Comme dans un LabBook, rien ne devra être supprimé de ce document.
Toute chose voulant être supprimée, devra être ~~barrée~~ ou commentée.

Jeudi 19 janvier :

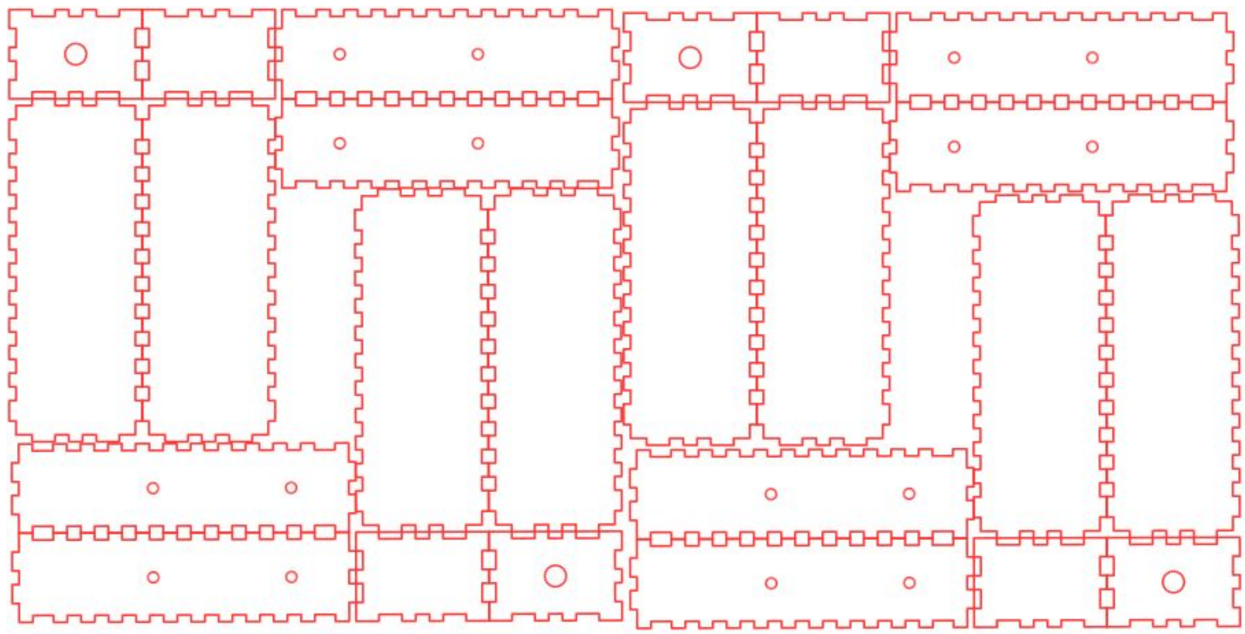
Wet lab material :
- 13 syringes + 15 needles + 2mmØ tubes
- Blue ink
- Water
- *Egeria densa*

OpenLab material :
- 13 boxes with 4 circle openings for 4 red LEDs, for each syringe.
- 1 photoreceptor or photoresistance
- 3 arduinos
- Cables
- 15 * 4 red LEDs - White LEDs (a few)
- One power source : computer

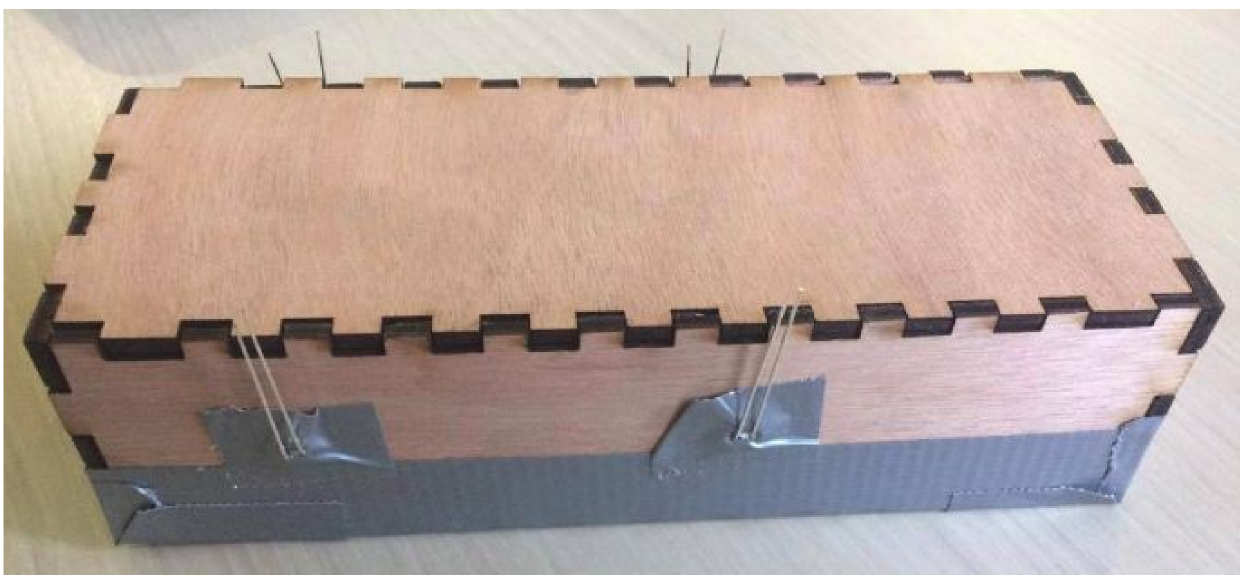
Protocols :

- Design of the boxes :
    - Design 13 boxes with 3mm medium, with *www.makercase.com* (our boxes are 6x16x4 cm)
    - Put hole in it, where you want the LED to light up and where the needle will go out of the box.
    - Cut it with a laser cuter.
    - When the boxes are cut, build them and scotch all the angle with an opac scotch so that no light can go through and you have complet darkness.

