Projet robotique autonome : Poppy

Rapport

Nina Docteur, Alexis Juven, Vincent Leconte

ENSEIRB-MATMECA - ENSC

Troisième année option Robotique

13 mars 2019

Tuteur de projet : Thibault Desprez

Table des matières

Table des figures

1 Introduction

Ce projet s'articule autour du projet Poppy, un projet Open Source visant à développer des robots faciles à construire et à utiliser dans le cadre éducatif mais aussi artistique et scientifique. Plusieurs kits basés sur ce projet ont été développés, dont le kit ErgoJr qui permet de construire son propre bras robotique à partir de moteurs, de pièces imprimées en 3D et d'une carte Raspberry. Afin d'agrandir la collection de robots Poppy et de montrer un exemple pratique de modification à partir d'un kit de base, le but de ce projet est de construire un nouveau robot qui soit capable de se déplacer, avec pour base 3 kits ErgoJr. Cela permettrait aux enseignants possédant des kits ErgoJr de fabriquer une toute nouvelle créature sans avoir à acheter un kit entier.

Un état de l'art a été réalisé au début du projet. Ceci a été suivi de plusieurs phases de création du robot. Une première partie du temps a été consacrée à la réflexion sur la structure et à sa création. Ce qui a impliqué une modélisation des pièces suivie de leur impression 3D. Il a ensuite été réalisé la partie mathématique du sujet, modèle géométrique direct et indirect. A la suite de cela il a été possible de tester les méthodes de mise en mouvement d'une patte. S'en sont suivie la mise en place et le test de différents modes de déplacements. Enfin, il a fallu rendre le code intégrable facilement à l'environnement logiciel de la plateforme Poppy. Le travail a été parallélisé avec la rédaction d'un guide permettant de reproduire le robot.

2 Organisation

Le projet a débuté par une phase d'état de l'art qui a duré de fin octobre à fin novembre. A la fin de cette phase nous avons rendu notre état de l'art du projet au client et aux encadrants.

Trois séances de 3 heures de travail ont ensuite eu lieu sur le mois de décembre. Celles-ci nous ont servi à prendre en main un ErgoJr (construction, installation et manipulation). Nous en avons aussi profité pour suivre une formation à l'impression 3D et essayer d'imprimer une petite pièce de test, afin d'appliquer notre connaissance nouvellement acquise, ce qui fût concluant. Le but était d'arriver préparé pour le mois de janvier afin de pouvoir travailler directement sur le coeur du projet, la création d'une nouvelle créature.

Durant la période des fêtes de fin décembre début janvier nous avons établi un succinct cahier des charges (disponible en annexe 3) visant à être en accord avec le client sur les objectifs et attentes du projet. Nous avons en même temps établi un planning du mois de janvier présenté sur la figure 1.

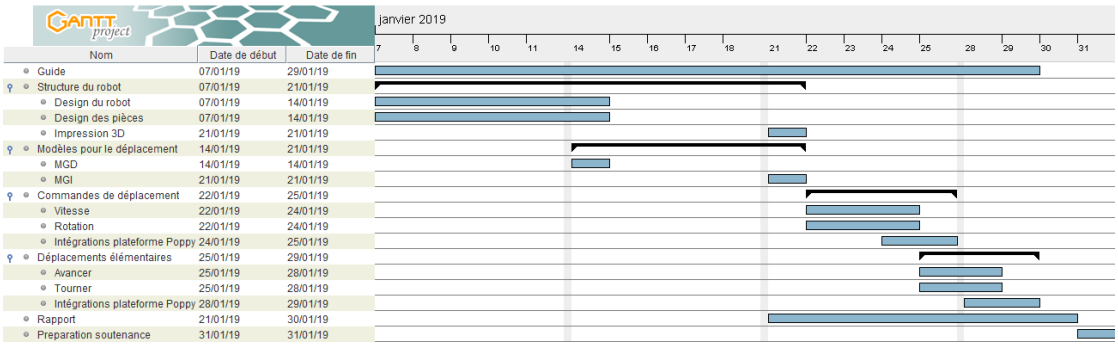

FIGURE 1 – Planning du mois de janvier.

Ce planning était très serré du fait du temps restant et du travail à réaliser. Le travail réalisé a principalement été concentré sur les deux dernières semaines. Néanmoins nous avons atteint

l'objectif principal qui était de construire un robot pouvant se déplacer et de l'intégrer dans la plateforme Poppy, même si certains aspects n'ont pas pu être traités tels que la commande en vitesse, l'intégration d'une visualisation de la créature et la production d'une activité de prise en main du robot.

3 Etat de l'art

Notre état de l'art s'est découpé en 3 grandes parties. La première visait à faire un tour d'horizon sur la robotique pédagogique. En effet, il était nécessaire pour nous de comprendre les besoins qui y sont associés et comment y répondre en regardant comment ils sont traités actuellement. Nous avons pu alors nous rendre compte que la robotique pédagogique vise a être utilisé par des personnes de tout âge et de tout niveau scolaire. Les personnes n'ayant pas forcément de compétences en informatique ou robotique, le robot doit être protégé logiciellement afin que les utilisateurs ne donnent pas d'ordre au robot qui puisse endommager celui-ci. Enfin, des langages appropriés selon le niveau des élèves doivent être utilisés, comme les langages de programmation par blocs pour les débutants.

La deuxième partie de l'état de l'art concernait les différentes formes que peuvent prendre un robot capable de se déplacer. Nous avons trouvé qu'il en existait plusieurs types et que chacun avait des caractéristiques particulières qui pouvaient avoir une influence sur sa capacité à se déplacer sur certains types de terrain. Ainsi, grâce à cela, nous avons pu choisir quelle type de forme allait avoir notre robot. Nous avons donc décidé que notre robot serait un robot à pattes car c'est un type de déplacement qui permet de se déplacer sur des sols irréguliers et qui est intéressant pour l'apprentissage.

Enfin, la dernière partie de notre état de l'art étudiait les différentes possibilités d'interaction entre le robot ErgoJr (plateforme de base du projet) et son environnement. Ces interactions nécessitent un ensemble de capteurs et d'actionneurs qui permettent de capter l'environnement et afin de pouvoir réaliser des actions en conséquence. Notre robot étant équipé d'une carte Raspberry, il peut posséder un microphone ou une enceinte, il est également équipé d'une caméra. De plus, ses moteurs permettent de pouvoir se mouvoir dans l'espace et d'émettre de la lumière. Cela donne à l'utilisateur une kyrielle de possibilité d'interactions que nous n'avons pas exploité ici pour les besoins de notre projet.

L'état de l'art est disponible en annexe 4 de ce rapport.

4 Structure du robot

A la suite de l'état de l'art et en tenant compte des contraintes et avantages du projet (en particulier l'utilisation de 3 kits ErgoJr) il a été décidé de réaliser un robot à 4 pattes avec un bras ErgoJr sur le dessus.

