

PROJET D'INTÉGRATION INNOVATION

PROJET : BRAS ROBOTIQUE À 5 DOF

**PRÉSENTÉ PAR : GUZEL, MORIN, DE CROZEFON,
GONCALVES, COULIBALY**

Dirigé par M.Abdelwahed & M.Ayad

SOMMAIRE

01

INTRODUCTION

02

EQUIPE

03

CAHIER DES CHARGES

04

GESTION DU PROJET

05

TECHNOLOGIE UTILISÉ

06

CONCEPTION MÉCANIQUE

07

ELECTRONIQUE

08

PROGRAMMATION

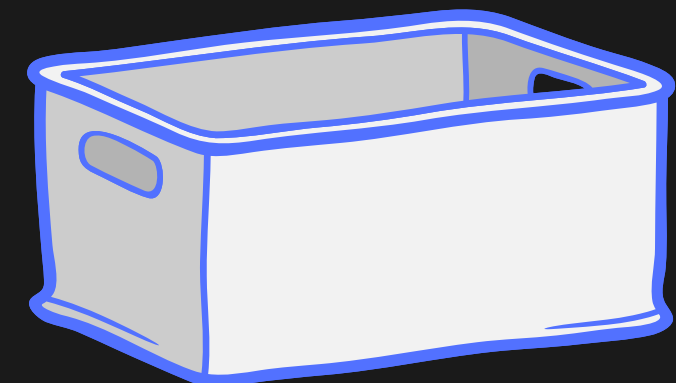
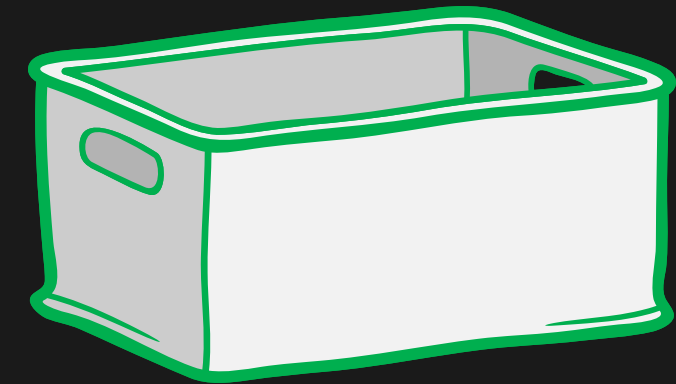
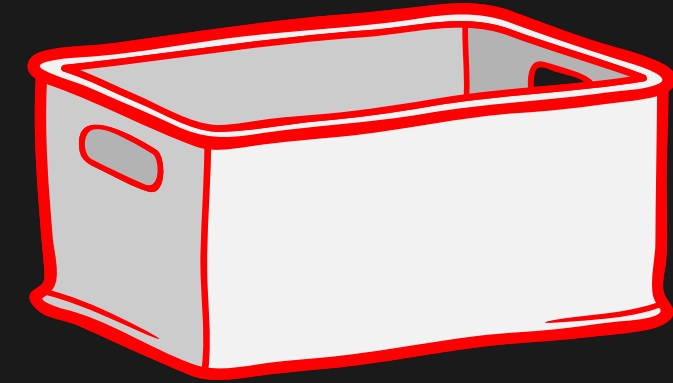
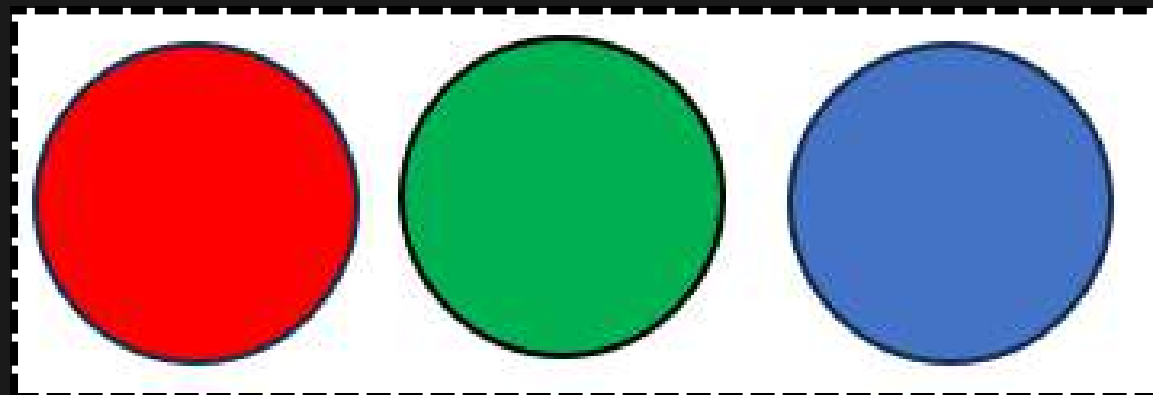
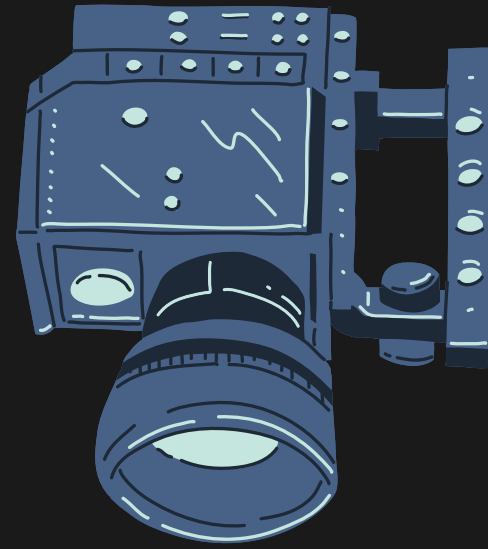
09

BILAN

10

DEMO

INTRODUCTION



EQUIPE



Michael
De Crozefon



Morin
Alexandre



Guzel
Halil



Goncalves
Vicente



Coulibaly
Ahmed

CAHIER DES CHARGES

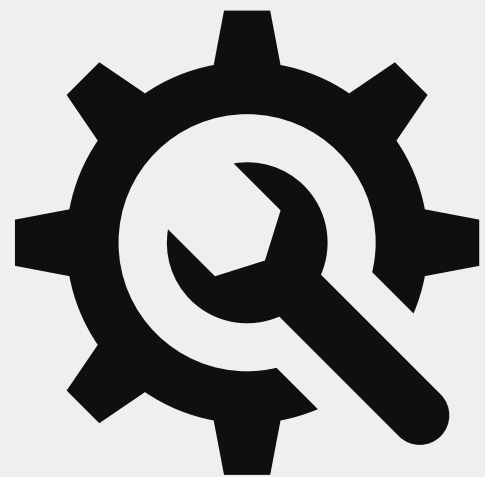
OBJET DE L'ÉTUDE

- FONCTIONNALITÉS DU BRAS ROBOTIQUE
- CONCEPTION MÉCANIQUE
- PARTIE ELECTRONIQUE
- PROGRAMMATION
- RÉOLUTION ET PERSPECTIVES
D'AMÉLIORATION

ANALYSE DES BESOINS



- Réalisation de la cinématique directe et inverse.
- Simulation précise de scénarios "pick and place".
- Intégration d'un système de détection.