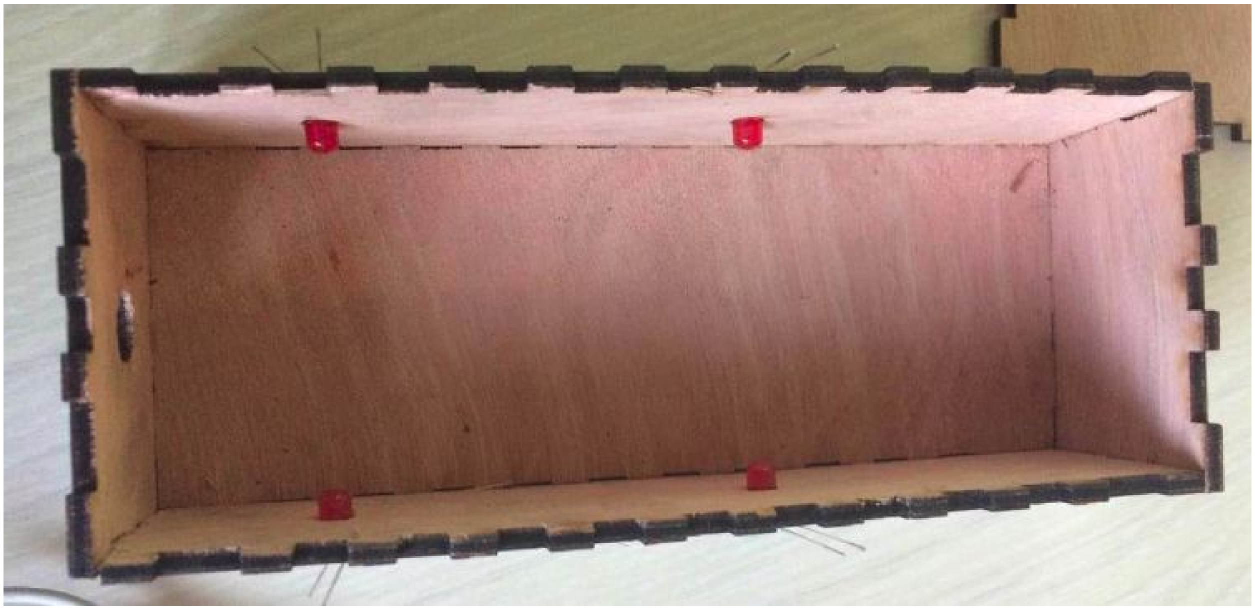
Drawings of the plans to design our boxes : 4 boxes can be cut with this plan



Pictures of the boxes :



Inside of the box :



- WetLab protocol :

  - Fill a first syringe with 60 mL of water
  - Add 8 cm of the top of the *Egeria densa* in the syringe
  - Remove any air that might be inside the syringe
  - Mix 8 drops of blue ink to 250 ml of water in a beaker
  - Fill the 2mm tube with the solution of water and ink until it is full
  - Do NOT plug the tube with the syringe containing *Egeria densa* otherwise, the reaction will start before you will take the measures.
  - Scotch the tube on the bench so that it will not move when doing the experiment
  - On the tube (or on the bench, or on a paper under the tube), write marks every 1cm
  - Place a camera able to picture time laps in a way that we can see all the setup on it
  - Now you can plug the tube full of ink with the syringe with *Egeria densa*.
  - As soon as the air passed by the first mark, you can start the time laps.

Vendredi 20 janvier

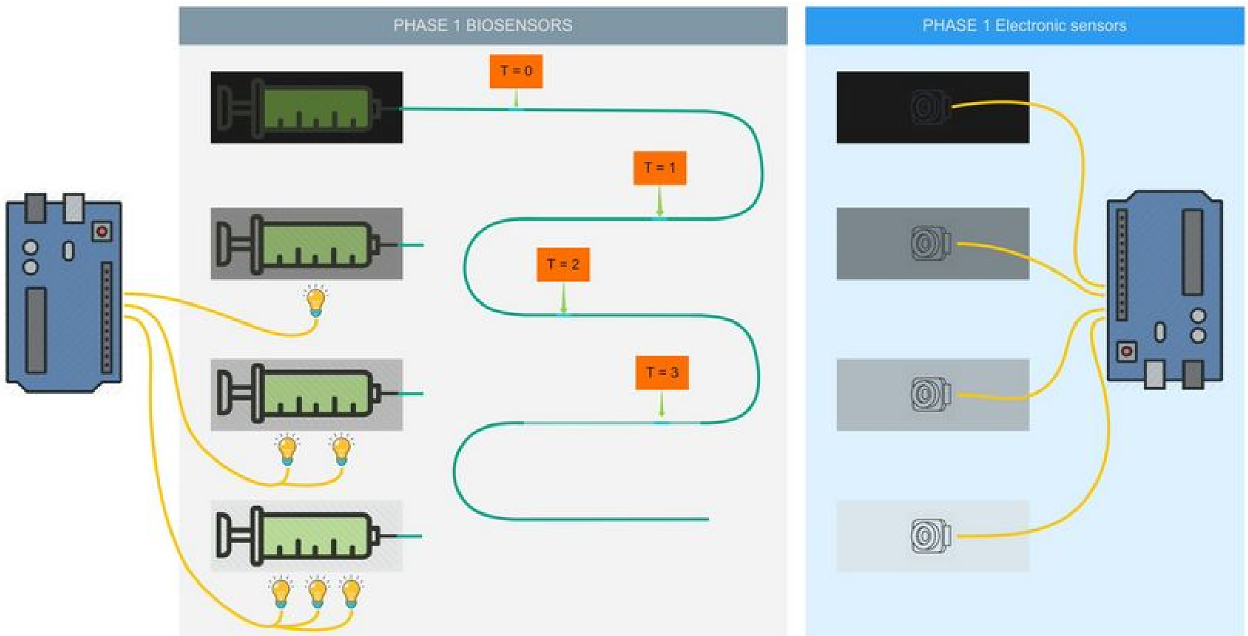
We took our time laps during 2 hours each time.
We did 3 experiments :

