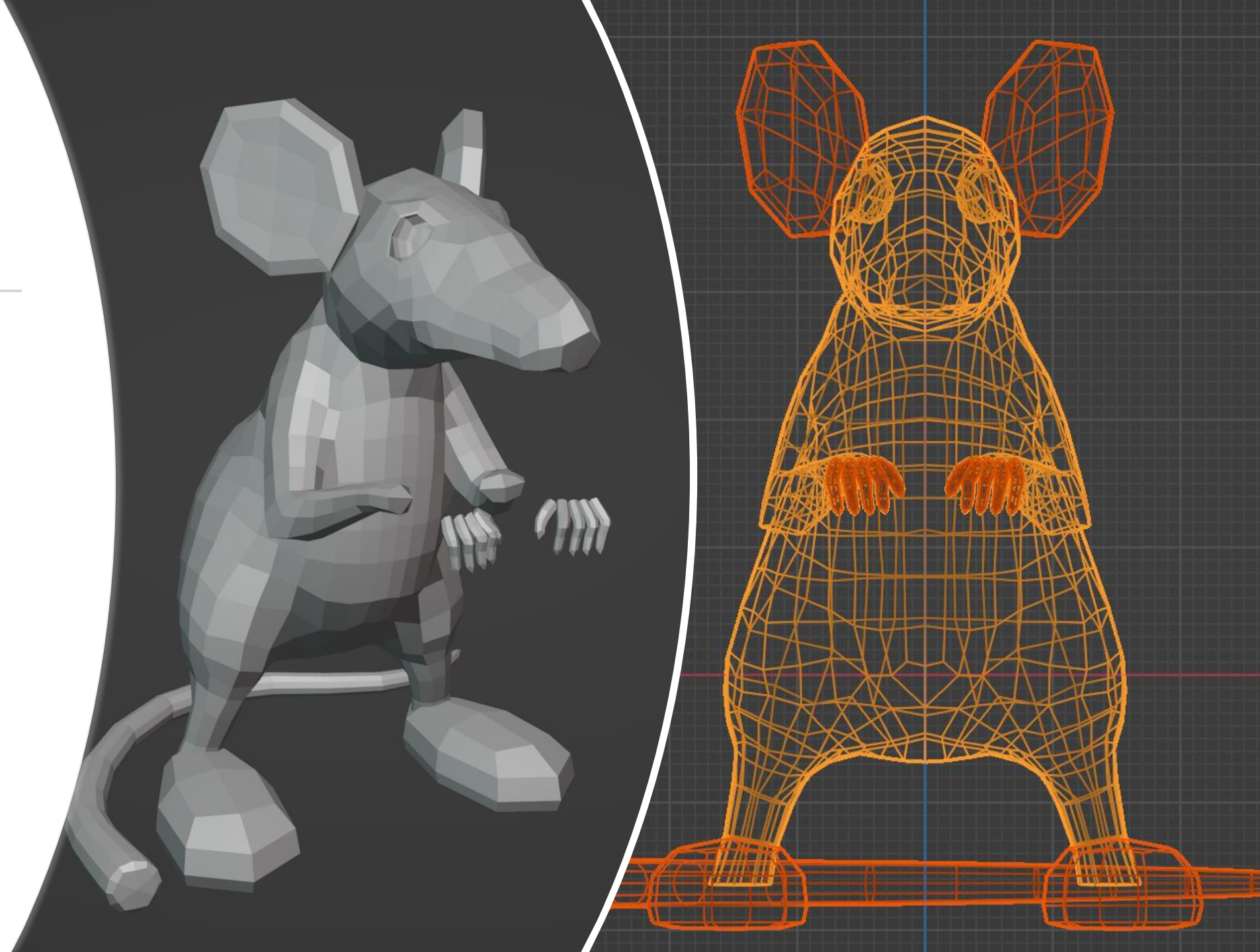
3D Blockout

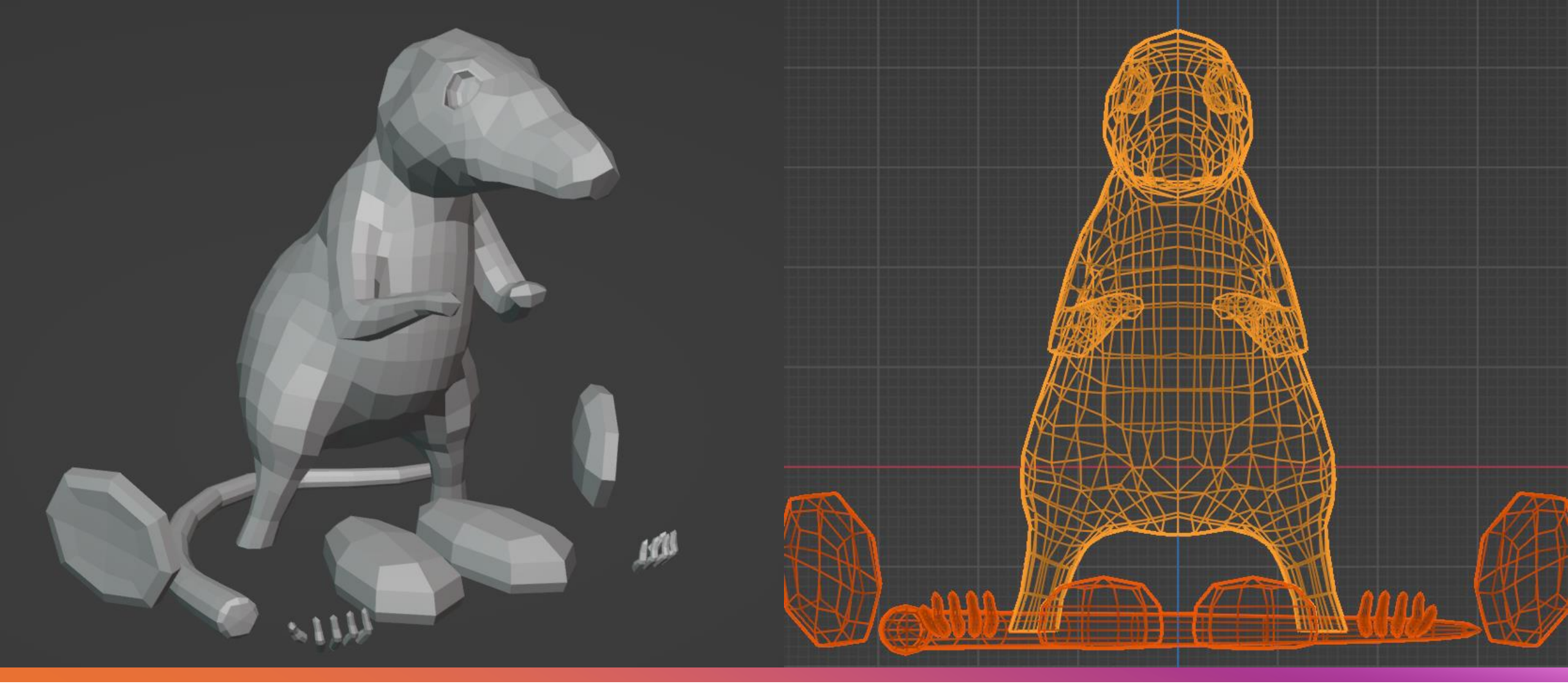
I then proceeded to block out the shape of the rat in Blender using big subdivided cubes. This way I can easily tweak the shape and silhouette of the model.



3D Final Result

Here is the final result after retopologising all the sculpted parts together. The hands, feet and body were kept separate since they all have to be printed separately then joined together with clay.