- Conception mécanique robuste et légère.
- Choix judicieux de composants électroniques et capteurs précis.
- Programmation structurée et optimisée.

DEVIS

Composant	Quantité	Prix (approximatif)	Commentaires
Arduino Mega	1	30 €	Contrôle central du bras robotique
Raspberry Pi	1	40 €	Gestion des tâches informatiques et de la caméra
Servo-moteurs NEMA 17	5	10 € chacun	Moteurs pour les articulations du bras
Caméra	1	25 €	Capteur visuel pour le système
Matériaux d'impression 3D (PLA)	-	15 €	Estimation pour l'impression 3D en PLA
Contrôleurs/ Shields pour Servo-moteurs	1	50 €	Gestion des servo-moteurs
Fils électriques	-	10 €	Câblage électrique
Alimentation électrique	1	20 €	Source d'alimentation principale
Batteries (si nécessaire)	-	30 €	Pour une alimentation mobile
Gripper avec ventouses	1	40 €	Système de préhension pour le bras

GESTION DU PROJET

RÉPARTITIONS DES TACHES

**Michael
De Crozefon**

Conception 3D
Impression 3D
Rédaction rapport

**Alexandre
Morin**

Programmation
Assemblage robot
Rédaction cahier des
charges

**Halil
Guzel**

Calcul mécanique
Polyvalence
développement
Coordination des idées

**Vicente
Goncalves**

Gestion des ressources
Supervision de
l'impression 3D
Gestion des outils

**Ahmed
Coulibaly**

Soutien aux diverses tâche :

- Programmation
- Assemblage robot
- Rédaction cahier des
charges

PLANIFICATION: TRELLO

bras robotique à 5 DOF ☆ Visible par l'espace de travail 000 Tableau ▾ Power-up Calendrier Power-ups ⚡ Automatisation Filtres HG GR VG

Project Resources ...


Lien

Youtube


Trello

Canva

Thingiverse

 Prusa Slicer

PrusaSlicer
🔗 1




+ Ajouter une carte 📎

A faire ...

Compte rendu

+ Ajouter une carte 📎

En cours ...



Architecture
☰ 🔗 3

Vision

Programmation

Assemblage

MGI / MGD

+ Ajouter une carte 📎

Bloqué ...

+ Ajouter une carte 📎

Terminé ...

Imprimer

Cao

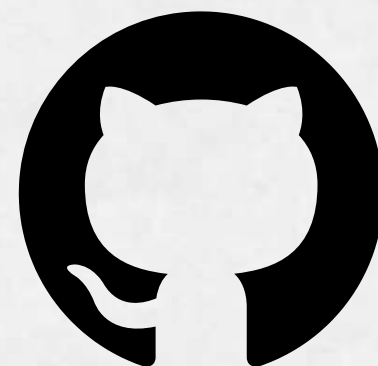
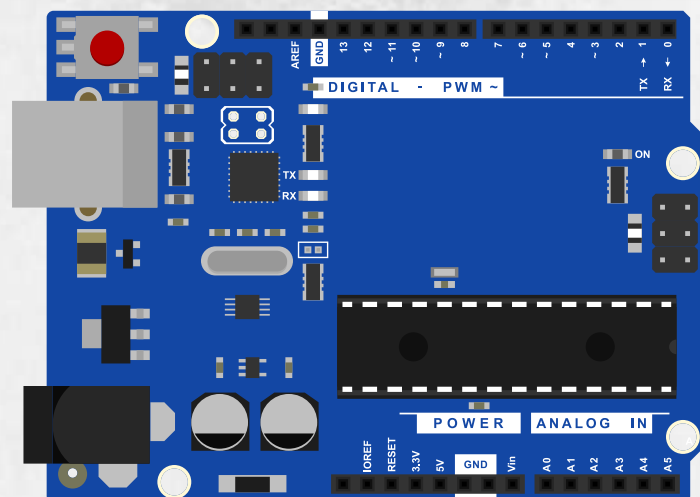
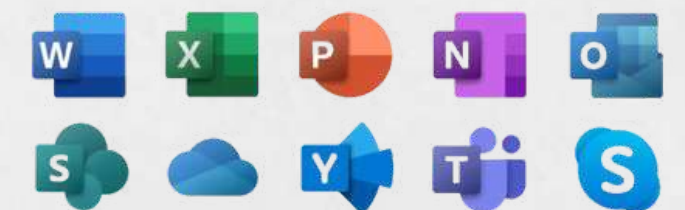
Planification