- 3 different boxes, with each the same LED intensity which is 25% (64)

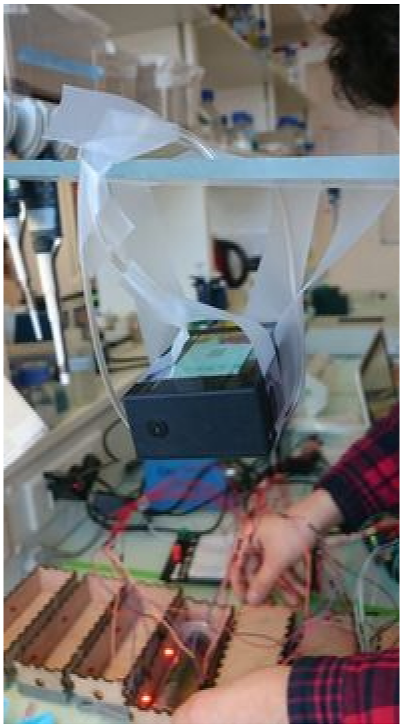
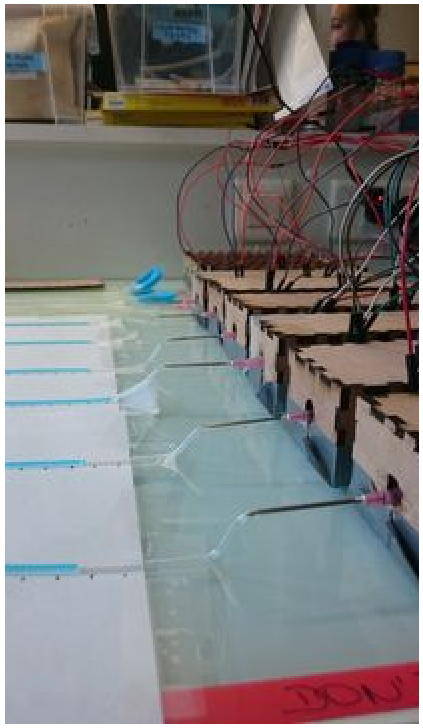
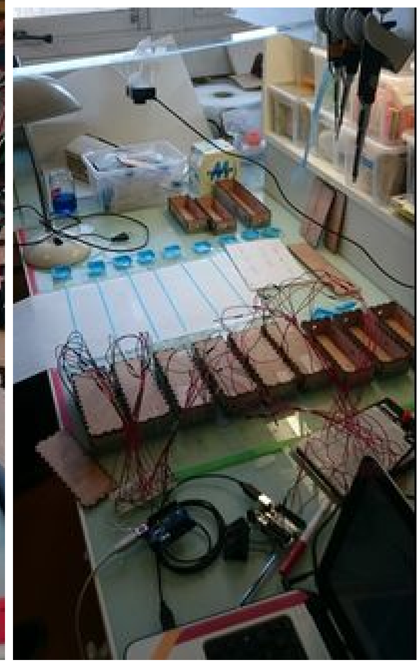
- 6 different boxes, with 3 at a 50% (127) intensity, and the 3 others with a 75% (191) intensity
- 6 different boxes, with 1 with no light that will be the negative control, 2 with a 100% intensity, 2 with 25 % intensit

