

Rapport de la séance 10

16 Janvier 2022

PORCEL Koralie

Robotique


Pendant les vacances et lors de cette séance, j'ai commencé à tester le code pour pouvoir faire déplacer l'araignée.

1) Redresser l'araignée

Pour commencer, j'ai voulu me familiariser avec la carte ssc-32 en testant comment les servomoteurs bougeaient. J'ai donc créé un code pour faire redresser l'araignée toute seule.

J'ai remarqué que les servomoteurs ne bougeaient pas tous en même temps mais j'ai quand même voulu essayer de la faire se relever (*problème expliqué au point 2*)

Lors du 1^{er} test, l'araignée s'est bien relevé mais lors du 2^{ème} test le côté droit s'est effondré sur lui-même car les servomoteurs n'ont pas bougé en même temps. Je pensais avoir cassé un axe de rotation d'un servomoteur mais finalement non. C'est

Parce que je n'avais pas mis de vis pour tenir  donc la pièce c'était un peu

décroché de l'axe et ça l'avait abîmé et on pouvait tourner le bout de l'araignée sans l'axe avec.



Video youtube de l'araignée qui se relève

La vidéo youtube est après avoir réglé les problèmes de synchronisations.

Suite à cela, je me suis dit qu'il fallait encore que je fasse des tests pour comprendre pourquoi il y avait des problèmes de synchronisation entre les servomoteurs.

2) Problèmes d'alimentation

J'ai remarqué que lorsque j'utilisais trop de servomoteurs en même temps il y avait des problèmes de synchronisations. Ils ne bougeaient pas en même temps et avec une intensité différente. J'ai essayé de comprendre pourquoi ça faisait ça.

-Ce n'était pas à cause de l'alimentation parceque elle donnait en sortie 5V et 30A. Or les servomoteurs TD-8135MG marchent sous 5V et consomment au maximum 200mA. Or $200 * 18 = 3600 \text{ mA} = 3.6\text{A}$ Donc l'ampérage et le voltage ne sont pas un problème.

-Je me suis ensuite demandé si ce n'était pas la carte ssc-32 qui contrôle les servomoteurs. Mais pour un courant continu, le max d'ampérage recommandé est entre 3-5A. Donc la carte ssc-32 n'est pas le problème.

Grâce à mon professeur, j'ai compris que le problème d'alimentation venait des petits cables qui relier l'alimentation avec la carte ssc-32. Il m'a donc donné un autre câble plus épais et ça à régler les problèmes de désynchronisations et d'intensité du mouvement.

Cependant, encore en ce moment, lorsque je mets de l'alimentation sur l'araignée pendant quelques secondes, les servomoteurs peuvent bouger un peu aléatoirement ce qui est bizarre.

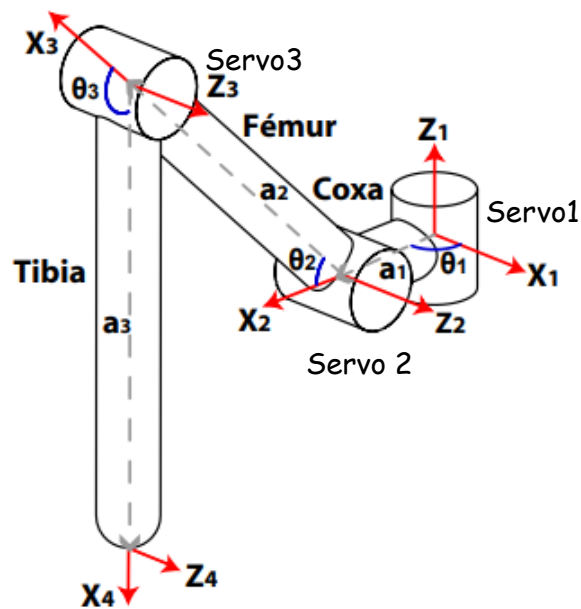
3) Pièces cassées

J'ai cassé deux fois une pièce de l'araignée. La première fois parce qu'elle est tombé de 50 cm et à cause du choque et du poids de l'araignée, la pièce s'est cassé. La seconde fois parce que lorsque je mets du courant les pièces bouge pendant quelques secondes bizarrement et donc lorsque la patte a voulu faire l'action que je lui ai demandée, comme elle n'était pas aux bons endroits, a touché la patte d'à coter et s'est fissuré.

4) Déplacement

Je me suis renseignée sur comment faire pour déplacer l'araignée.

Pour une marche tripode, il faut toujours avoir trois points statiques sur le sol. J'ai trouvé beaucoup de travaux avec des équations sur comment faire déplacer l'araignée.



1/ Image représentative d'une patte

Le déplacement se fait en 3 étapes. On appelle bloc 1 les pattes 1, 3 et 5 et bloc 2 les pattes 2, 4 et 6 :

-on lève le servo2 du bloc 1.

-On fait bouger les servo3 vers l'avant du bloc 1 en même temps que les servo3 du bloc 2 vers l'arrière. Sauf que pour le bloc 2 il faut aussi faire bouger en même temps les servo1 et servo2. Puisque les servomoteurs doivent toujours rester au sol tout en faisant un déplacement fluide. Il faut donc bien ajuster les angles des servo1 et 2.