+ Ajouter une carte 📎

TECHNOLOGIE UTILISÉ

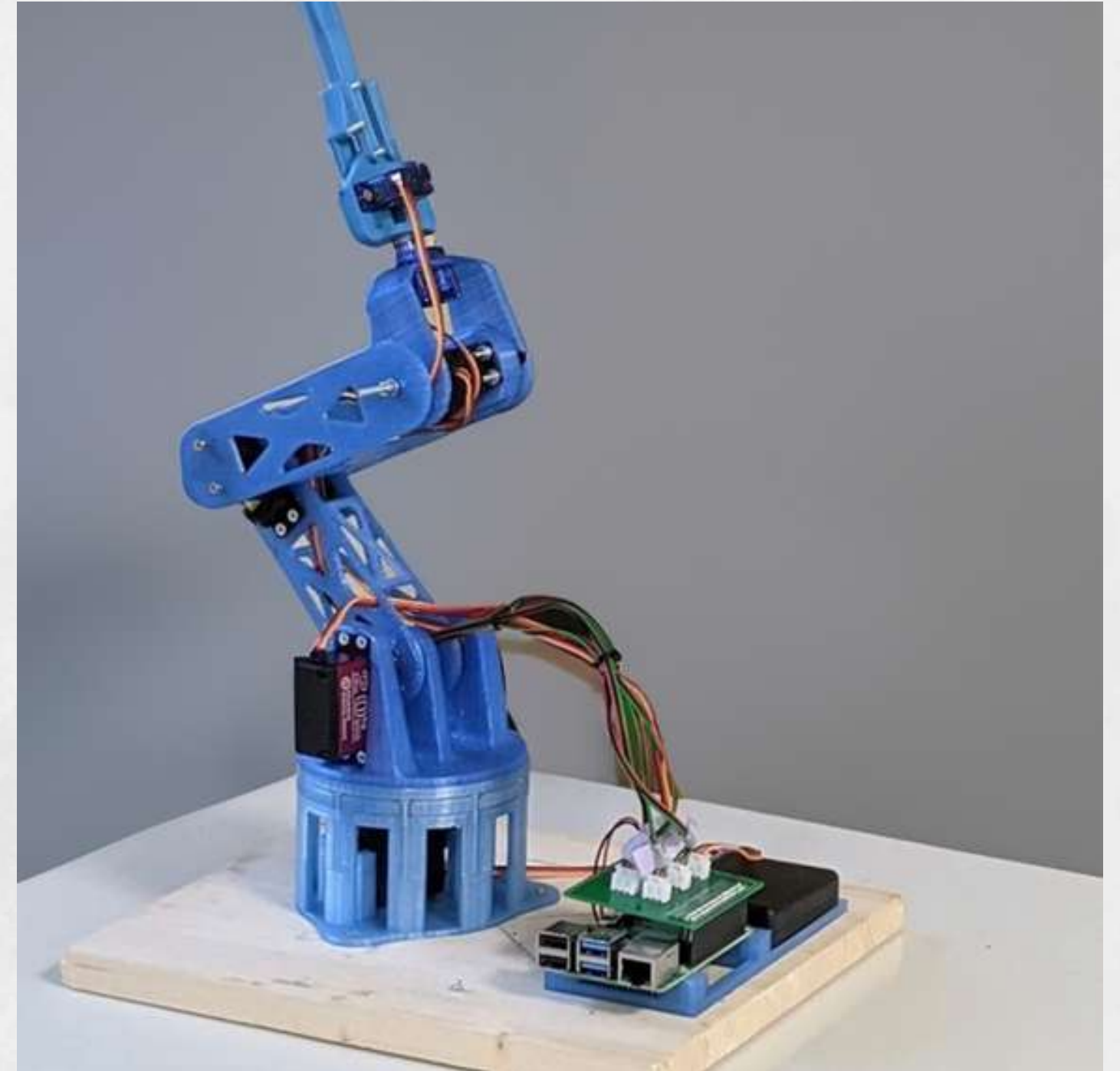


Ressources utilisées



CONCEPTION MECANIQUE

CONCEPTION



ANALYSE DU MÉCANISME

MGD

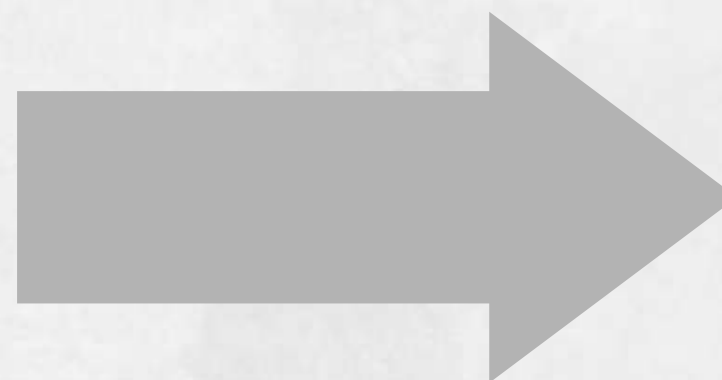
Joint Robot



TCP du robot
Position et
orientation

MGI

TCP du robot
Position et
orientation



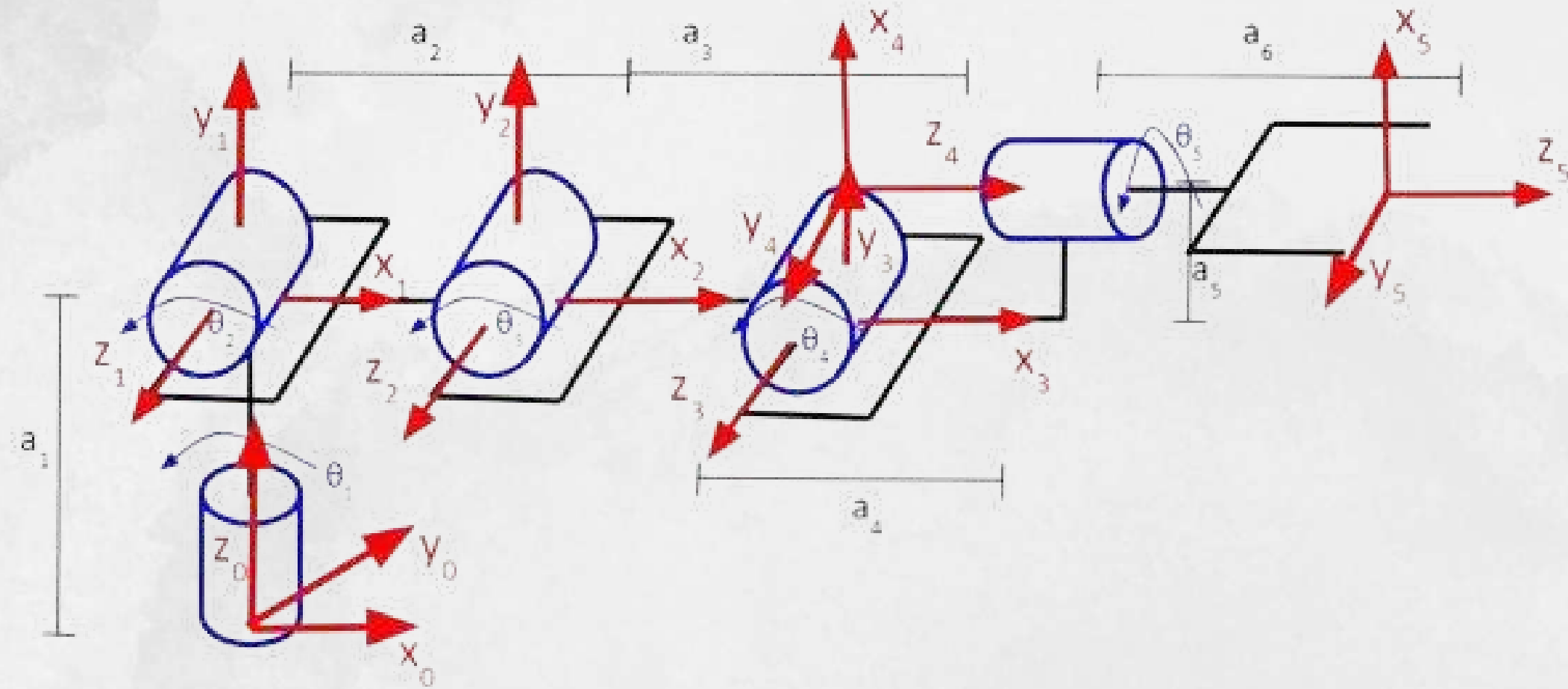
Joint Robot

ANALYSE DU MÉCANISME

4 Règles :

- Z doit être l'axe de rotation
- L'axe X doit être perpendiculaire à Z et Z-1
- Respecter la règle de la main droite
- L'axe X doit intersecter avec Z-1