Ce choix permettait d'assurer la stabilité du robot relativement facilement en évitant d'avoir beaucoup de matériel électronique à rajouter par rapport à ce qui était fourni dans les 3 kits ErgoJr, en particulier au niveau électronique. De plus, cela permettait de potentiellement réutiliser du code déjà développé pour les bras ErgoJr sur le bras fixé au-dessus de la structure.

L'ensemble des modèles 3D modélisés est disponible sur le GitHub du projet fourni en annexe 1.

4.1 Les pattes

Pour la forme des pattes nous avons vu que le metabot se rapprochait de ce que l'on souhaitait faire en terme de robot, nous sommes donc parti de la forme des pattes du metabot. Elles avaient

également l'avantage de permettre de réutiliser des éléments déjà présents dans les kits ErgoJr. Les pattes complètes sont présentées dans la figure 2.

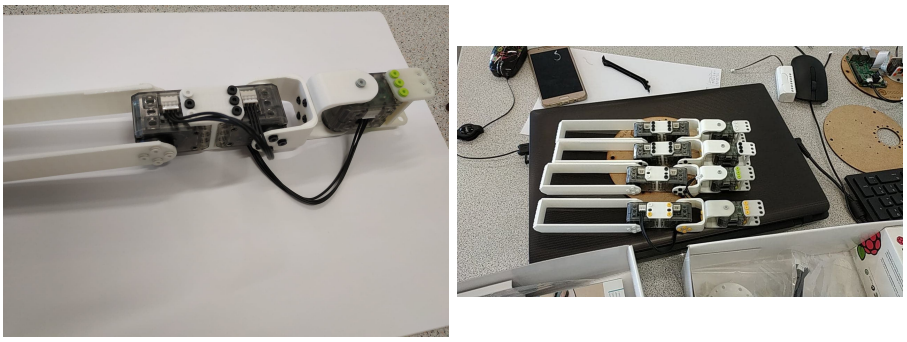

FIGURE 2 – Patte complète

Les pattes ont nécessité la création de deux nouvelles pièces non présentes dans les kits ErgoJr que sont un H long et un pied (Figure 3). Pour le pied nous sommes partis sur quelque chose de simpliste mais qui remplit sa fonction compte tenu du temps que nous avions à disposition.

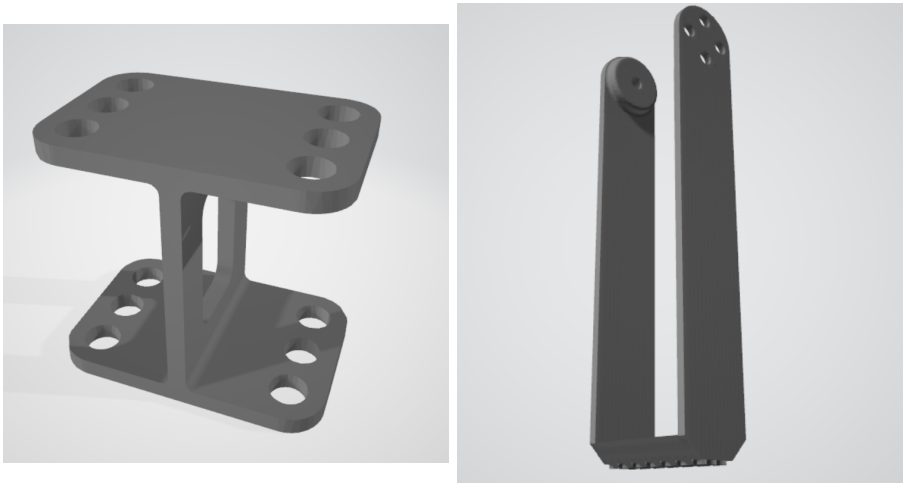

FIGURE 3 – H long (gauche), pied (droite).

Les 3 moteurs permettent d'obtenir 3 degrés de liberté et de pouvoir imposer n'importe quelle position de l'extémité de la patte dans l'espace (exemple de position de la patte figure 4).

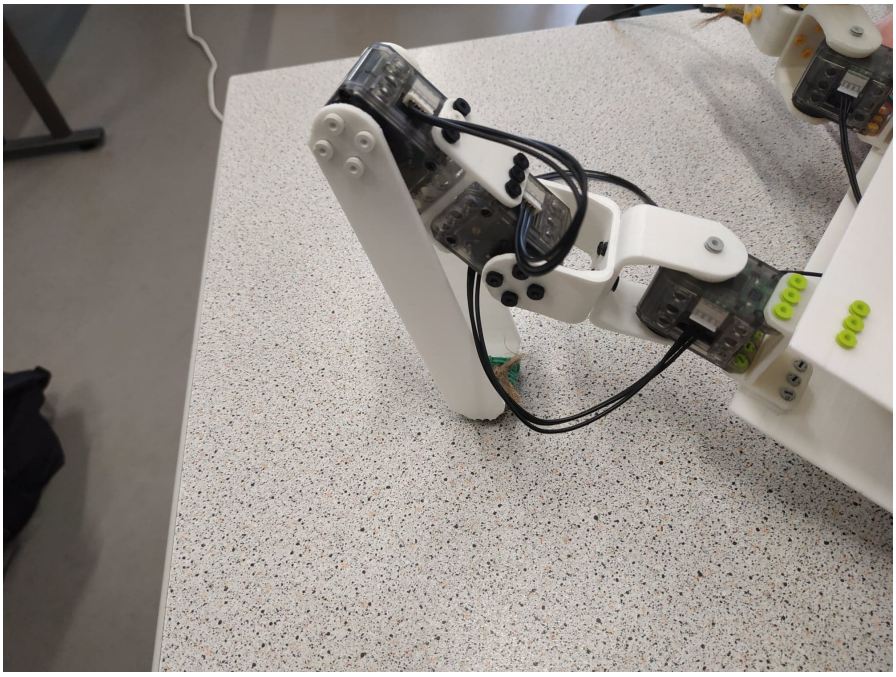

FIGURE 4 – Patte en position de déplacement sur le robot

4.2 La base

La base du robot devait permettre de loger les fixations des 4 pattes, le bras ErgoJr placé au-dessus, la Raspberry et les deux batteries assurant l'autonomie du robot. Il a été décidé d'une base carrée pour des raisons d'optimisation de la place. Il a également été trouvé pratique, pour des raisons de solidité et de stabilité, de mettre deux plaques l'une en-dessous de l'autre, avec sur celle d'au-dessus le bras ErgoJr. Cela permet également de pouvoir fixer les pattes entre les deux plaques et d'y placer la Raspberry et les deux batteries (voir Figure 5).