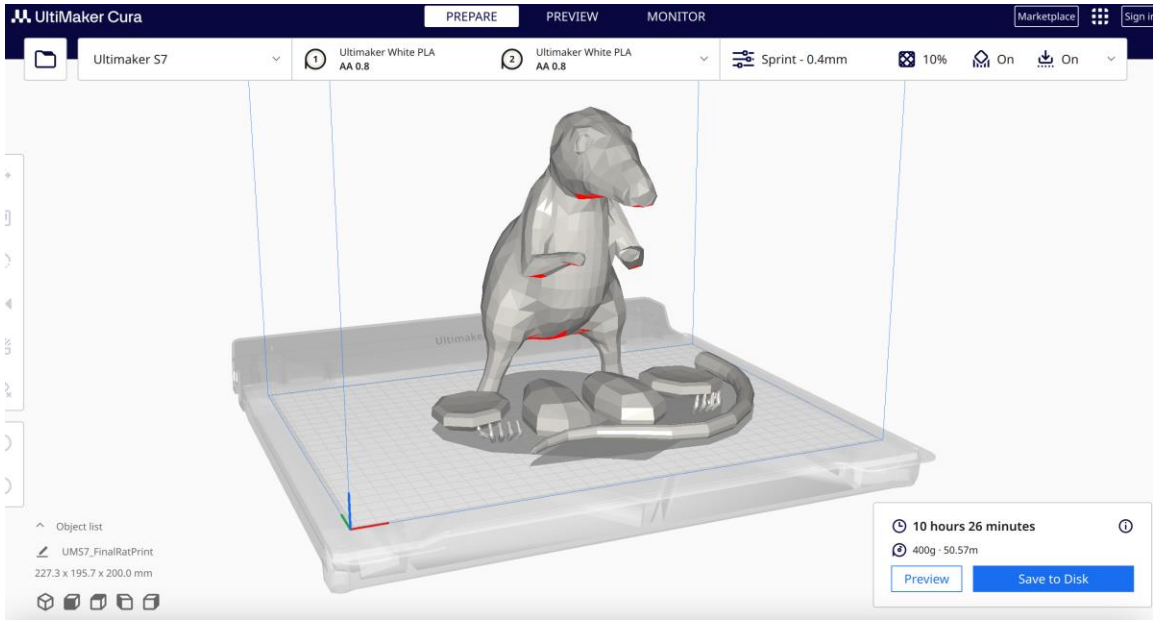


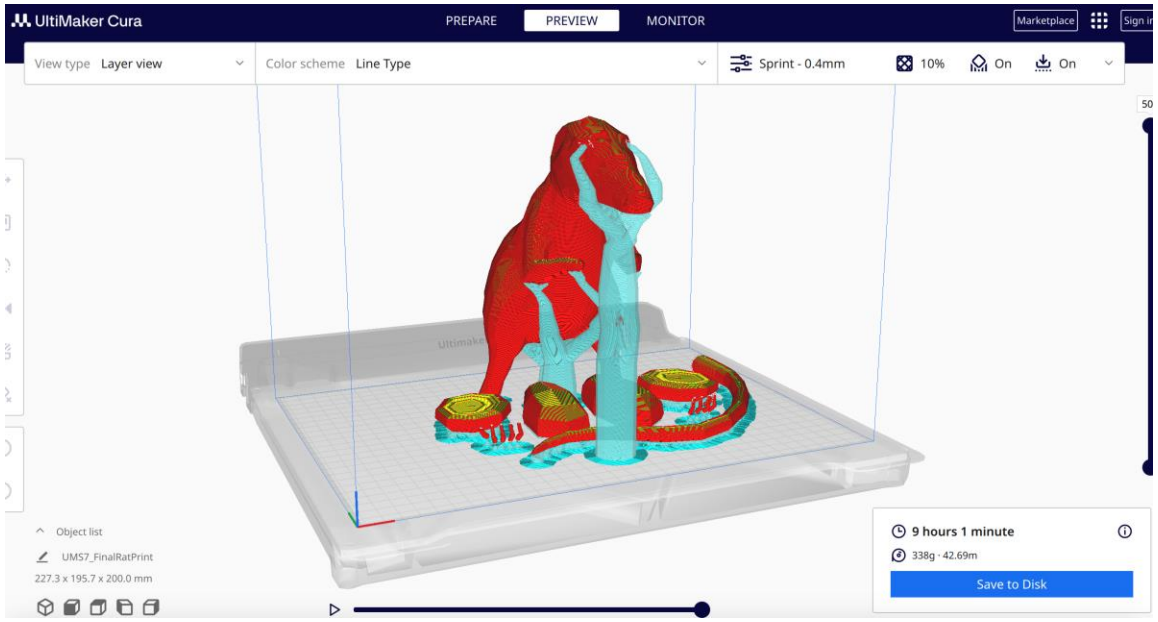


Positioning Out the Parts for 3D Printing

3D Printing (Ultimaker Cura S7)

- Extruder one: White PLA AA 0.8
- Extruder two: White PLA AA 0.8
- Print Settings for Time Efficiency: Sprint Mode 0.4, Triangle Infill, Tree Supports, Adhesion on, 0.8 shell thickness