Schéma cinématique



ANALYSE DU MÉCANISME



Utilisation du
programme python
pour le calcul du
modèle géométrique
inverse

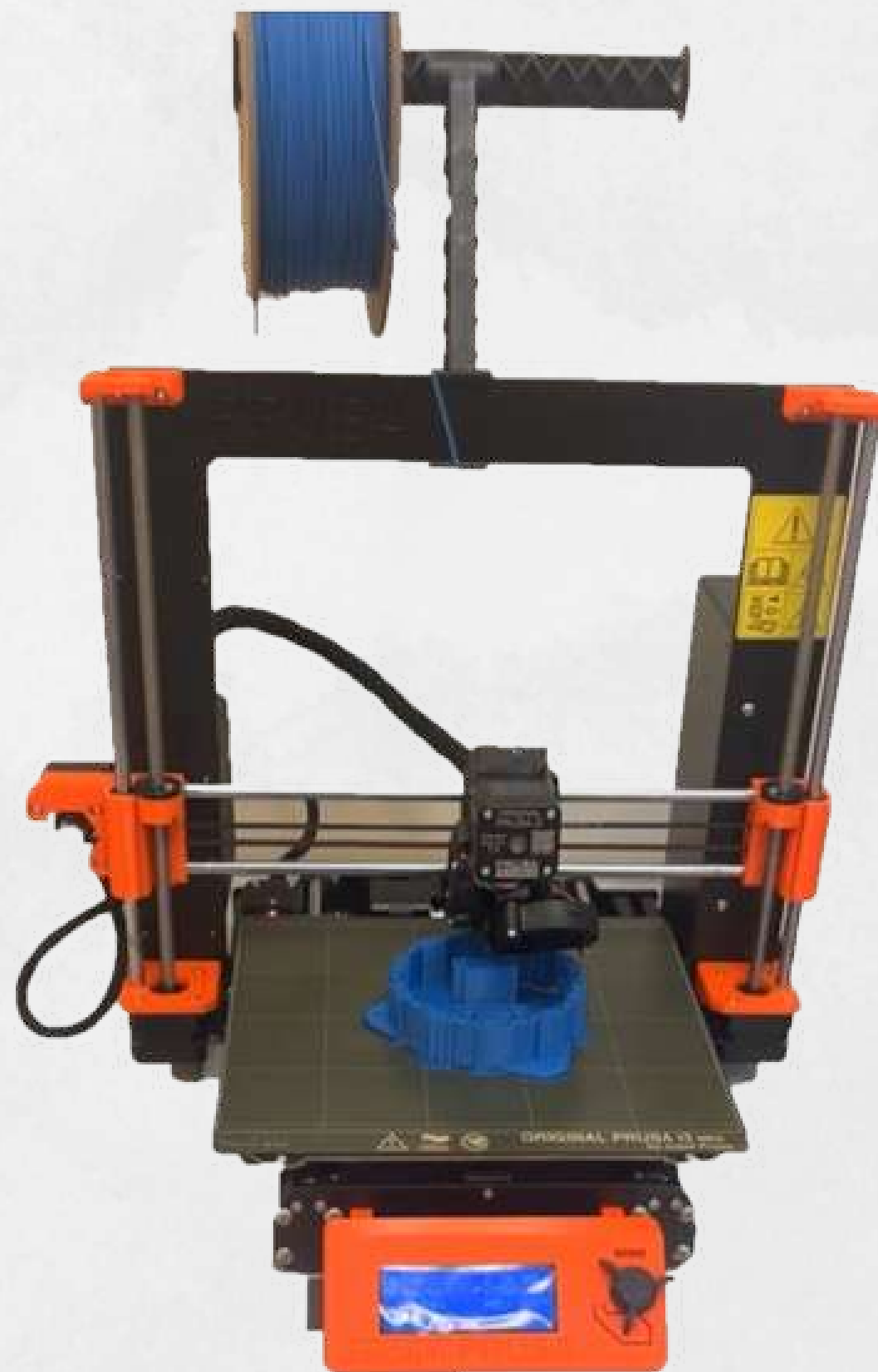
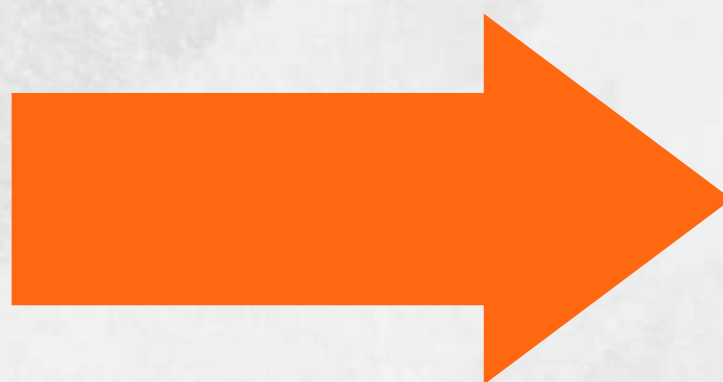
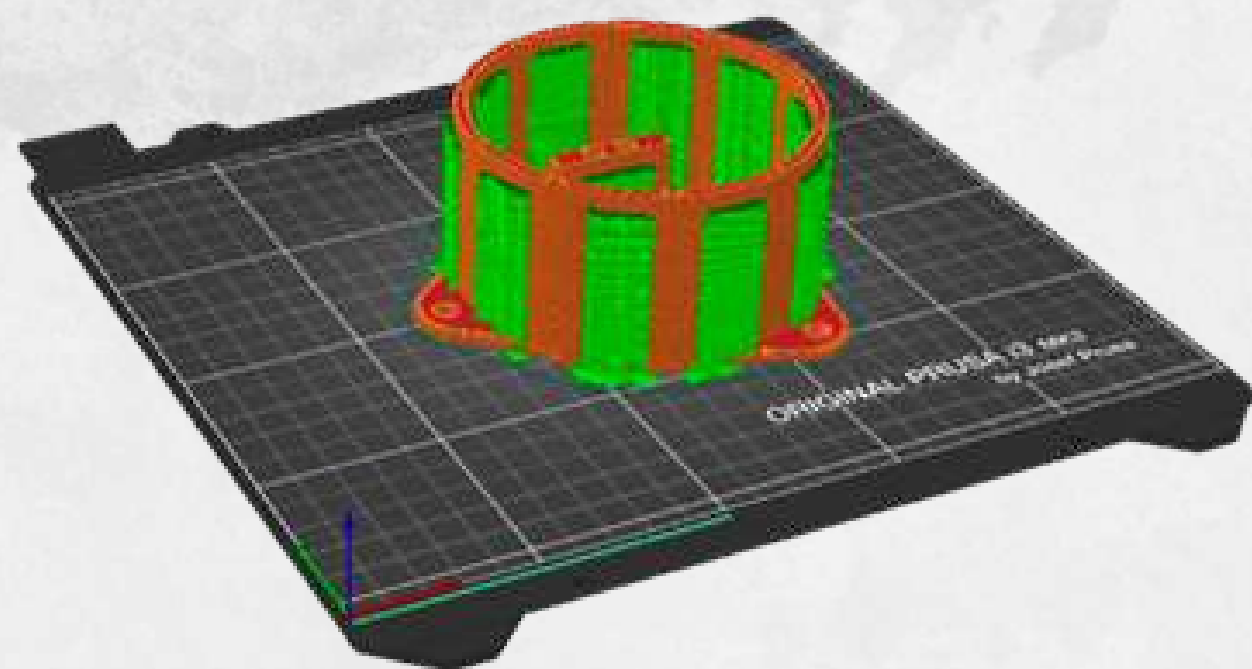
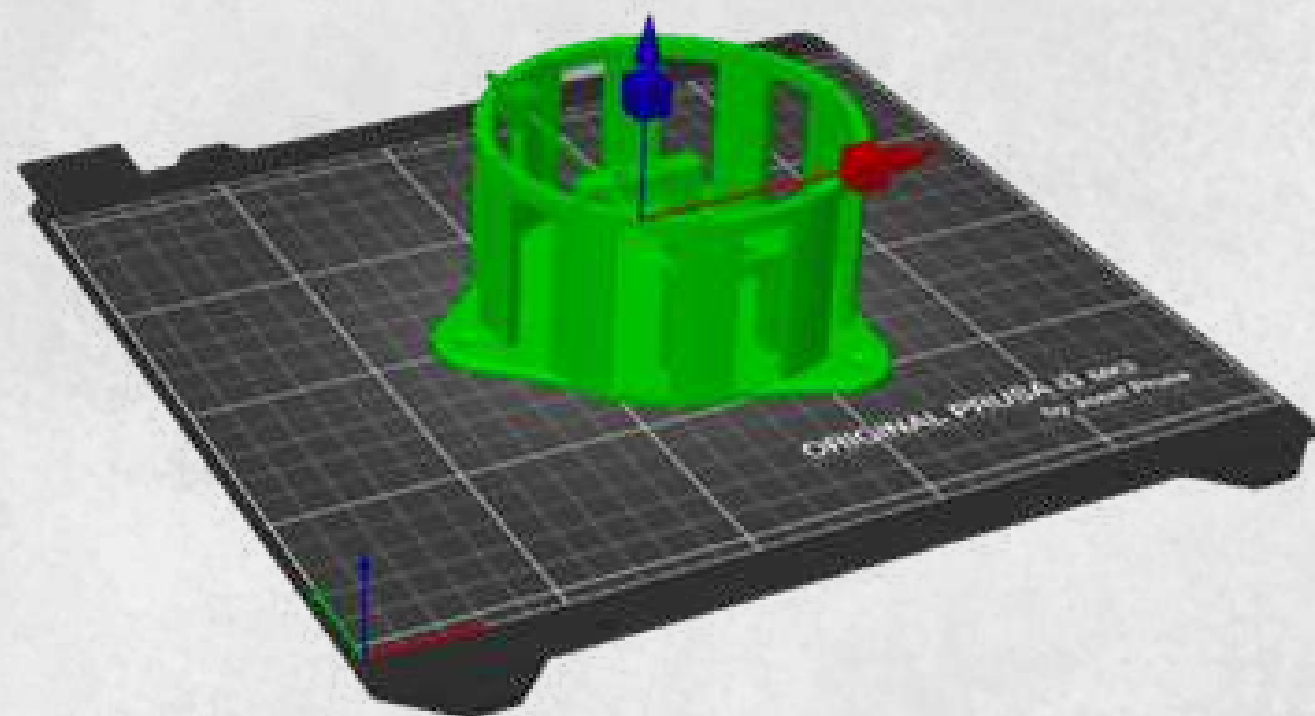
```
Users > mikhaïl > Downloads > Personnel > robot.py > ...
1  import numpy as np
2
3  # Paramètres DH
4  d1 = 10
5  a2 = 15
6  a3 = 15
7  a4 = 8
8
9  # Valeurs des articulations (en radians)
10 theta1 = 0.5
11 theta2 = 0.8
12 theta3 = -0.4
13 theta4 = 1.2
14 theta5 = -0.7
15
16 # Matrices de transformation homogène pour chaque articulation
17 A1 = np.array([
18     [np.cos(theta1), -np.sin(theta1), 0, 0],
19     [np.sin(theta1), np.cos(theta1), 0, 0],
20     [0, 0, 1, d1],
21     [0, 0, 0, 1]
22 ])
23
24 A2 = np.array([
25     [np.cos(theta2), -np.sin(theta2), 0, a2],
26     [0, 0, -1, 0],