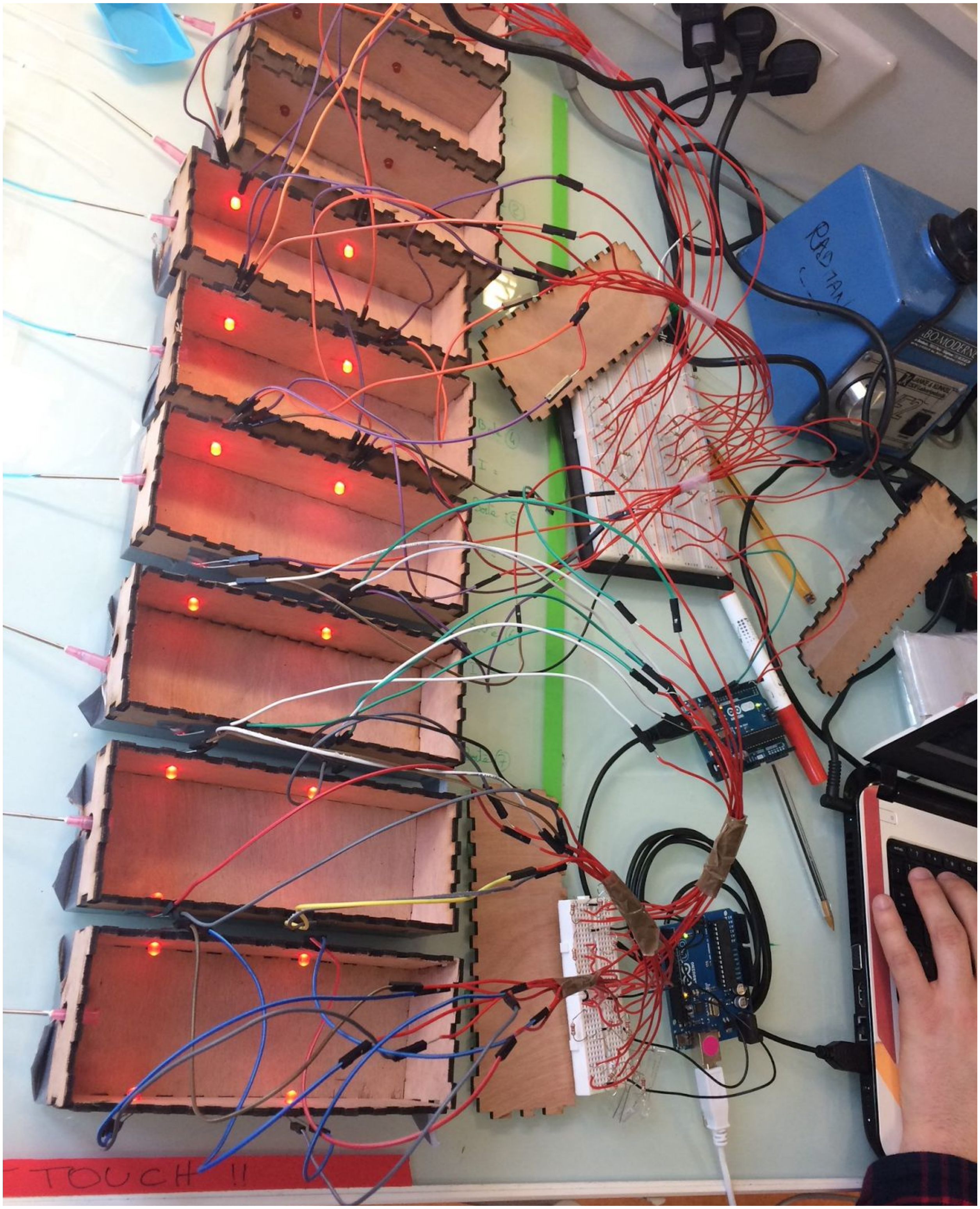
We did several boxes at the same intensity to have replicates.
We introduce carefully the needles in the tube, otherwise the tubes would have been pierced.

Schematics of the setup :

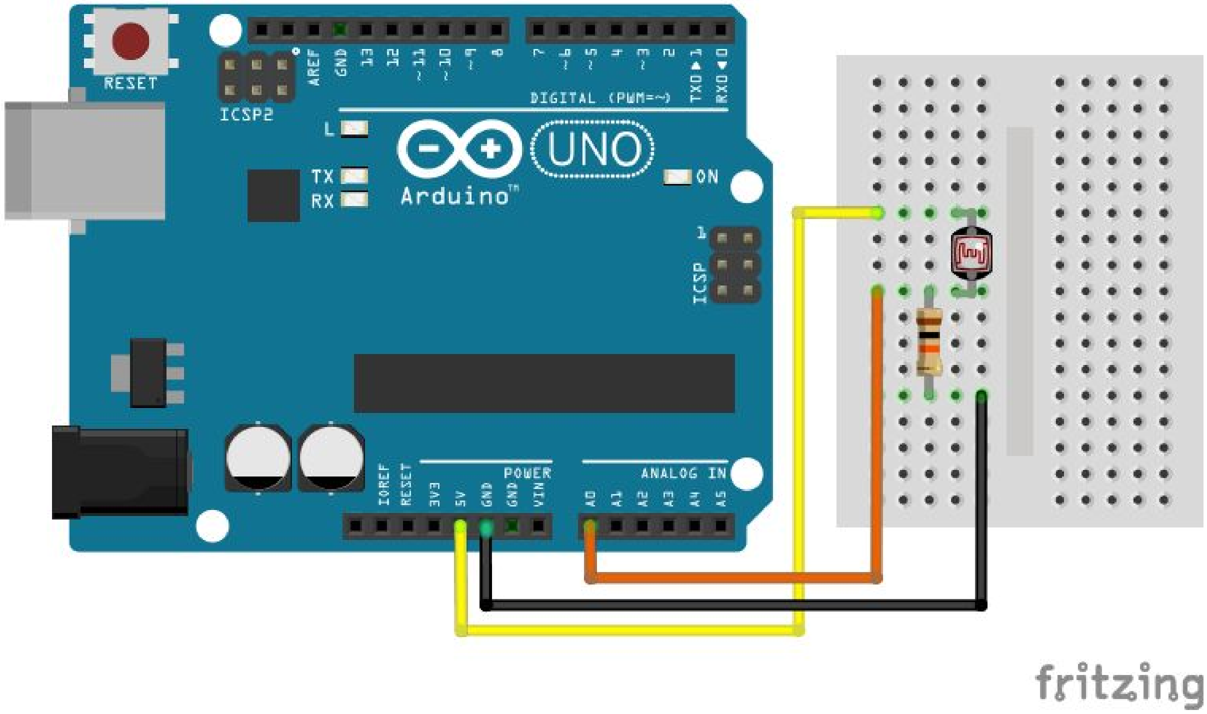


Pictures of the setup :