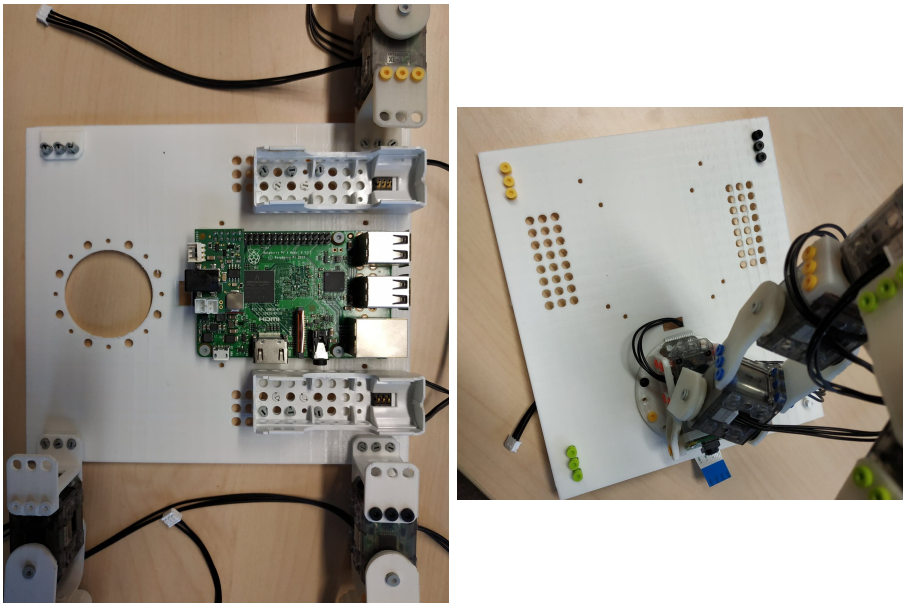

FIGURE 5 – Base du robot : (gauche) partie inférieure, (droite) partie supérieure

La solution avec les deux plaques permet un gain de place, elle permet aussi de renforcer la

solidité en particulier au niveau de la fixation des pattes (base complète figure 6). Pour aller plus vite la plaque du dessous et du dessus sont identiques (modèle en figure 7), ces deux plaques pourraient être revues et optimisés pour, en particulier, diminuer les efforts au niveau des pattes.

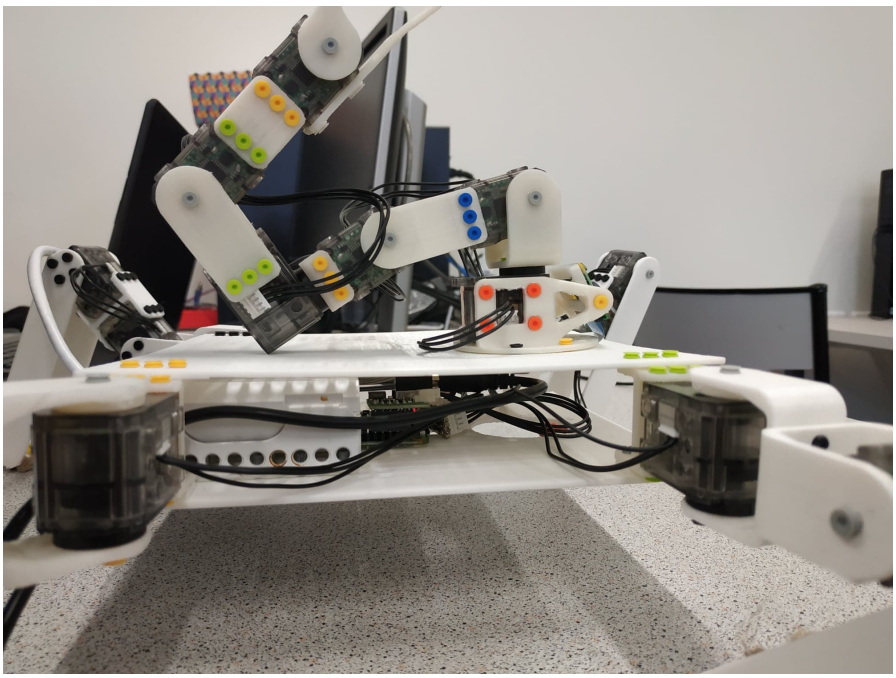

FIGURE 6 – Base complète du robot

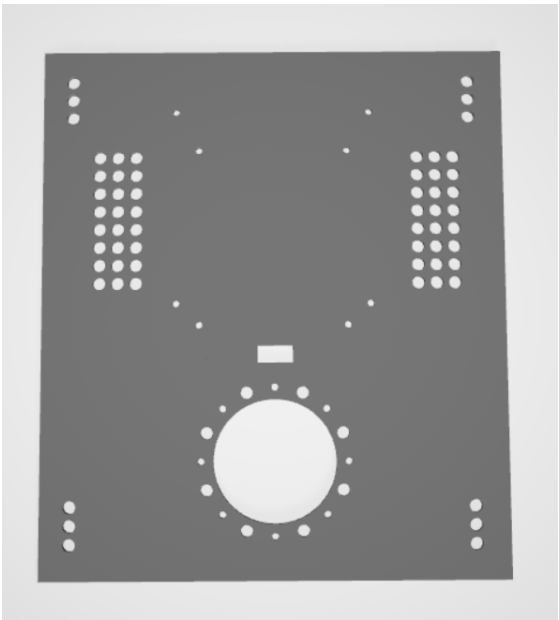

FIGURE 7 – Modèle 3D de la base

Pour que tous les moteurs soient reliés entre eux et à la Raspberry il a été nécessaire d'ajouter un hub à 6 ports pour permettre de connecter les 4 pattes et le bras ErgoJr à la Raspberry (Figure 8). Ainsi, chaque patte est connectée à un port, le bras également et le dernier port permet de relier le tout à la carte Raspberry.

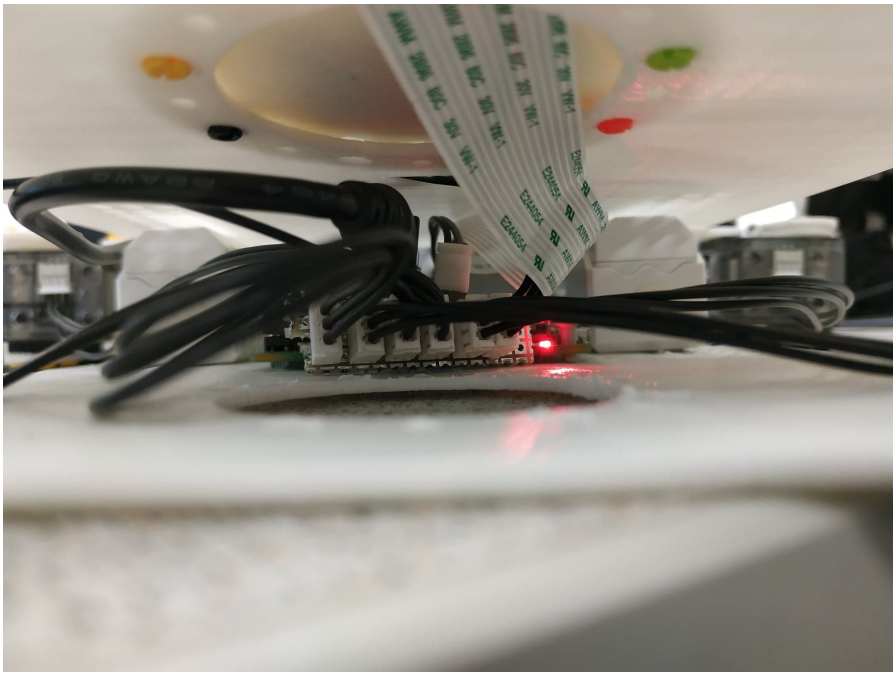

FIGURE 8 – Hub

4.3 Problèmes rencontrés

L'utilisation des imprimantes 3D de l'Eirlab a posé beaucoup de problèmes. Face à de nombreux problèmes nous retardant et la fin du projet approchant à grand pas nous avons demandé de l'aide à notre tuteur, qui a accepté de nous imprimer les pièces manquantes pour lesquelles nous lui avons fourni les modèles 3D.

