Date :25/03/2025

Compte-rendu séance 13 Téo Baillot d'Estivaux :

Objectif de la séance :

Pendant cette séance, je devais implémenter le fonctionnement du capteur de pression pour vérifier si le servo moteur qui ferme la pince force ou non et faire en sorte qu'il arrête de forcer s'il force. Finalement on a décidé de plutôt mesurer le courant pour savoir si ce servo moteur force mais on n'a pas le composant qui permet de le faire donc à la place, j'ai décidé que j'allais travailler sur l'automatisation des mouvements de la pince lorsqu'on appuie sur un bouton. L'objectif est d'avoir un bouton qui permettrait d'activer et désactiver une série de mouvements que le bras robotique pourrait faire de façon automatique pour pouvoir déplacer un objet.

Bouton pressoir :

J'ai donc commencé par essayer de faire un code pour gérer le bouton pressoir. L'idée est qu'il y aura deux actions à faire. La première sera celle de contrôler les mouvements de la pince avec les données de l'accéléromètre et du capteur flex, et la deuxième action serait que la pince fasse des mouvements répétitifs de façon indépendante, indépendamment des données de l'accéléromètre et du capteur flex.

J'ai donc fait un premier code qui utilise le bouton poussoir qui permet de passer d'une action à l'autre lorsqu'on clique sur le bouton. Ce code m'a servi de test donc pour le moment les actions une et deux ne font qu'écrire des messages dans le moniteur série :

Premier code pour le bouton :

```
int bouton = 25;  // Pin du bouton poussoir

bool action1Active = false;  // Indique si l'Action 1 est active

bool buttonPressed = false;  // Évite les appuis répétés

unsigned long lastDebounceTime = 0;

const unsigned long debounceDelay = 50;


void setup() {

  pinMode(bouton, INPUT);  // Lecture du bouton en mode INPUT

  Serial.begin(115200);

}


void loop() {
```

```
  int etat = digitalRead(bouton);  // Lire l'état du bouton

  // Gestion du debounce et basculement d'état
  if (etat == HIGH && !buttonPressed && (millis() - lastDebounceTime > debounceDelay)) {
    buttonPressed = true;  // Empêche la répétition tant que le bouton est maintenu
    action1Active = !action1Active;  // Change d'action
    Serial.print("Nouvelle action : ");
    Serial.println(action1Active ? "Action 1" : "Action 2");

    lastDebounceTime = millis();
  }

  if (etat == LOW) {
    buttonPressed = false;  // Autorise un nouvel appui
  }

  // Exécuter l'action correspondante
  if (action1Active) {
    action1();
  } else {
    action2();
  }
}

void action1() {
  Serial.println("Action 1 en cours...");
  delay(100);
}

void action2() {
  Serial.println("Action 2 en cours...");
```

```
  delay(100);
}
```

Réparation de la pince :

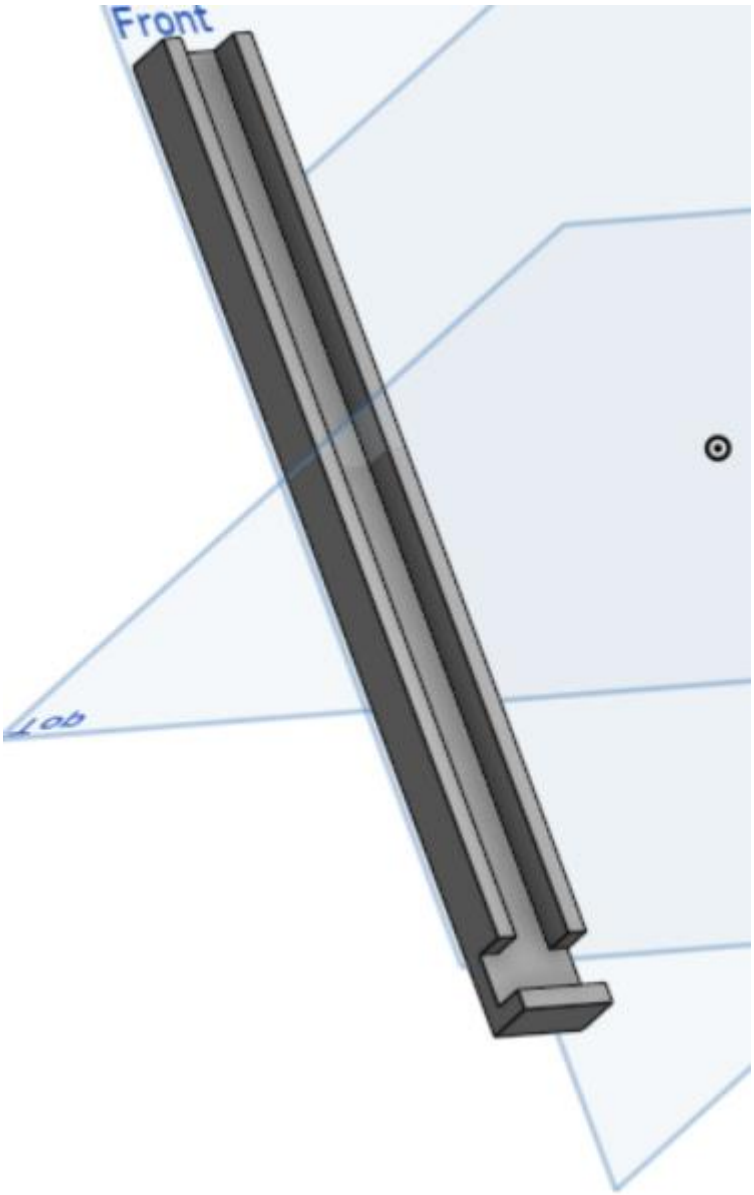
Lors des tests de la dernière séance, j'avais cassé la pince, j'ai donc récupéré les pièces nécessaires qui ont été imprimées et l'ai réparée.