27     [np.sin(theta2), np.cos(theta2), 0, 0],
28     [0, 0, 0, 1]
29 ])
30
31 A3 = np.array([
32     [np.cos(theta3), -np.sin(theta3), 0, a3],
33     [0, 0, 1, 0],
34     [-np.sin(theta3), -np.cos(theta3), 0, 0],
35     [0, 0, 0, 1]
36 ])
```

```
37
38 A4 = np.array([
39     [np.cos(theta4), -np.sin(theta4), 0, a4],
40     [0, 0, -1, 0],
41     [np.sin(theta4), np.cos(theta4), 0, 0],
42     [0, 0, 0, 1]
43 ])
44
45 A5 = np.array([
46     [np.cos(theta5), 0, np.sin(theta5), 0],
47     [0, 1, 0, 0],
48     [-np.sin(theta5), 0, np.cos(theta5), 0],
49     [0, 0, 0, 1]
50 ])
51
52 # Matrice totale du MGD
53 T_total = np.dot(np.dot(np.dot(np.dot(A1, A2), A3), A4), A5)
54
55 # Coordonnées du point final
56 x = T_total[0, 3]
57 y = T_total[1, 3]
58 z = T_total[2, 3]
59
60 print("Coordonnées du point final :")
61 print(f"x = {x} cm")
62 print(f"y = {y} cm")
63 print(f"z = {z} cm")
```