Le deuxième problème important en terme de temps fût la création du hub. Les soudures n'étaient pas faciles à réaliser et nous ont demandé plusieurs essais qui ont été chronophages.

5 Déplacements

Cette partie développe ce qui a été mis en place afin de produire le mouvement des pattes générant le déplacement du robot.

L'ensemble du code développé est disponible sur le GitHub du projet fourni en annexe 1.

5.1 Modèle géométrique de la patte

Dans le but de faire se déplacer la créature, il a fallu avant tout être capable d'estimer les angles des articulations nécessaires pour imposer une certaine position de l'extrémité de la patte, et inversement, d'estimer la position du bout de la patte en fonction de l'état des articulations.

Une classe python `DiploLeg` a été crée afin de pouvoir instancier un ensemble de pattes, bénéficiant de primitives permettant d'imposer les angles de leurs moteurs, ou bien d'imposer les coordonnées de leur extrémité.

5.1.1 Conventions

Les graphiques suivants (Figure 9 et 10) montrent le repère utilisé pour chaque patte ainsi que les conventions de nommage d'angles et de distances.

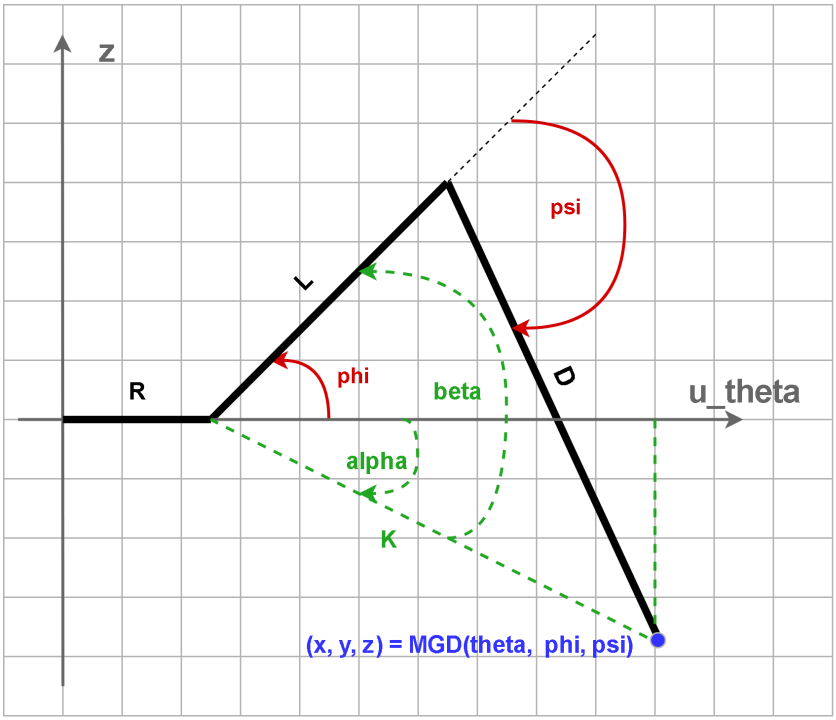

FIGURE 9 – Modèle géométrique d'une patte vue de côté

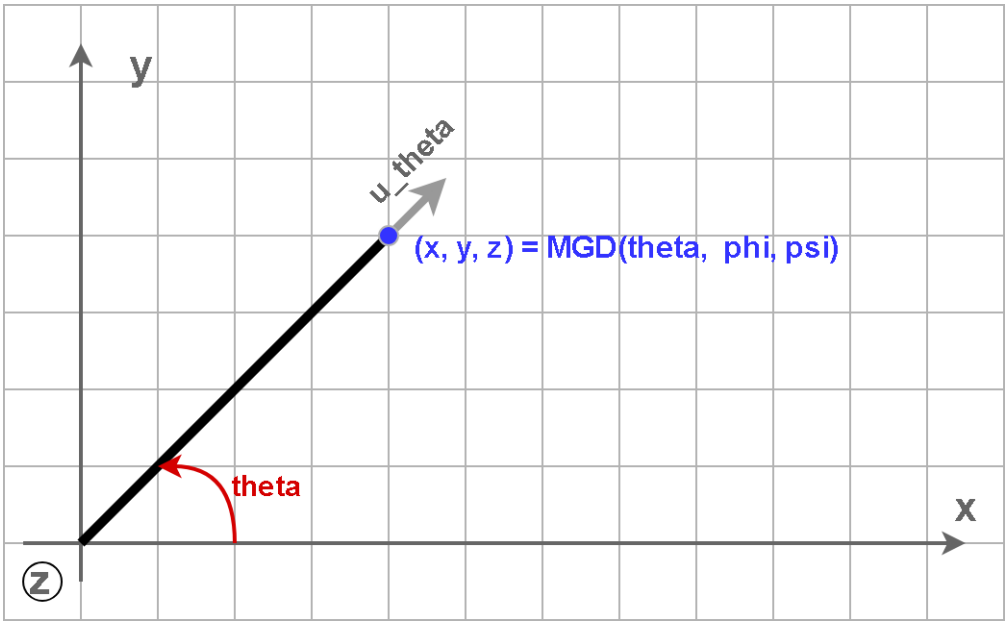

FIGURE 10 – Modèle géométrique d'une patte vue de dessus

5.1.2 Modèle géométrique direct

Le modèle géométrique direct consiste à calculer les coordonnées (x, y, z) de l'extrémité d'une patte, connaissant chacun de ses angles $(theta, phi, psi)$. La classe `DiploLeg` fournit une méthode `MGD` effectuant un calcul simple pour renvoyer ce résultat.

5.1.3 Modèle géométrique inverse

Le modèle géométrique inverse, consiste à trouver des angles $(theta, phi, psi)$ permettant de placer l'extrémité de la patte aux coordonnées (x, y, z) voulues. Il n'existe pas forcément de solution, ni de solution unique. La méthode `MGI` de la classe `DiploLeg` calcule une solution, et renvoie un résultat s'approchant le plus possible de la commande désirée, lorsque celle-ci est géométriquement impossible à atteindre.