Modélisations 3d :

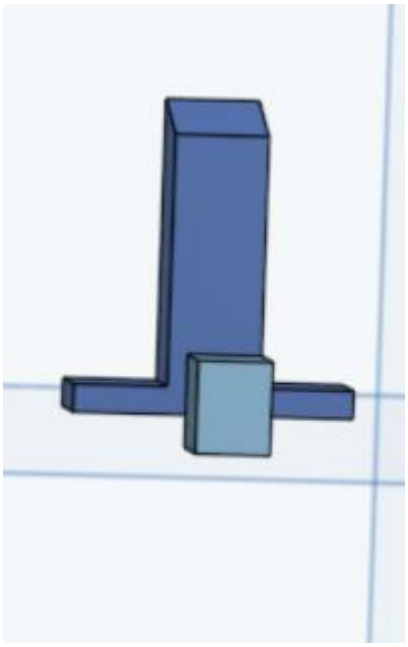
J'ai ensuite travaillé sur la conception de modélisation 3d pour pouvoir faire la démonstration des mouvements automatiques du bras robotique. L'idée ici est de faire une rampe et l'objectif est que le bras robotique attrape un objet en bas de la rampe, le remette en haut de la rampe, et retourne le chercher, ainsi de suite.

Pour ce faire j'ai fait les modélisations suivantes :

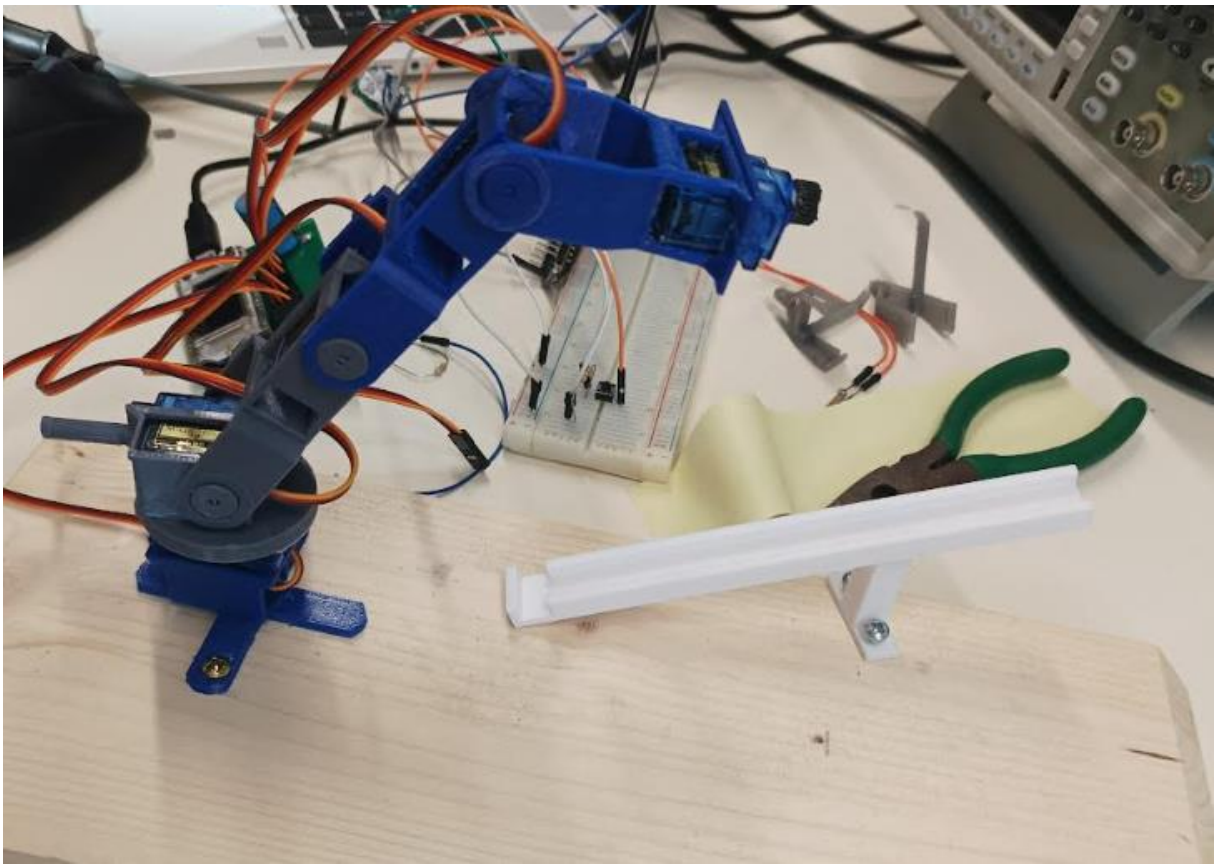
Rampe :


Front

Pied de la rampe et rectangle à attraper :



J'ai ensuite monté cette rampe pour pouvoir faire des tests :



Mise en place du mouvement automatique :

J'ai ensuite voulu prendre des mesures de positions pour les utiliser pour faire l'automatisation des mouvements mais malheureusement un servo-moteur ne bougeait plus sans que je

comprenne pourquoi. J'ai donc passé le reste de la séance a essayé de le faire marcher sans succès. En faisant des mesures au multimètre, il est pourtant bien alimenté en 5V donc le problème doit venir de la PWM.

Objectif de la prochaine séance :

L'objectif de la prochaine séance sera donc de mettre en place les mouvements automatiques du bras robotique lorsqu'on appuie sur le bouton poussoir.