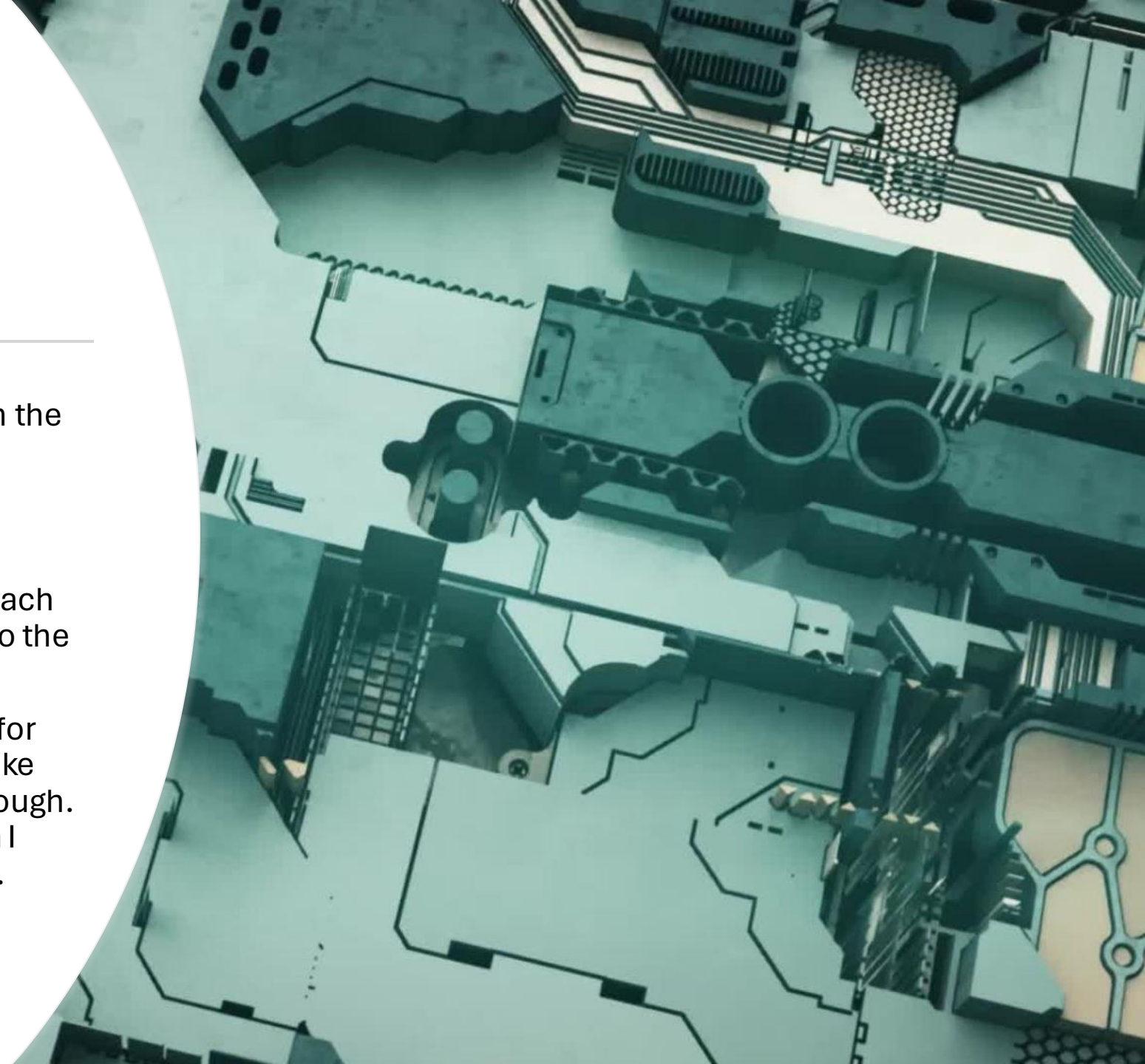


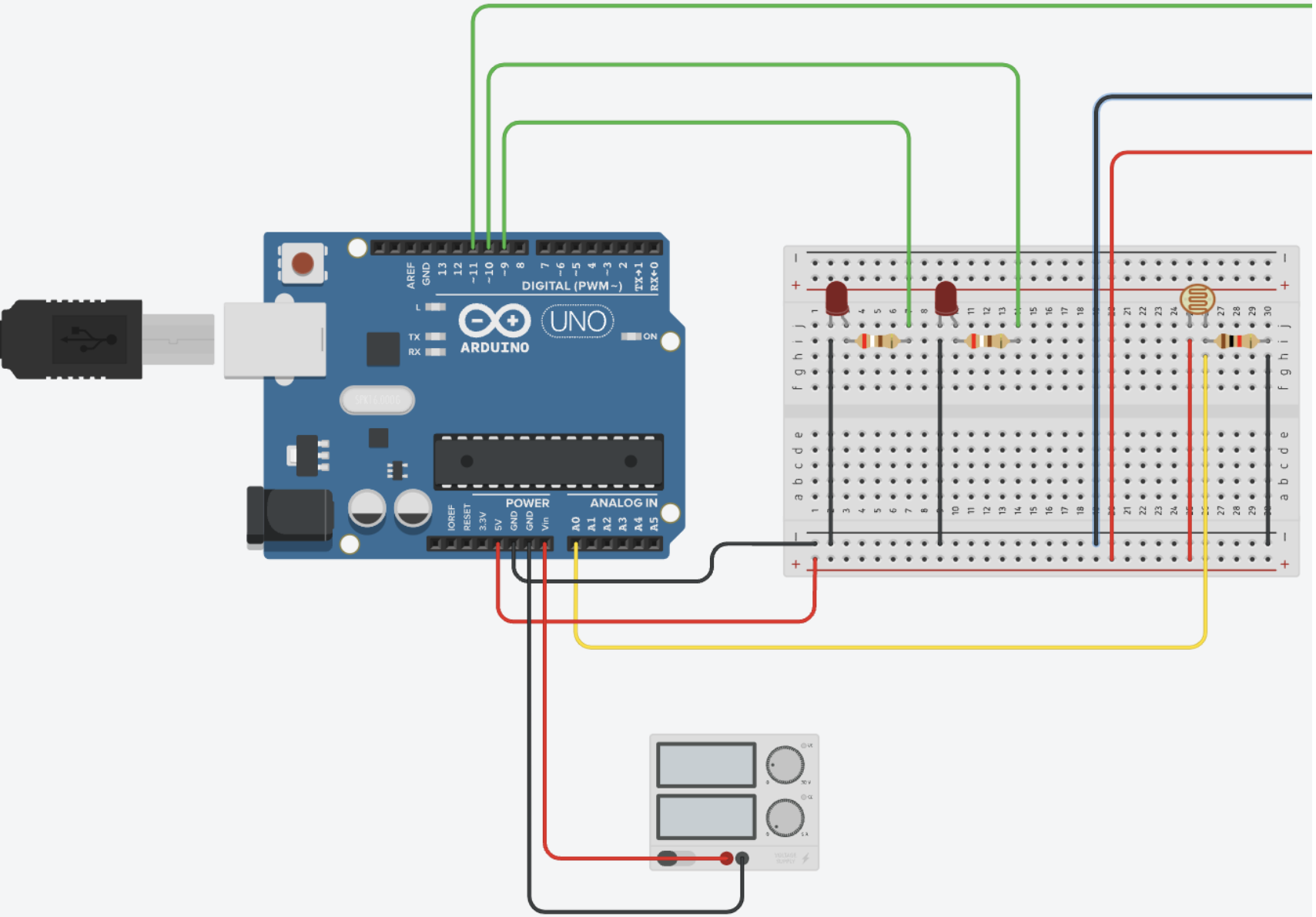


3D Print Result

Errors and what I learned (3D and printing)

- Non manifold edges, there can be no holes in the outer shell of your model
- Needs to be a full outer shell
- In future projects I would try and split up the model into more components that slot into each other. That way I can put the components into the model less destructively.
- I didn't know how to make the object hollow for easier fitting of parts, there is a triangle grid like structure within the print that makes it very tough. It is very difficult to take apart and may mean I cannot house the components within the rat.