L'angle *theta* est triviallement calculé à partir de x et y. Pour obtenir les angles *phi* et *psi*, des calculs intermédiaires sont nécessaires. La figure 9 montre en vert les variables intermédiaires utilisées. On utilise le théorème d'Al-Kashi, permettant de calculer les angles d'un triangle dont on connaît les longueurs des côtés. Ainsi, on obtient *psi* à partir du triangle de longueurs (L, D, K). Du fait de la non unicité des solutions, on impose $psi < 0$ afin d'obtenir une disposition de patte adaptée à la marche. De la même manière, *phi* est obtenu à partir de deux angles intermédiaires, *alpha* et *beta*, calculés respectivement grâce à (z, K) et le triangle de longueurs (L, D, K).

Ainsi, il est possible grâce à cette méthode de traduire une commande sur les coordonnées cartésienne de l'extrémité de la patte (dans un repère lié au robot) en commande sur chaque angle des moteurs.

5.2 Commandes de déplacement

Le modèle géométrique inverse couplé aux méthodes de la plateforme Poppy permettent d'imposer au bout de la patte une position dans l'espace propre au robot. Pour que le robot se déplace de différentes manières et afin de tester rapidement différentes méthodes de déplacement, des outils de création de trajectoires ont été créés.

5.2.1 Création du motif de trajectoire

Une classe `PositionPattern` a été écrite afin d'obtenir une commande en position horizontale et verticale au cours du temps. On créé un objet de cette classe en lui fournissant un ensemble de couples (instant, position). L'instance créée est une fonction d'interpolation disposant des couples précédents, et qui interpole entre deux instants consécutifs par un polynôme cubique, de dérivée nulle aux extrémités (une allure de courbes d'interpolation est présente en figure 12).

Le motif tracé en figure 11 a été retenu pour se déplacer. Il est composé d'une première phase où la patte avance horizontalement, puis d'une deuxième phase où la patte revient à sa position de départ en levant la patte vers le haut.

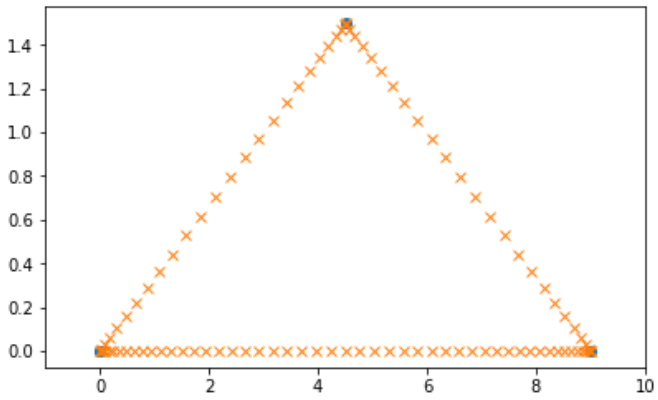

FIGURE 11 – Allure du motif retenu : horizontal en abscisse, vertical en ordonnée

La figure 12 montre le tracé des deux fonctions d'interpolation obtenues à partir des quelques points d'intérêt fournis.

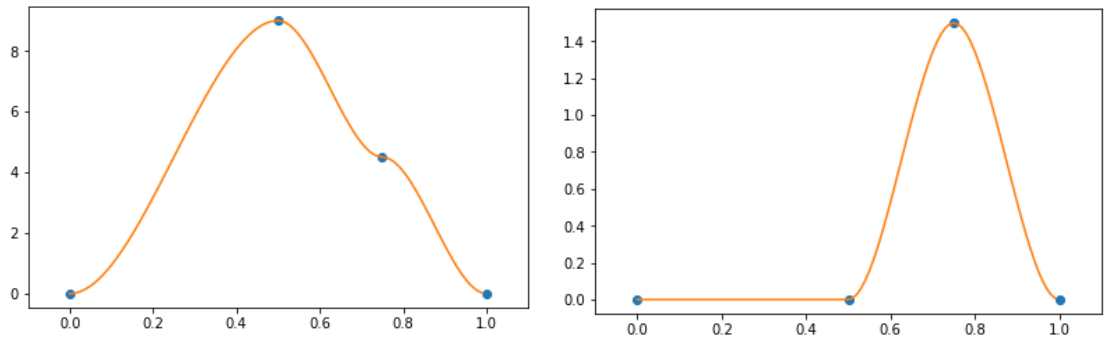

FIGURE 12 – Motif horizontal (gauche) et vertical (droit) en fonction du temps

5.2.2 Positionnement du motif dans l'espace

Une fois un motif créé, il peut être placé dans l'espace par la création d'un objet de la classe `PositionCommand`. En modifiant le point de départ du motif et sa direction horizontale, nous avons pu implémenter les déplacements globaux du robot que sont : avancer, tourner sur lui-même à droite et tourner sur lui-même à gauche.

6 Guide

Le robot créé a pour vocation d'être refait par des utilisateurs disposant de 3 kits ErgoJr. Il était donc nécessaire de réaliser un guide détaillant toutes les étapes de la construction pour pouvoir le reproduire, à la fois au niveau de la structure que des programmes. Il a donc été réalisé un guide complet.

Ce guide a été séparé en trois grandes parties :

1. Matériel et composants
2. Programmes supplémentaires Raspberry
3. Assemblage du robot

En suivant l'ensemble du guide, n'importe quel utilisateur disposant du matériel et des composants peut construire notre créature, que nous avons nommé Diplo, de manière fonctionnelle. La créature est manipulable via des primitives accessibles dans Snap ! ou bien par l'utilisation de nos fonctions Python qui sont modifiables à la guise de l'utilisateur.

6.1 Matériel et composants

Cette première partie recense l'ensemble du matériel nécessaire à la construction de notre robot. L'accent a été mis sur les composants fournis dans les kits ErgoJr, ceux qu'il faut imprimer en 3D et ceux à récupérer d'une autre manière (extrait en figure 13).

2.1 Pièces du Diplo (ne sont pas toutes comprises dans les kits Ergo Jr)

Pièces comprises dans les kits Ergo Jr, disponibles sur le github du Poppy Ergo Jr : https://github.com/poppy-project/poppy-ergo-jr/tree/master/hardware/STL

- 1 x carte d'extension Pixl (carte électronique de contrôle des moteurs XL320 depuis une Raspberry Pi)
- 3 x horn2horn.stl
- 3 x side2side.stl
- 1 x shortU.stl
- 1 x supportcamera.stl
- 1 x lamp.stl
- 1 x gripper-fixedpart.stl
- 1 x gripper-rotativepart.stl
- 1 x pen-holder.stl
- 1 x pen-screw.stl

Pièces non comprises dans les kits Ergo Jr, disponibles sur le github du Diplo : https://github.com/poppy-project/poppy-ergo-jr/tree/master/hardware/STL

- 2 x support.stl (0 par kit Ergo Jr, **2** à imprimer en 3D)
- 4 x cale.stl (0 par kit Ergo Jr, **4** à imprimer en 3D)
- 9 x longU.stl (1 par kit Ergo Jr, **6** à imprimer en 3D)
- 5 x gripper-fixation.stl (1 par kit Ergo Jr, **2** à imprimer en 3D)
- 4 x patte.stl (0 par kit Ergo Jr, **4** à imprimer en 3D)
- 4 x longH.stl (0 par kit Ergo Jr, **4** à imprimer en 3D)

FIGURE 13 – Extrait guide matériel et composants

6.2 Programmes supplémentaires Raspberry

Cette partie vise à expliquer comment installer les fichiers nécessaires au bon fonctionnement du robot sur la Raspberry (extrait en figure 14). Cette partie vient avant la partie assemblage du robot car elle est nécessaire pour pouvoir configurer les moteurs et avoir les programmes nécessaires pour contrôler le robot.