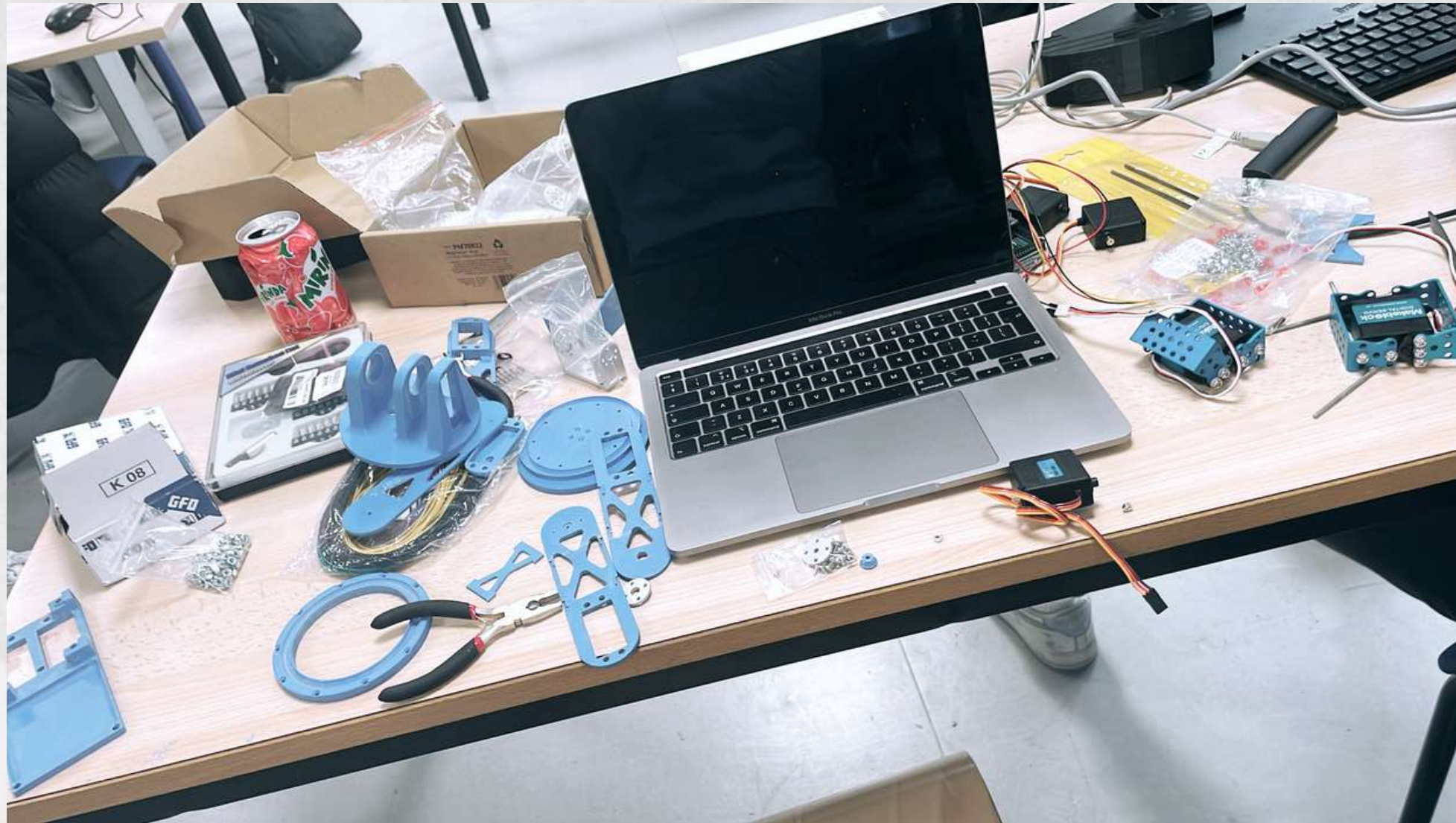
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
/Users/mikhaïl/anaconda3/bin/python /Users/mikhaïl/Downloads/Personnel/robot.py
(base) mikhaïl@MBP-de-Mikhaïl ~ % /Users/mikhaïl/anaconda3/bin/python /Users/mi
/Users/mikhaïl/anaconda3/bin/python /Users/mikhaïl/Downloads/Personnel/robot.py
Coordonnées du point final :
x = 28.333803752586796 cm
y = 11.928908846787504 cm
z = 26.046171076596348 cm
```


IMPRESSION 3D



ASSEMBLAGE

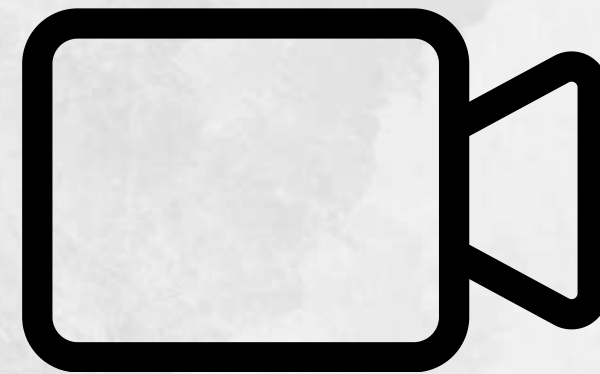
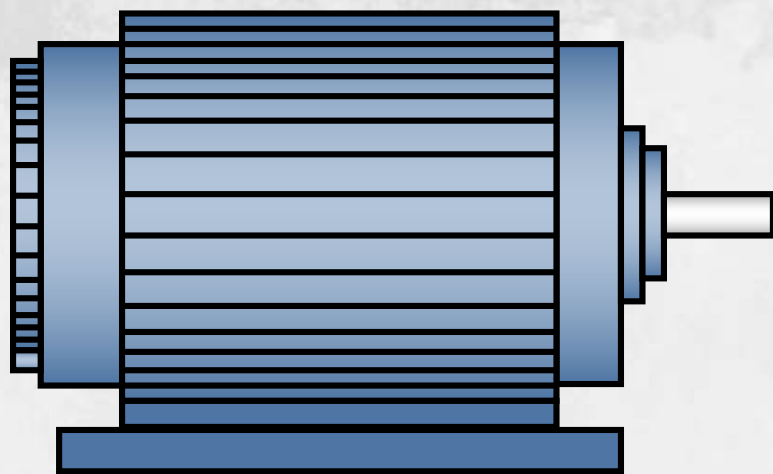


ELECTRONIQUE

CHOIX DES COMPOSANTS

Puissance :

Servo Moteur Makeblock
Servo Moteur Feetech
Servo Moteur SG90

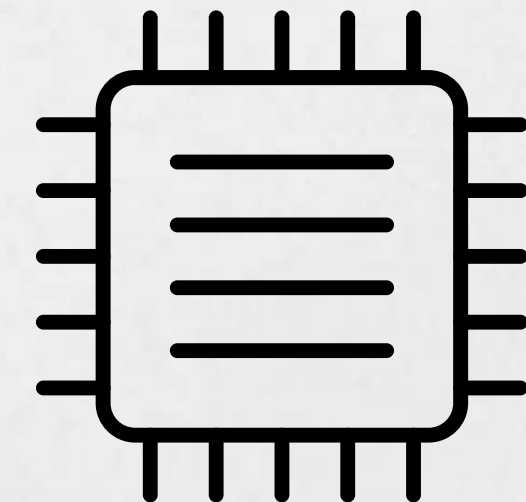


Commande :

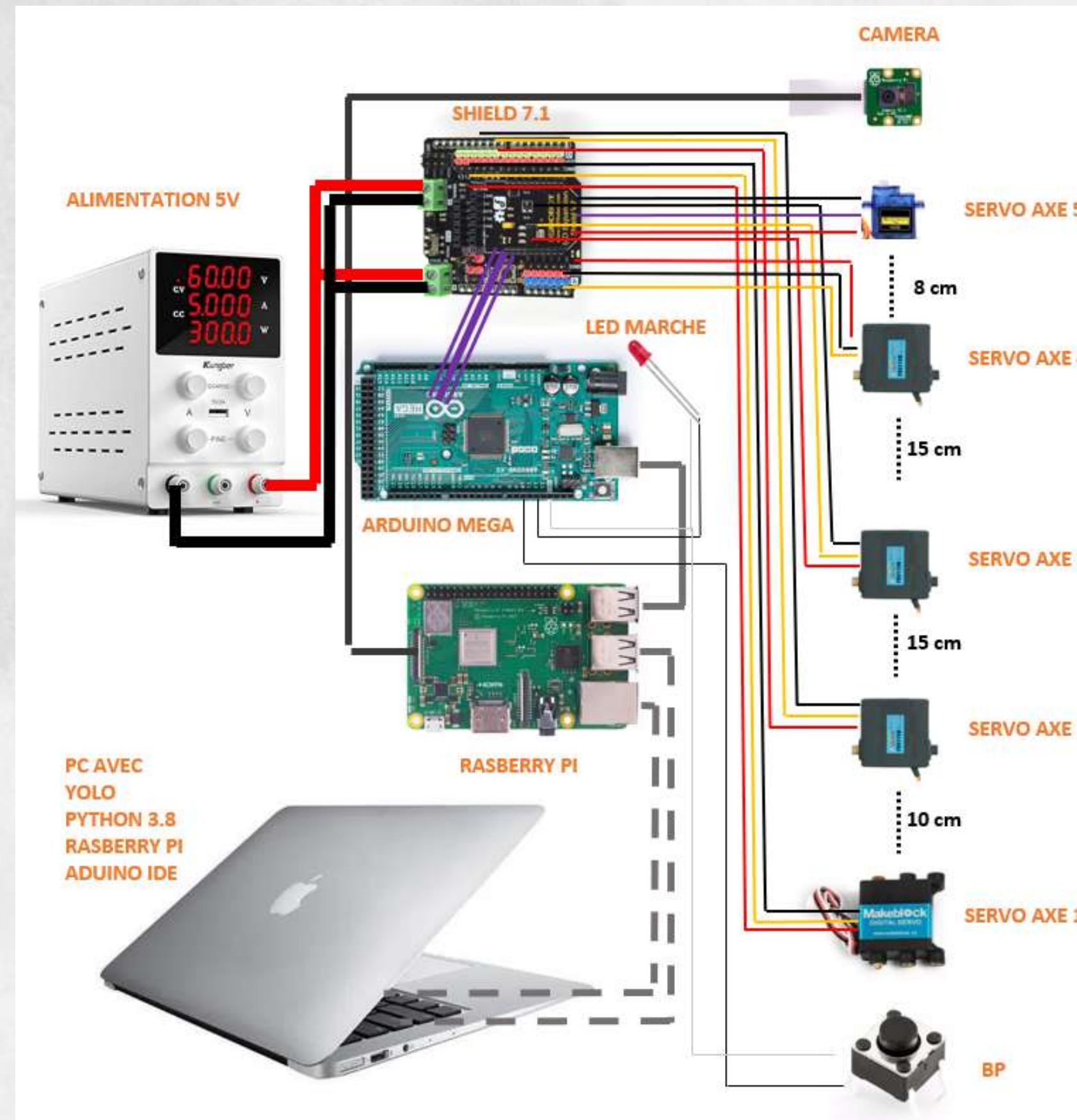
Contrôleur Arduino
Mini-Ordinateur Raspberry Pi
Shield I/O Expansion

Autres :

Caméra OV4657
Alimentation
LEDs



SCHÉMAS DE FONCTIONNEMENT



PROGRAMMATION

5 AXES

3 fonctions dans le programme :

- MoveArm : Lancement du déplacement (X,Y,Z)
- Inverse Cinématique : calcul des angles de la trajectoire en fonction des longueurs des segment du robot
- Loop : récupération des coordonnées du raspberry, analyse des coordonnées, lancement de la trajectoire

STRUCTURE DU PROGRAMME