-Recommencer en levant cette fois le servo2 du bloc 2 et en baissant le servo2 du bloc 1.

Nino m'avait conseillé d'utiliser Isaac Sim qui est un simulateur très complet de robot créer par Nvidia. J'ai essayé de le faire fonctionner mais malheureusement pour pouvoir l'utiliser il faut une carte Nvidia rtx ce que je ne possède pas. Ce simulateur m'aurait permis d'optimiser les déplacements en fonction du terrain.

J'ai commencé à faire un code à la main pour tester le déplacement de l'araignée.



Vidéo youtube du déplacement de l'araignée

5) Code

J'ai regroupé et amélioré les codes que j'avais faits pour tester les servomoteurs dans une classe *Moving*

Dans cette classe il y a 3 types de fonctions :

- Les fonctions *bouger* et *bougerListe*: ces fonctions ont en entrée une liste de pins et une position ou une liste de position. Pour chaque pin de la liste pin, la fonction commande aux servomoteurs d'aller à la position voulue en utilisant *Serial.print*. Explication de comment marche la carte ssc-32 pour contrôler les servomoteurs dans le rapport de séance 8.

```
void Moving::bouger(int liste[],int positio, int taille){
    for(int i=0; i<taille; i++){
        Serial.print("#");
        Serial.print(liste[i]);
        Serial.print(" P");
        Serial.print(angle(liste[i],positio)); // on utilise angle pour que le déplacement soit symetrique
    }
    Serial.print(" T");
    Serial.println(500);
}
```

- La fonction *angle* qui permet de faire avancer l'araignée symétriquement. Sinon, pour le servomoteur le plus proche de la base (celui qui fait tourner horizontalement) entre le côté droit et le côté gauche ne tourne pas dans le

même sens.

```
// Permet de faire tourner les servomoteurs de facon symetrique
double Moving::angle(int pin ,int positio){
    if (pin==0 or pin==4 or pin==8){
        return (3000-positio); //renvoie la symetrique par rapport à 1500);
    }
    else{return positio;}
}
```

- Les fonctions initialisation et avancer qui utilise les fonctions bouger et bougerListe pour faire avancer l'araignée avec des déplacements prédéfinis.

```
//Fonction qui permet de faire le premier test de déplacement de l'araignée
void Moving::avancer(){
    bouger (pinMilieu1,1800,TAILLE3PIN);
    delay(100);
    int a[6] = {1200,1800,1200,1800,1200,1800};
    bougerListe (pinDevant,a,TAILLE6PIN);
    delay(100);
    bouger (pinMilieu1,1500,TAILLE3PIN);
    delay(100);
}
```

2/ Début du code avancer

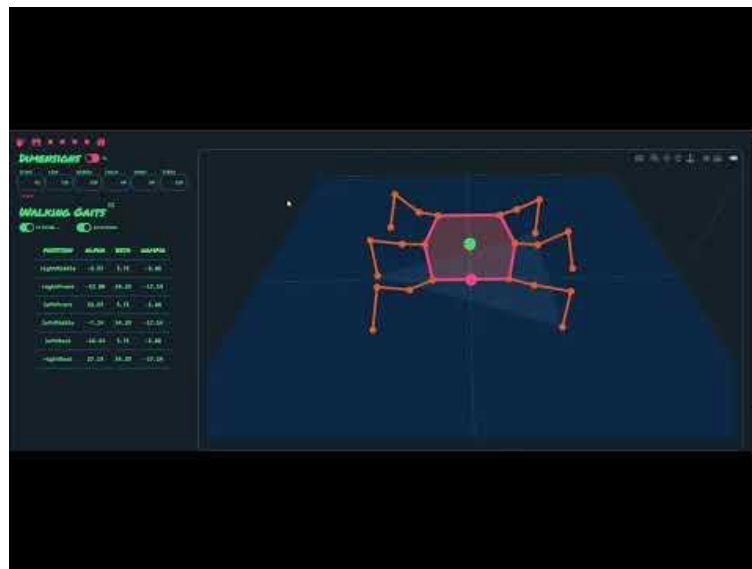
Le code complet est sur [GitHub](#)

J'ai eu des problèmes pour utiliser une liste en entrée d'une fonction dans arduino. En effet, pour les fonction bouger, j'utilise une liste avec tous les pins que je dois bouger simultanément. Or pour parcourir la liste il me faut la taille de cette liste. Sur internet ils me disaient d'utiliser : `int size = sizeof (myarray) / sizeof(int) ;`

Sauf que cela ne fonctionnait pas dans la fonction. En effet quand un tableau devient un paramètre d'une fonction, `sizeof` correspond à la taille du pointeur de la liste et non à la taille de la liste en elle-même. Pour pallier ce problème, j'ai rajouté un paramètre aux fonctions bouger et bougerListe qui correspond à la taille de la liste que j'utilise. Pour ne pas utiliser trop de mémoire et ne pas avoir des chiffres de partout, je les définit dans des références : `#define`

6) Simulateur

J'ai trouvé sur internet un simulateur de mouvement de l'araignée. Cela m'a aidé à comprendre comment le mouvement de l'araignée fonctionne.



Vidéo Youtube du simulateur