3. Cliquez sur Upload (en haut à droite)
4. Allez chercher votre fichier "poppy_ergo_jr.tar.gz" et uploadez le, votre écran doit être comme celui de la figure 2.

FIGURE 2 – Etape 4 état du dossier poppy_src

5. Cliquez maintenant sur "New" (en haut à droite) puis sur "Terminal"
6. Entrez la commande : export PATH=~/miniconda/bin :/usr/bin :/bin et exécutez la

FIGURE 14 – Extrait guide programmes supplémentaires Raspberry

6.3 Assemblage du robot

Cette dernière partie sert à monter le robot. A la fin de celle-ci, le robot est complet et prêt à être utilisé. La première partie du montage consiste à monter un bras ErgoJr. Pour cela, il est fait référence au guide déjà écrit dans la documentation de la plateforme Poppy.

La suite implique le montage des pattes, la configuration des moteurs, le montage de la base et la finalisation de la structure. Les explications sont agrémentées de nombreuses images pour

rendre la construction plus aisée et éviter les ambiguïtés (extrait en figure 15). Vous pouvez retrouver notre guide d'assemblage en annexe 2 de ce document.

FIGURE 5 – Etape 4

Étape 5 : Fixation du long H sur le deuxième moteur Le long H doit être fixé sur les faces où le U n'a pas été fixé. Ceci est important pour laisser du jeu au pied de la patte.

FIGURE 6 – Etape 5

Étape 6 : Fixation du troisème moteur sur le long H Attention à ce que les parties noires tournantes des moteurs deux et trois soient dans le même sens.

FIGURE 15 – Extrait guide assemblage du robot

7 Conclusion

Ce projet nous a permis de nous immerger dans un domaine en pleine expansion qu'est la robotique éducative et d'appréhender ses contraintes. Nous avons pu également nous confronter à la réalité de la réalisation et de la construction d'un nouveau robot.

L'objectif principal de ce projet était de permettre aux enseignants, disposants déjà de kits ErgoJr, de renouveller leurs enseignements en produisant une nouvelle créature à partir de ces kits. Il fallait également l'intégrer à la plateforme Poppy afin que les utilisateurs puissent reproduire notre créature et l'utiliser comme ils le font habituellement avec les autres créatures de la plateforme. Enfin un guide permettant de reproduire le robot devait être fourni.

Les objectifs ont été rempli par la création de la créature nommée Diplo qui est capable de se déplacer en avant et de tourner à droite et à gauche. L'intégration sur la plateforme Poppy est assuré par les éléments disponibles sur le GitHub du projet. Le guide d'assemblage permettant la reproduction de la créature a été achevé.

Le projet est en open source, il est donc possible à quiconque de le reprendre pour l'utiliser mais aussi pour l'améliorer.

8 Annexe

8.1 Annexe 1 : Lien GitHub

Le GitHub du projet est composé de deux dossiers. Le dossier Hardware contient les modèles 3D de la créature qu'il est nécessaire d'imprimer en 3D. Le dossier Software contient le code permettant de faire fonctionner la créature.

Lien GitHub : https://github.com/aJuvennn/poppy-creature

8.2 Annexe 2 : Guide de montage de Diplo

Poppy Diplo

TUTORIEL

ENSEIRB-MATMECA - ENSC - INRIA

30 janvier 2019

Créateurs : Nina Docteur, Alexis Juven, Vincent Leconte

Tuteur : Thibault Desprez

Contact : thibault.desprez@inria.fr

Table des matières

Table des figures

1 Introduction

Ce tutoriel s'articule autour du projet Poppy, un projet Open Source visant à développer des robots faciles à construire et à utiliser dans le cadre éducatif mais aussi artistique et scientifique. Plusieurs kits basés sur ce projet ont été développés, dont le kit ErgoJr qui permet de construire son propre bras robotique à partir de moteurs, de pièces imprimées en 3D et d'une carte Raspberry. Afin d'agrandir la collection de robots Poppy, un projet a été mis en place pour construire un nouveau robot qui soit capable de se déplacer, avec pour base 3 kits ErgoJr. Ici est présenté le tutoriel visant à reproduire ce robot, du nom de Diplo.

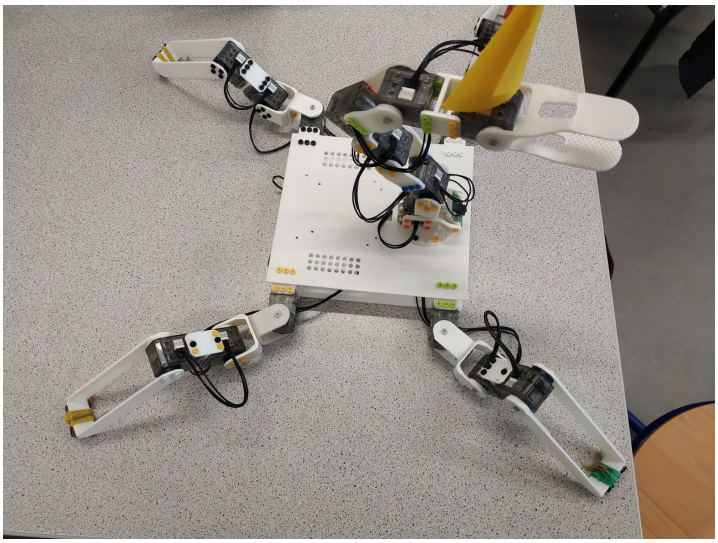

FIGURE 1 – Robot Diplo

2 Matériel et composants

Le Diplo est basé sur 3 kits Ergo Jr, néanmoins il vous faudra avoir accès à une imprimante 3D pour imprimer plusieurs pièces supplémentaires. Il vous faudra également 2 cables de liaison des moteurs un peu plus longs que ceux standards de l'Ergo Jr et un hub pour 6 cables moteurs.