```
void loop() {  
  // Mouvement du bras robot vers une position spécifique  
  
  // Attendre les coordonnées XYZ depuis le Raspberry  
  if (Serial.available() >= 12) { // Attendre au moins 12 caractères (3 valeurs flottantes séparées par des virgules)  
    String input = Serial.readStringUntil('\n');  
    input.trim();  
    int commaIndex1 = input.indexOf(',');  
    int commaIndex2 = input.indexOf(',', commaIndex1 + 1);  
  
    if (commaIndex1 != -1 && commaIndex2 != -1) {  
      String xStr = input.substring(0, commaIndex1);  
      String yStr = input.substring(commaIndex1 + 1, commaIndex2);  
      String zStr = input.substring(commaIndex2 + 1);  
  
      float x = xStr.toFloat();  
      float y = yStr.toFloat();  
      float z = zStr.toFloat();  
  
      moveArm(x, y, z); // Déplacer le bras robot aux coordonnées  
      delay(700); // Attente de 5 secondes avant de passer à la prochaine commande  
    }  
  }  
}
```

RÉCUPÉRATION DES
COORDONNÉES

ANALYSE DES ÉLÉMENTS
REÇUS

LANCEMENT DE LA
FONCTION MOVEARM

STRUCTURE DU PROGRAMME

```
// Fonction pour déplacer le bras robot en fonction des coordonnées XYZ
void moveArm(float x, float y, float z) {
    // Calcul des angles nécessaires pour chaque servo moteur
    float baseAngle, shoulderAngle, elbowAngle, wristAngle, gripperAngle;
    inverseKinematics(x, y, z, baseAngle, shoulderAngle, elbowAngle, wristAngle, gripperAngle);

    // Déplacement des servomoteurs aux angles calculés
    baseServo.write(baseAngle);
    shoulderServo.write(shoulderAngle);
    elbowServo.write(elbowAngle);
    wristServo.write(wristAngle);
    gripperServo.write(gripperAngle);

    // Attente pour que le bras robot atteigne sa position
    delay(2000);
}
```

RÉCUPÉRATION DES
COORDONNÉES



CALCUL DE LA
CINÉMATIQUE INVERSE
EN FONCTION DES
SEGMENTS DU ROBOT



ECRITURE DE LA
POSITION DANS LES
SERVOS MOTEURS

CAMERA

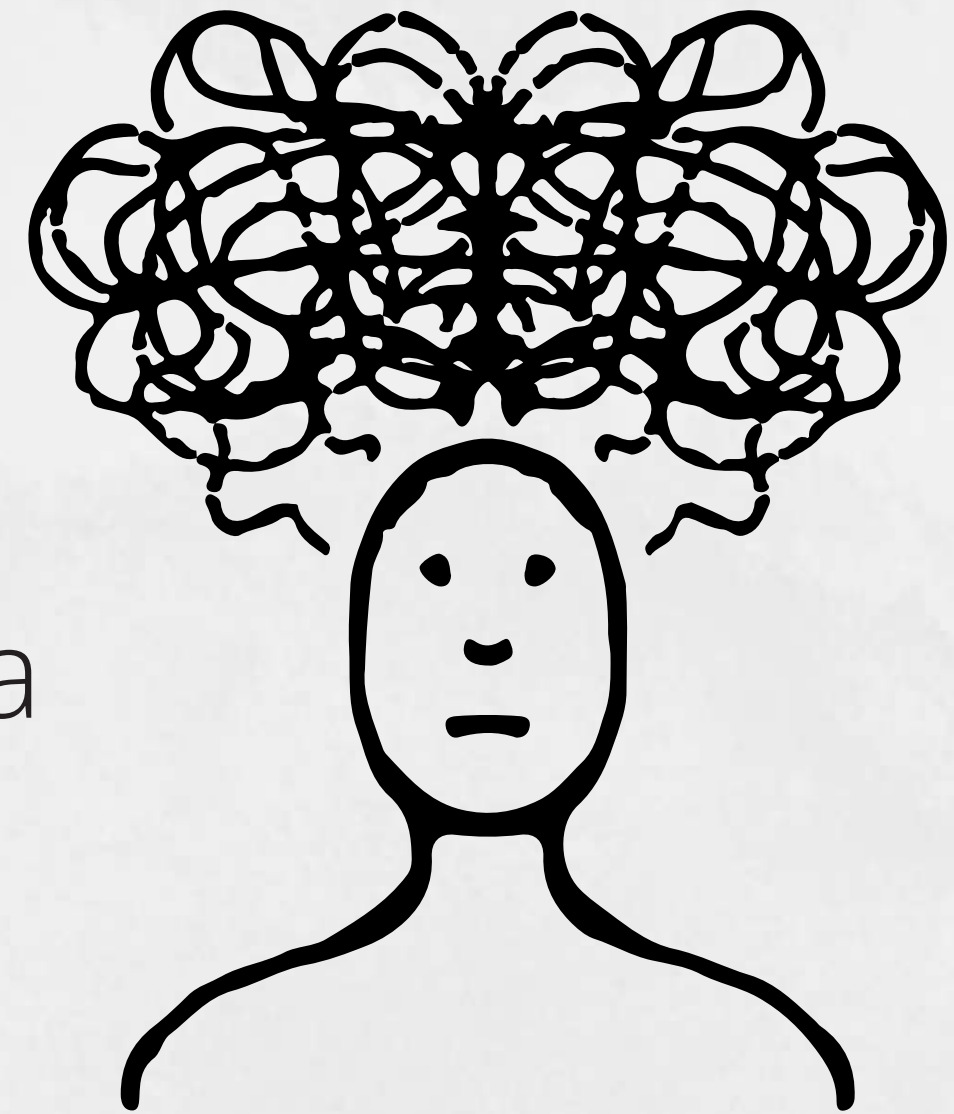