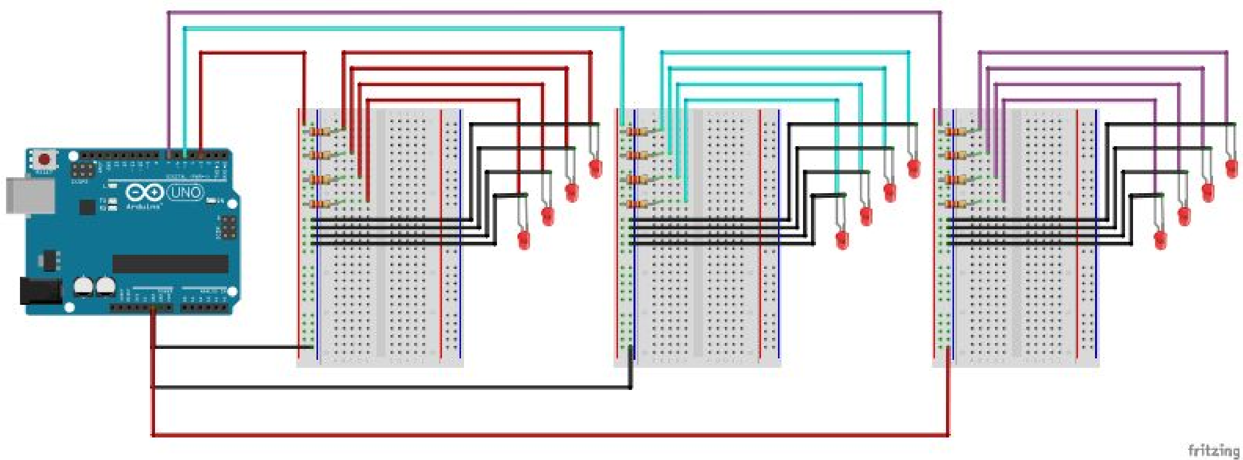





TOUCH !!

Arduino’s setup with a photo-resistance :



Arduino’s setup to light up the LEDs :



The Adruino’s code to set the 4 LEDs intensity for 3 boxes on the biological sensor :

```
//Initialisation
const int ledPin1 = 3;
const int ledPin2 = 5;
const int ledPin3 = 9;
int valPWM = 64; //64 = 25%, 127 = 50%, 191 = 75%, 255 = 100% - règle l'intensité des LEDs

void setup() {
  // Initialise le Pin comme une sortie
  pinMode(ledPin1, OUTPUT);
  pinMode(ledPin2, OUTPUT);
  pinMode(ledPin2, OUTPUT);
}

void loop() {
  analogWrite(ledPin1, valPWM);
  analogWrite(ledPin2, valPWM);
  analogWrite(ledPin3, valPWM);
}
```

The Arduino’s code used to do 50 measures for one LED intensity for the electronic sensor :

```
//Constantes
const int ledPin1 = 3;
int valPWM = 255; //64 = 25%, 127 = 50%, 191 = 75%, 255 = 100%
int count = 0; //variable

// Fonction setup(), appelÃ©e au dÃ©marrage de la carte Arduino
void setup() {
   // Initialise le Pin comme une sortie | Initialize the digital pin as an output with pinMode()
  pinMode(ledPin1, OUTPUT);
  // Initialise la communication avec le PC
  Serial.begin(9600);
}

// Fonction loop(), appelÃ©e continuellement en boucle tant que la carte Arduino est alimentÃ©e
void loop() {
  if (count == 0) {
    delay(5000);
  }
  analogWrite(ledPin1, valPWM);
  if (count <= 50) {
    int valeur = analogRead(A0);   // Mesure la tension sur la broche A0
  //  int valeurnew = map(valeur, 0, 1024, 0, 10);
    // Envoi la mesure au PC pour affichage et attends 250ms
    Serial.print(valPWM);
    Serial.print(',');
    Serial.println(valeur);
    count ++ ;1;
    delay(100);
  }
  }
```

The Arduino’s code used to measure all the intensities with a photoresistor - 100ms (change delay) :

```
/*
 * Code d'exemple pour une photorÃ©sistance.
 */
//Constantes
const int ledPin1 = 3;
int valPWM = 0; //64 = 25%, 127 = 50%, 191 = 75%, 255 = 100%

// Fonction setup(), appelÃ©e au dÃ©marrage de la carte Arduino
void setup() {
   // Initialise le Pin comme une sortie | Initialize the digital pin as an output with pinMode()
  pinMode(ledPin1, OUTPUT);
  // Initialise la communication avec le PC
  Serial.begin(9600);
}

// Fonction loop(), appelÃ©e continuellement en boucle tant que la carte Arduino est alimentÃ©e
void loop() {
  if (valPWM == 0) {
    delay(5000);
  }
  if (valPWM <= 255) {
    analogWrite(ledPin1, valPWM);
    int valeur = analogRead(A0);   // Mesure la tension sur la broche A0
  //  int valeurnew = map(valeur, 0, 1024, 0, 10);
    // Envoi la mesure au PC pour affichage et attends 250ms
    Serial.print(valPWM);
    Serial.print(',');
    Serial.println(valeur);
    valPWM ++ ;1;
    delay(100);
  }
  }
```

The code used to plot the biological data without the smoothed curves :