Coding Iterations

- To begin with I just want to map out all my components and attempt to get them working. Shown on this slide is the Tinkercad circuit

The code here simply maps the LDR value to servo position and LED brightness

```ino
#include <Servo.h>

#define eyeOne 9
#define eyeTwo 10
Servo myservo;

#define sensorPin A0

//for LDR
int sensorValue = 0;
//for servo
int pos = 0;
int mappedPos = 0;

void setup()
{

  Serial.begin(9600);
  pinMode(eyeOne, OUTPUT);
  pinMode(eyeTwo, OUTPUT);
  myservo.attach(11);
  pinMode(sensorPin, INPUT);

}

void loop()
{
    //reading sensor value
  sensorValue = analogRead(sensorPin);
    //position integer reading sensor value
  pos = analogRead(sensorPin);
    //writing to eyes One and Two via mapping the sensorValue
    //to the range in brightness
  analogWrite(eyeOne, map(sensorValue,0,679,0,255));
  analogWrite(eyeTwo, map(sensorValue,0,679,0,255));
    //mapping the position integer to 0-360 degrees
  mappedPos = map(pos, 0, 679, 0, 360);
    //writing the mappedPos into the servo so it moves
  myservo.write(mappedPos);

  Serial.println(sensorValue);
  //delay(10);
}

//I have used the LDR to interact with all items

//Step One: There is a brightness isue with the LEDS, check all
//resistors are correct, and that there is sufficient power supplied

//Step Two: Convert the code so that for at a certain sensorValue
//the tail moves continuously and the LEDs light up.
//e.g when sensorValue = 800, the eyes start lighting up,
//when sensorValue = 600, the servo starts moving left and right.


//Step Three: Make it so that the rate of change of the sensorValue
//is what affects the speed at which the servo operates. and also
//the rate at which the red eyes blink, like a warning
```

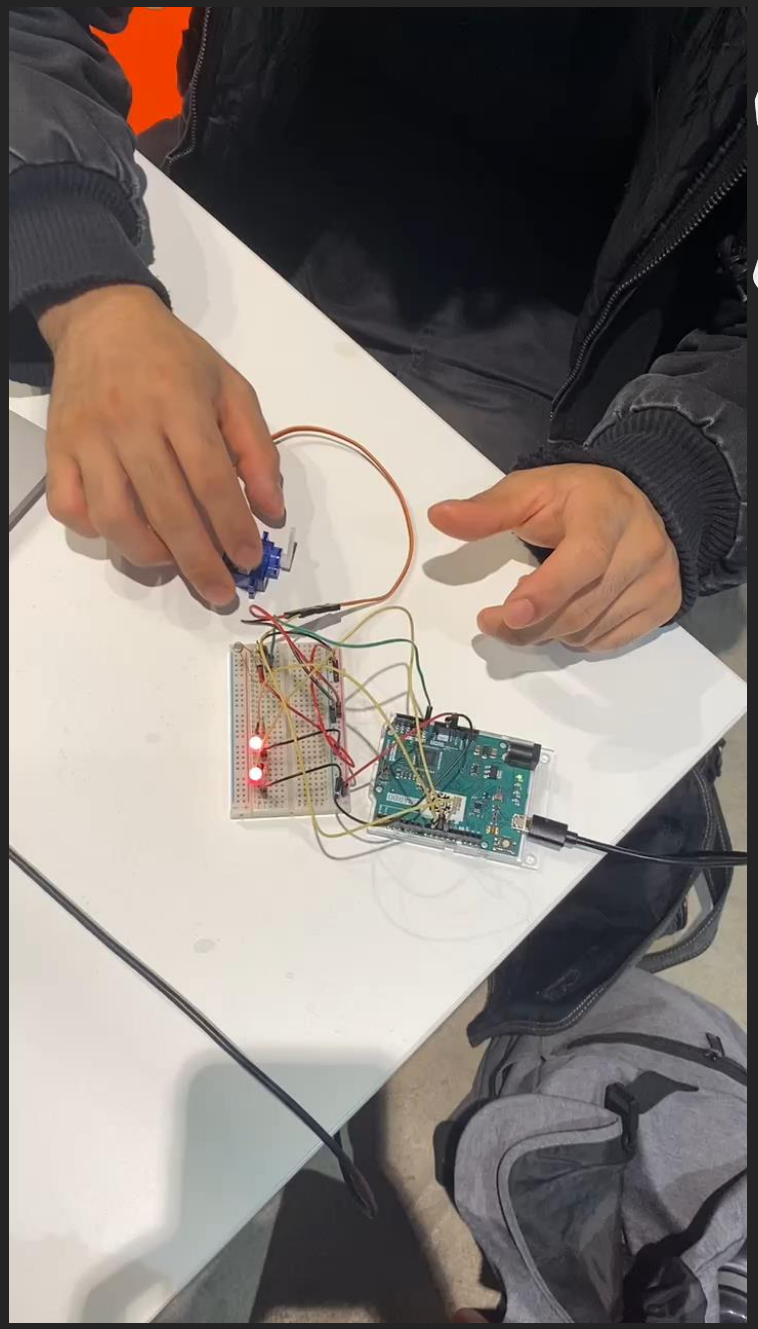
Adjustments to code and project direction