```
Users > mikhail > anaconda3 > envs > 🐍 YoloMaster2.py > ...  
1  from ultralytics import YOLO  
2  from ultralytics.models.yolo.detect.predict import DetectionPredictor  
3  import cv2  
4  
5  model= YOLO("yolov8s.pt")  
6  
7  results=model.predict(source="0", show=True)  
8  
9  print(results)
```

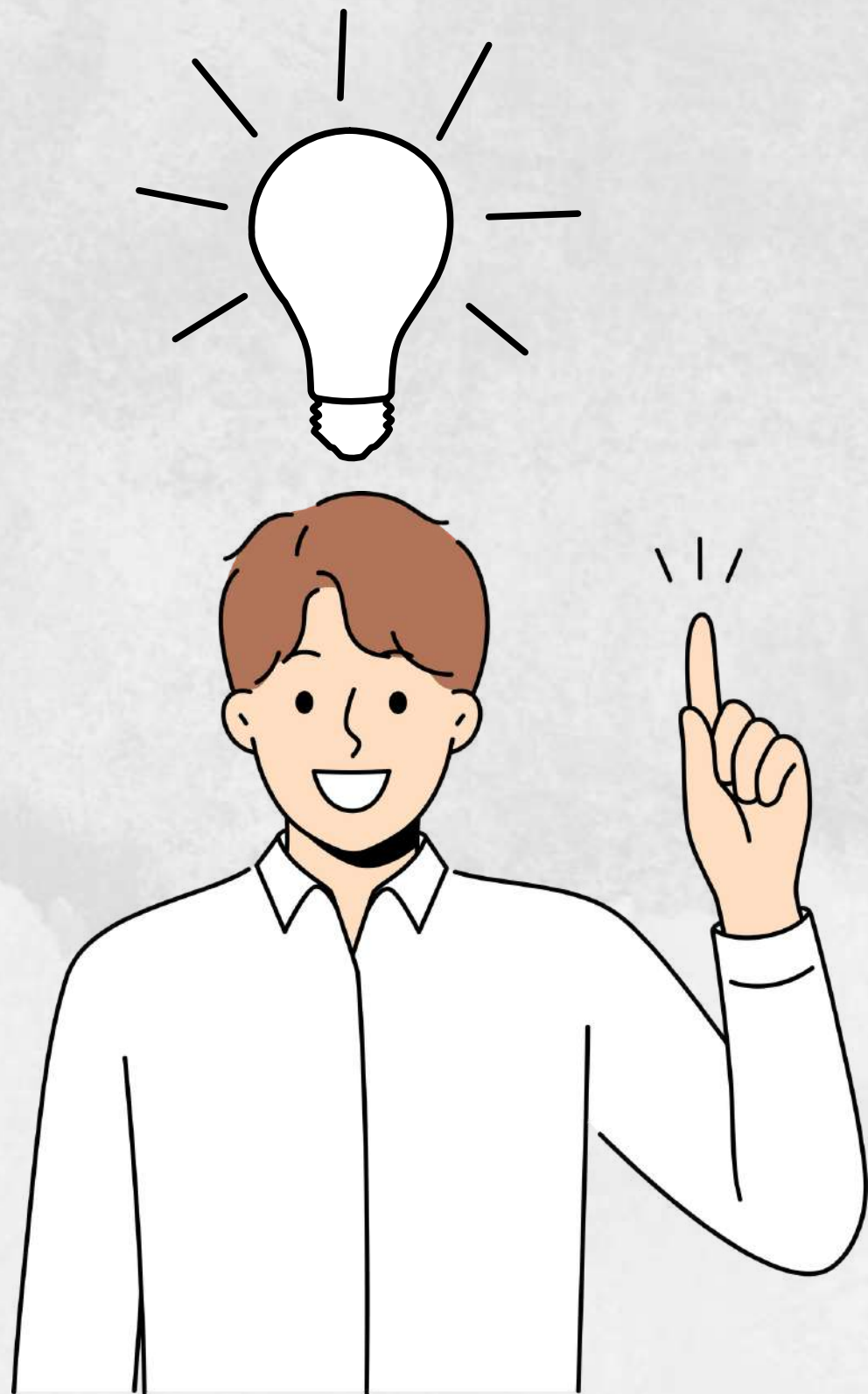
BILAN

PROBLÈMES RENCONTRÉS

- Manque de ressources, de matériel
- Délais assez court
- Problème d'impression 3D
- Problèmes d'assemblage
- Préhenseur électro aimanté absent
- Difficulté à connecter la caméra avec la Raspberry (carte SD flashé)
- Manque de certaines connaissances



AMÉLIORATIONS



- Bras robotique fonctionnel malgré contraintes temporelles.
- Conception mécanique nécessitant attention supplémentaire.
- Investissements dans l'amélioration électronique et logicielle.
- Base solide pour évolutions futures.
- Potentiel d'interface interactive pour contrôle avancé.
- Ouverture vers nouvelles utilisations et expérimentations.

BILAN

Ce projet nous a offert une opportunité précieuse d'explorer et d'approfondir nos connaissances dans le domaine de la robotique. Voici un aperçu des principaux enseignements tirés et des bénéfices que cela nous a apportés

Apprentissages :

- Connaissance en robotique approfondie:
 - Programmation
 - Calcul cinématique
- Base solide pour évolutions futures.
- Compréhension des concepts clés de la robotique.
- Expérience pratique avec manipulation d'un bras robotique.

Apports :

- Amélioration des compétences techniques.
- Prise de décision éclairée pour le choix des outils.
- Renforcement de la collaboration en équipe.
- Élargissement des horizons professionnels.

DEMO

THE END