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates

time501, velocity501 = np.genfromtxt("data-50%-1.csv", unpack=True, delimiter=",", skip_header=2)

time502, velocity502 = np.genfromtxt("data-50%-2.csv", unpack=True, delimiter=",", skip_header=2)

time503, velocity503 = np.genfromtxt("50%-3.csv", unpack=True, delimiter=",", skip_header=2)

time751, velocity751 = np.genfromtxt("data75%-1.csv", unpack=True, delimiter=",", skip_header=2)

time752, velocity752 = np.genfromtxt("data75%-2.csv", unpack=True, delimiter=",", skip_header=2)

time251, velocity251 = np.genfromtxt("25%-1.csv", unpack=True, delimiter=",", skip_header=2)

time252, velocity252 = np.genfromtxt("25%-2.csv", unpack=True, delimiter=",", skip_header=2)
```

```
time253, velocity253 = np.genfromtxt("25%-3.csv", unpack=True, delimiter=",", skip_header=2)

timeMAX1, velocityMAX1 = np.genfromtxt("MAX1.csv", unpack=True, delimiter=",", skip_header=2)

timeNoir, velocityNoir = np.genfromtxt("Noir.csv", unpack=True, delimiter=",", skip_header=2)

Velocity501 = velocity501 / 1.8
Velocity502 = velocity502 / 1.8
Velocity503 = velocity503 / 1.8
Velocity751 = velocity751 / 1.8
Velocity752 = velocity752 / 1.8
Velocity251 = velocity251 / 1.8
Velocity252 = velocity252 / 1.8
Velocity253 = velocity253 / 1.8
VelocityMAX1 = velocityMAX1 / 1.8
VelocityNoir = velocityNoir / 1.8

axes = plt.gca()

plt.subplot(221)
plt.xlabel("Time in min")
plt.ylabel("Velocity of the gas release in cm.min⁻¹")
plt.title("Velocity of the gas release according to time - 50% LED intensity")
plt.plot(time501, Velocity501, "b", label="Replicate 1")
plt.plot(time502, Velocity502, "c", label="Replicate 2")
plt.plot(time502, Velocity503, "m", label="Replicate 3")
plt.ylim([0,0.8])
plt.legend()

plt.subplot(222)
plt.xlabel("Time in min")
plt.ylabel("Velocity of the gas release in cm.min⁻¹")
plt.title("Velocity of the gas release according to time - 75% LED intensity")
plt.plot(time751, Velocity751, "g", label="Replicate 1")
plt.plot(time752, Velocity752, "y", label="Replicate 2")
plt.ylim([0,0.8])
plt.legend()

plt.subplot(223)
plt.xlabel("Time in min")
plt.ylabel("Velocity of the gas release in cm.min⁻¹")
plt.title("Velocity of the gas release according to time - 25% LED intensity")
plt.plot(time251, Velocity251, "r", label="Replicate 1")
plt.plot(time252, Velocity252, "k", label="Replicate 2")
plt.plot(time253, Velocity253, "m", label="Replicate 3")
plt.ylim([0,0.8])
plt.legend()

plt.subplot(224)
plt.xlabel("Time in min")
plt.ylabel("Velocity of the gas release in cm.min⁻¹")
plt.title("Velocity of the gas release according to time - Controls")
plt.plot(timeMAX1, VelocityMAX1, "y", label="Positive control")
plt.plot(timeNoir, VelocityNoir, "k", label="Negative control")
plt.ylim([0,0.8])
plt.legend()
```

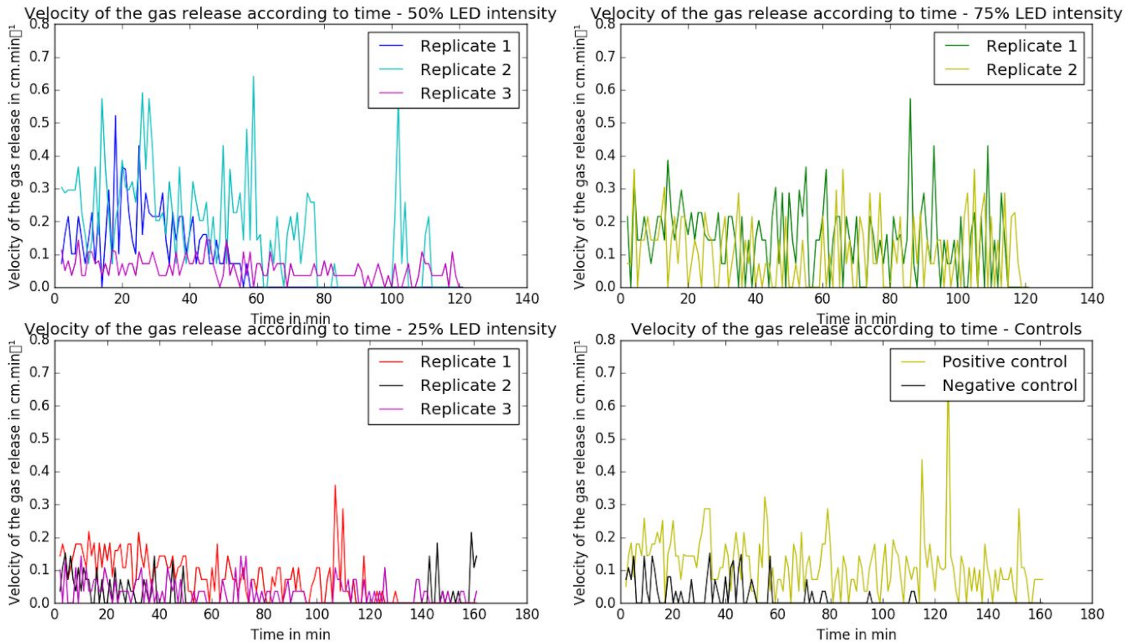
```
plt.show()
```

The code used to plot the biological data with the smoothed curves :

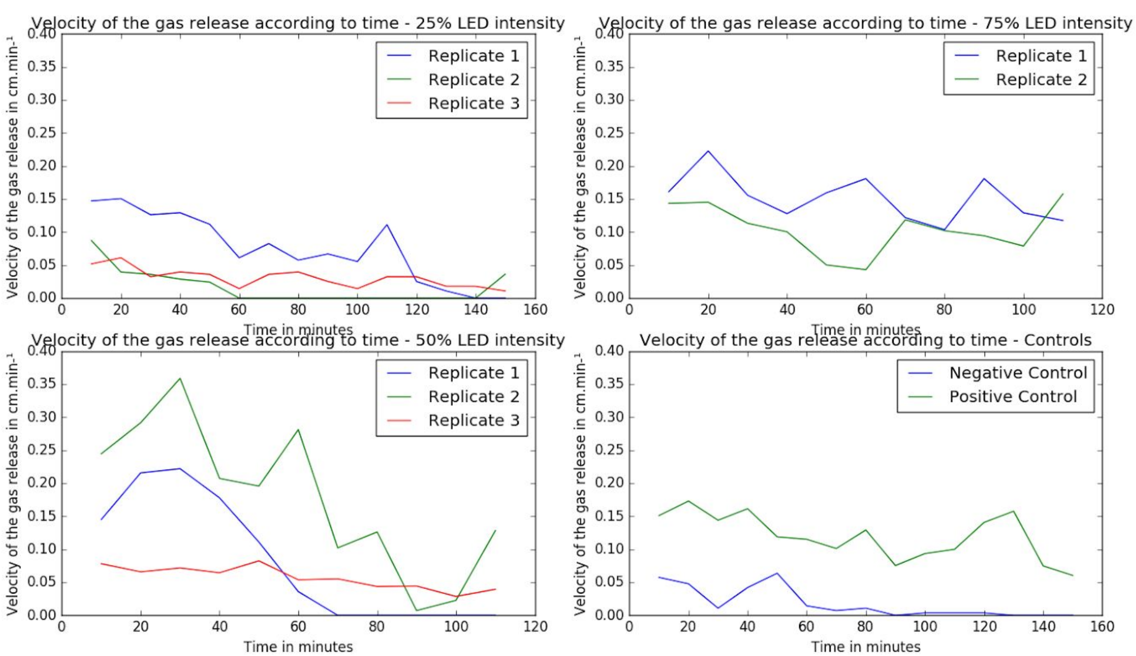
```
import numpy as np
import matplotlib.pyplot as plt