2.1 Pièces du Diplo (ne sont pas toutes comprises dans les kits Ergo Jr)

Liste des pièces comprises dans les kits Ergo Jr, utilisées pour le Diplo : (vous pouvez vous procurer les fichiers des pièces pour l'impréssion 3D avec l'extention '.stl' sur le github du Poppy Ergo Jr : https://github.com/poppy-project/poppy-ergo-jr/tree/master/hardware/STL)

- 1 x carte d'extension Pixl (carte électronique de contrôle des moteurs XL320 depuis une Raspberry Pi)
- 2 x horn2horn.stl
- 2 x side2side.stl
- 1 x shortU.stl
- 1 x supportcamera.stl
- 1 x gripper-fixedpart.stl

— 1 x gripper-rotativepart.stl

Liste de pièces non comprises dans les kits Ergo Jr : (vous pouvez vous procurer les fichiers des pièces pour l'impréssion 3D avec l'extention '.stl' sur le github du Diplo : https://github.com/poppy-project/poppy-ergo-jr/tree/master/hardware/STL)

— 2 x support.stl (0 par kit Ergo Jr, **2** à imprimer en 3D)

— 4 x cale.stl (0 par kit Ergo Jr, **4** à imprimer en 3D)

— 9 x longU.stl (1 par kit Ergo Jr, **6** à imprimer en 3D)

— 5 x gripper-fixation.stl (1 par kit Ergo Jr, **2** à imprimer en 3D)

— 4 x patte.stl (0 par kit Ergo Jr, **4** à imprimer en 3D)

— 4 x longH.stl (0 par kit Ergo Jr, **4** à imprimer en 3D)

2.2 Pièces faites par Robotis (comprises dans trois kits Ergo Jr)

— 18 x servomoteurs dynamixel XL-320

— 3 x jeu de rivets OLLO (vous aurez besoin 148+70 (218)rivets colorés et de 34+4 (38) longs rivets gris, donc un sachet de riviets est suffisant)

— 1 x outil OLLO

2.3 Visserie (comprise dans un kit Ergo Jr)

— 4 x M2.5x6mm vis (pour fixer la Raspberry Pi sur le socle) (possible de les remplacer par des rivets longs)

— 4 x M2x5mm vis (pour fixer la caméra)

— 4 x écrous M2 (fixation caméra)

— 1 x entretoise M2.5 mâle/femelle 10mm

2.4 Électronique (comprise dans un kit Ergo Jr à l'exception des batteries

— 1 x Raspberry Pi 2 ou 3

— 1 x micro SD 8Go (ou plus)

— 1 x caméra Raspberry Pi

— 1 x alimentation 7.5V 2A avec un connecteur 2.1 x 5.5 x 9,5 (celle-ci par exemple)

— 1 x câble Ethernet court

— 2 x batterie Li-ion 3.7V 1300mAh LB-040

— 2 x support de batterie Li-ion LBB-041

3 Programmes supplémentaires Raspberry

Dans cette partie nous allons installer sur la Raspberry les fichiers nécessaires au fonctionnement du robot. Cette partie est également nécessaire pour pouvoir configurer les moteurs dans la partie "Assemblage du robot".

Vous devez avant tout suivre le guide suivant comme si vous souhaitiez préparer votre Raspberry pour un Poppy Ergo Jr : https://docs.poppy-project.org/fr/getting-started/connect.html.

Vous allez maintenant avoir besoin des fichiers modifiés du Poppy Ergo Jr destinés au Diplo. Pour cela rendez-vous sur le lien github suivant : https://github.com/aJuvennn/poppy-creature . Le fichier qui nous intéresse est situé dans le dossier software et se nomme "poppy_ergo_jr.tar.gz". Téléchargez cette archive en cliquant dessus puis en cliquant sur Download.

Ouvrez maintenant l'interface web comme expliqué en bas du guide fait précédemment (https://docs.poppy-project.org/fr/getting-started/connect.html). Nous allons maintenant vous guider pas à pas.

1. Ouvrez Jupyter depuis l'interface web
2. Cliquez sur le dossier poppy_src
3. Cliquez sur Upload (en haut à droite)
4. Allez chercher votre fichier "poppy_ergo_jr.tar.gz" et uploadez le, votre écran doit être comme celui de la figure 2.

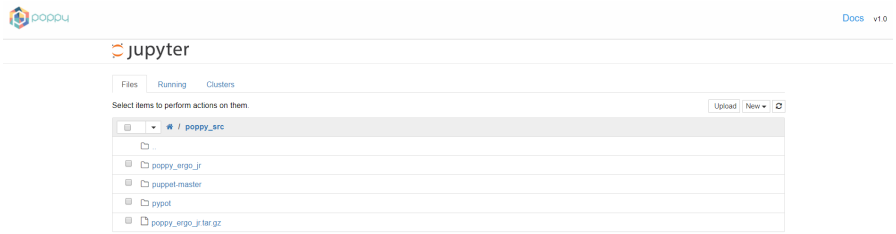

FIGURE 2 – Etape 4 état du dossier poppy_src

5. Cliquez maintenant sur "New" (en haut à droite) puis sur "Terminal"
6. Entrez la commande : `export PATH=~/miniconda/bin:/usr/bin:/bin` et exécutez la. Elle permet de pouvoir exécuter les commandes suivantes.
7. Entrez la commande : `cd poppy_src` et exécutez la. Elle permet de se mettre dans le bon dossier.
8. Entrez la commande : `rm -r poppy_ergo_jr` et exécutez la. Elle permet de supprimer le dossier poppy_ergo_jr.
9. Entrez la commande : `tar -xzvf poppy_ergo_jr.tar.gz` et exécutez la. Elle permet de dézipper l'archive.

Votre Raspberry est maintenant configurée pour le Diplo, vous pouvez passer à l'assemblage.

4 Assemblage du Robot

L'assemblage du robot comprend la configuration des moteurs. Il est possible de configurer les moteurs après avoir construit le robot mais il est plus facile de le faire pendant. Des codes couleurs ont été utilisés pour les rivets et nous vous conseillons de les suivre pour éviter des erreurs de réglages et de positionnement des pattes.

4.1 Assemblage d'un Ergo Jr

Le Diplo comprend un bras Ergo Jr, sa configuration et sa construction sont nécessaires pour pouvoir monter le Diplo, la première étape de l'assemblage du Diplo est donc d'assembler un Ergo Jr. Pour cela suivez le guide suivant : https://docs.poppy-project.org/fr/assembly-guides/ergo-jr/ .

Si vous avez suivi la partie 3, "Programmes supplémentaires Raspberry" ne touchez pas à la configuration de votre Raspberry. Mais si vous ne l'avez pas faite nous vous conseillons de la faire maintenant car elle est nécessaire pour faire fonctionner le Diplo et qu'après l'assemblage du robot, la carte SD n'est plus facilement accessible.

4.2 Assemblage du Diplo

A ce niveau vous devez avoir en votre possession un bras Ergo Jr déjà monté accompagné de sa Raspberry déjà configurée. La première étape va être de monter chacune des pattes. Ensuite nous allons configurer leurs moteurs. Puis nous préparerons la base pour finir par réuninir les différents éléments.

4.2.1 Les pattes

Il y a 4 pattes à monter, chacune est construite exactement de la même manière, seule la configuration des moteurs diffère. Vous avez donc deux méthodes pour monter les pattes, soit faire les 4 pattes les unes à la suite des autres soit faire les 4 en même temps étape par étape.

Chaque patte nécessite les 5 éléments imprimés en 3D suivants :

— Un petit H

— Deux U

— Un long H

— Un pied

Ainsi que :

— 3 moteurs

— 34 petits rivets

— 3 longs rivets

Pour chacune des étapes avec présence d'un moteur vérifiez que la petite encoche de la plaque noire tournante est bien alignée avec la marque sur le plastique du moteur (comme montré sur la Figure 3) lorsque votre patte est à plat.