I found that the LDR value was changing sporadically despite no changes in environment. In order to smooth out these values that my components are dependant on. I decided to take an average of 10 readings then using that value as a variable.

```
#include <Servo.h>

#define eyeOne 9
#define eyeTwo 10
Servo myservo;

#define sensorPin A0

//for LDR
int sensorValue = 0;
//for servo
int pos = 0;

void setup()
{
  Serial.begin(9600);
  pinMode(eyeOne, OUTPUT);
  pinMode(eyeTwo, OUTPUT);
  myservo.attach(11);
  pinMode(sensorPin, INPUT);
}

void loop()
{
 //change = sensorValue - copyValue; RATE OF CHANGE IDEA

 //DATA SMOOTHING FOR LIGHT SENSOR//
 //sensorValue = analogRead(sensorPin);
 int numReads = 10; // number of samples
 int sensorSum = 0; // sum of readings
```

```

 //Getting a sum of readings via for loop
 for( int i = 0; i < numReads; i++){
  sensorValue = analogRead(sensorPin);
  sensorSum += sensorValue;
  delay(1); //totals up to a delay of 10 per loop
  }
  int sensorAverage = sensorSum / numReads; //average of sensor rea
  int sensortoBrightness = map(sensorAverage, 900, 0, 0, 255);//for
  Serial.println(sensorValue);
  Serial.println(sensorAverage); //printing the average to terminal
  Serial.println(sensortoBrightness);
  digitalWrite(eyeOne, sensortoBrightness);
  digitalWrite(eyeTwo, sensortoBrightness);

  //make it so that tail isnt wagging to begin with so motor doesnt
  //make it so that if servoAverage = highValue, delay = 5
  //and if ServoAverage = lowValue. delay = 3
  //and if ServoAverage = superlowValue, delay = 2

  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 1
       // in steps of 1 degree
    myservo.write(pos);
    if (sensorAverage > 650) {
      delay(5); //neutral speed wagging of tail SHOULD ADJUST TO A
    }
    else if ((sensorAverage > 550) && (sensorAverage <= 650)) {
        delay(3); //frantic speed wagging of tail
    }
```

```
    else if (sensorAverage <= 550) {
        delay(2); //very frantic wagging of tail
    }
  }

  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    if (sensorAverage > 650) {
      delay(5); //neutral speed wagging of tail
    }
    else if ((sensorAverage > 550) && (sensorAverage <= 650)) {
        delay(3); //frantic speed wagging of tail
    }
    else if (sensorAverage <= 550) {
        delay(2); //very frantic wagging of tail
    }
  }
    |
}


//I have used the LDR to interact with all items

//Step One: There is a brightness isue with the LEDS, check all
//resistors are correct, and that there is sufficient power supplied

//Step Two: Convert the code so that for at a certain sensorValue
//the tail moves continuously and the LEDs light up.
//e.g when sensorValue = 800, the eyes start lighting up,
//when sensorValue = 600, the servo starts moving left and right.

//CREATE: One more case for if the person is extremely close, with
//the eyes of the rat flashing.
```



Video of code and components working

Discovered an issue with my LED brightness not changing as I approach the LDR. It was just turning off and on.

I need to replace the digitalWrite() for LED brightness to analogWrite() as digitalWrite() only accepts high and low, whilst analogWrite() accepts integers between 0-255 for brightness. And now all my code works.

How would you like to improve and correct the project. What ideas has it given you?

- I would love to find a way to create a 3D printable hollow object with slots for electrical components. It would really assist in the final look of the project. Also making the material weaker so I can cut it open and instert components.
- Perhaps make the input more exciting. Such as calculating the rate of change of the sensor value so the rat reacts more dynamically
- Inclusion of a speaker that emits rat noises and squeaks depending on the situation
- Using a proximity sensor instead of an LDR as the values are more linear
- Streamline the workflow of 3D to print to finish.
- Paint and add fur/felt to the rat to make it more lifelike.