time501, velocity501 = np.genfromtxt("data-50%-1.csv", unpack=True, delimiter=",", skip_header=2)

time502, velocity502 = np.genfromtxt("data-50%-2.csv", unpack=True, delimiter=",", skip_header=2)

time503, velocity503 = np.genfromtxt("50%-3.csv", unpack=True, delimiter=",", skip_header=2)

time751, velocity751 = np.genfromtxt("data75%-1.csv", unpack=True, delimiter=",", skip_header=2)

time752, velocity752 = np.genfromtxt("data75%-2.csv", unpack=True, delimiter=",", skip_header=2)

time251, velocity251 = np.genfromtxt("25%-1.csv", unpack=True, delimiter=",", skip_header=2)

time252, velocity252 = np.genfromtxt("25%-2.csv", unpack=True, delimiter=",", skip_header=2)

time253, velocity253 = np.genfromtxt("25%-3.csv", unpack=True, delimiter=",", skip_header=2)

timeMAX1, velocityMAX1 = np.genfromtxt("MAX1.csv", unpack=True, delimiter=",", skip_header=2)

timeNoir, velocityNoir = np.genfromtxt("Noir.csv", unpack=True, delimiter=",", skip_header=2)

Velocity501 = velocity501 / 1.8
Velocity502 = velocity502 / 1.8
Velocity503 = velocity503 / 1.8
Velocity751 = velocity751 / 1.8
Velocity752 = velocity752 / 1.8
Velocity251 = velocity251 / 1.8
Velocity252 = velocity252 / 1.8
Velocity253 = velocity253 / 1.8
VelocityMAX1 = velocityMAX1 / 1.8
VelocityNoir = velocityNoir / 1.8

def smoothitout(OneDnumpy, number_of_bins):
  data = np.split(OneDnumpy, len(OneDnumpy) / number_of_bins)
  Mean = []
  for i in range(len(data) - 1):
    Mean.append(np.mean(data[i]))
  return Mean

dataNoir = smoothitout(VelocityNoir, 10)
dataMAX1 = smoothitout(VelocityMAX1, 10)
data501 = smoothitout(Velocity501, 10)
data502 = smoothitout(Velocity502, 10)
data503 = smoothitout(Velocity503, 10)
data251 = smoothitout(Velocity251, 10)
data252 = smoothitout(Velocity252, 10)
```

```
data253 = smoothitout(Velocity253, 10)
data751 = smoothitout(Velocity751, 10)
data752 = smoothitout(Velocity752, 10)
Time11 = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110]
Time15 = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150]


plt.subplot(221)
plt.xlabel("Time in minutes")
plt.ylabel("Velocity of the gas release in cm.min-¹")
plt.title("Velocity of the gas release according to time - 25% LED intensity")
plt.plot(Time15, data251, label='Replicate 1')
plt.plot(Time15, data252, label='Replicate 2')
plt.plot(Time15, data253, label='Replicate 3')
plt.legend()
plt.ylim([0,0.4])

plt.subplot(222)
plt.xlabel("Time in minutes")
plt.ylabel("Velocity of the gas release in cm.min-¹")
plt.title("Velocity of the gas release according to time - 75% LED intensity")
plt.plot(Time11, data751, label='Replicate 1')
plt.plot(Time11, data752, label='Replicate 2')
plt.legend()
plt.ylim([0,0.4])

plt.subplot(223)
plt.xlabel("Time in minutes")
plt.ylabel("Velocity of the gas release in cm.min-¹")
plt.title("Velocity of the gas release according to time - 50% LED intensity")
plt.plot(Time11, data501, label='Replicate 1')
plt.plot(Time11, data502, label='Replicate 2')
plt.plot(Time11, data503, label='Replicate 3')
plt.legend()
plt.ylim([0,0.4])

plt.subplot(224)
plt.xlabel("Time in minutes")
plt.ylabel("Velocity of the gas release in cm.min-¹")
plt.title("Velocity of the gas release according to time - Controls")
plt.plot(Time15, dataNoir, label='Negative Control')
plt.plot(Time15, dataMAX1, label='Positive Control')
plt.legend()
plt.ylim([0,0.4])
plt.show()
```

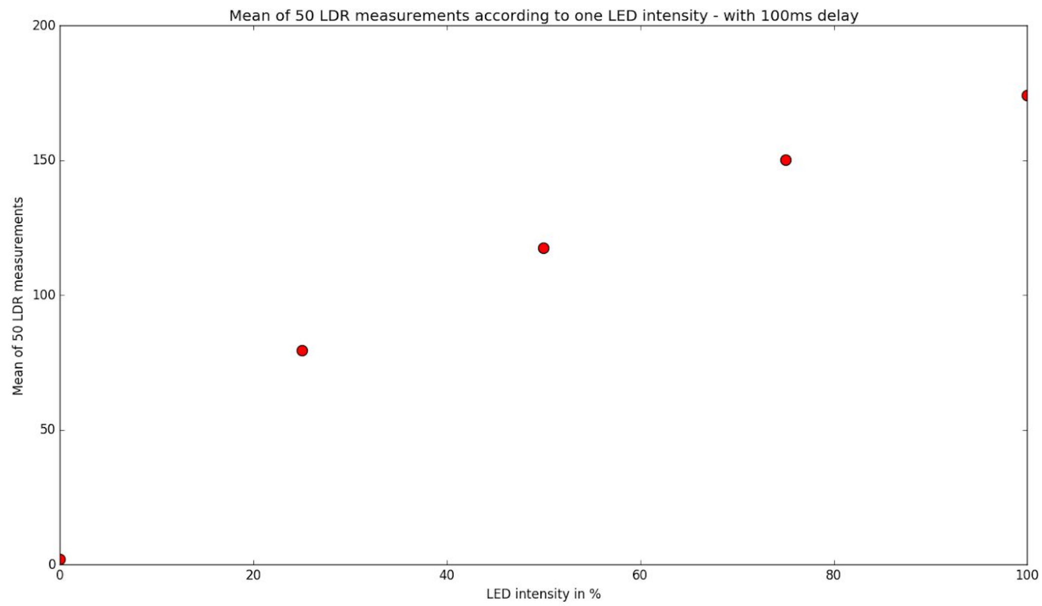
Graph of the biological data - without smoothed :



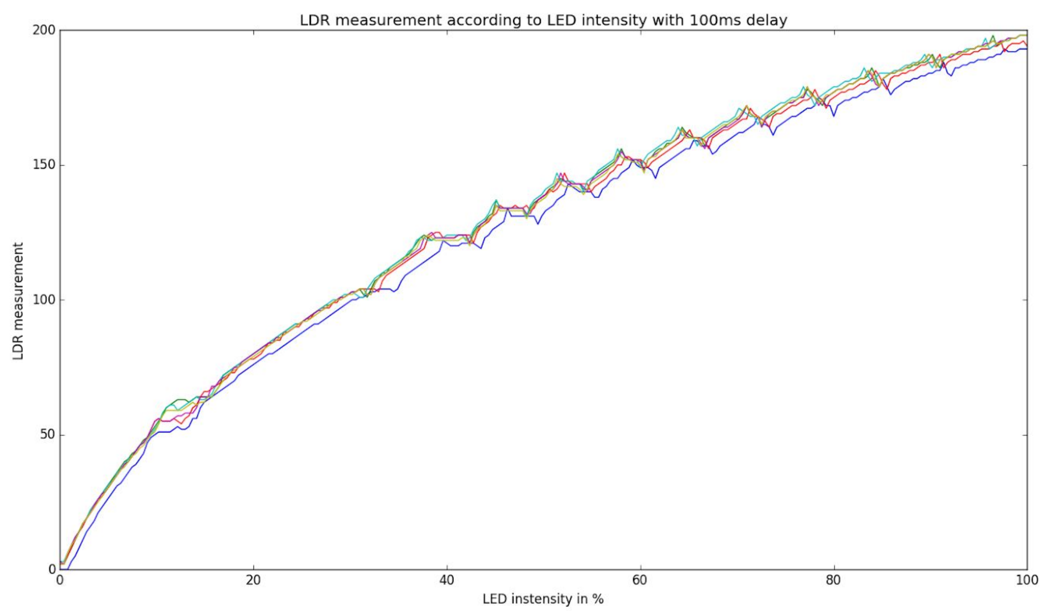
Graph of the biological data - smoothed :



Graph of electronic means for 5 values :



Graph of electronic values for different intensities - at 100ms delay between each value :



Graph of electronic values for different intensities - at 500ms delay between each value :