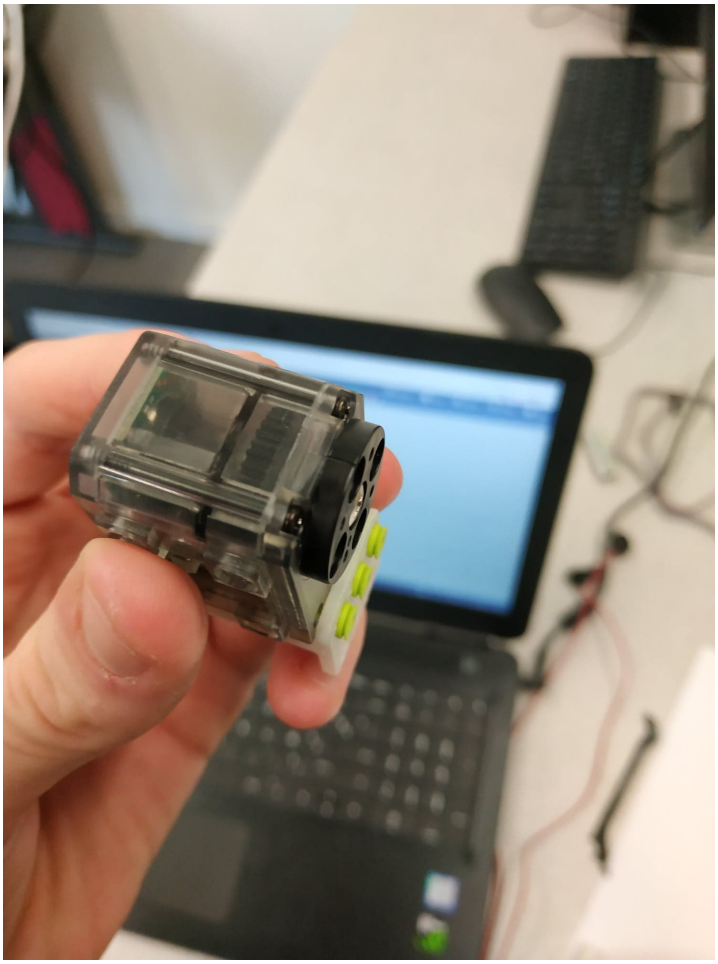

FIGURE 3 – Encoche moteur

Étape 1 : Partie base de la patte La première étape est de fixer un petit H à un premier moteur, nous vous conseillons d'appliquer un code couleur pour fixer les rivets de cette pièce :

- Avant gauche : blanc
- Arrière gauche : noir
- Arrière droit : jaune
- Avant droit : vert

En appliquant ce code vous serez cohérent avec les images présentes dans ce tutoriel et aurez moins de chance de vous tromper lors de la configuration des moteurs.

La partie plus courte du H doit être fixée sur le moteur (voir Figure 4).

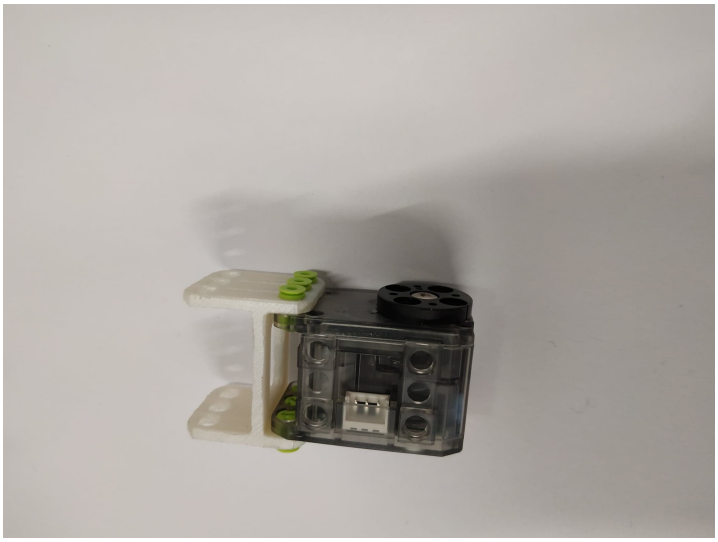

FIGURE 4 – Etape 1

Étape 2 : Fixation du deuxième U sur le premier U Attention pour cette étape faites attention de bien fixer le deuxième U dans le même sens que présenté sur la figure 5, sinon vous n'allez pas fixer les moteurs dans le bon sens.

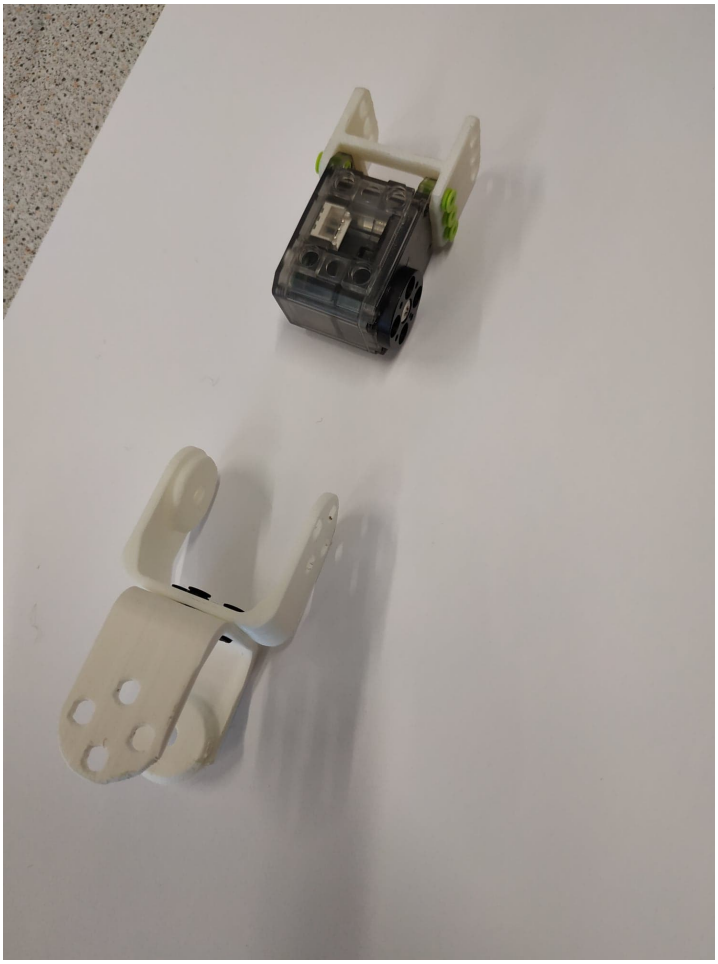

FIGURE 5 – Etape 2

Étape 3 : Fixation d'un U au premier moteur Fixez le premier U au moteur que vous avez lié au petit H. Vous devez obtenir quelque chose semblable à ce qui est montré sur la figure 6.

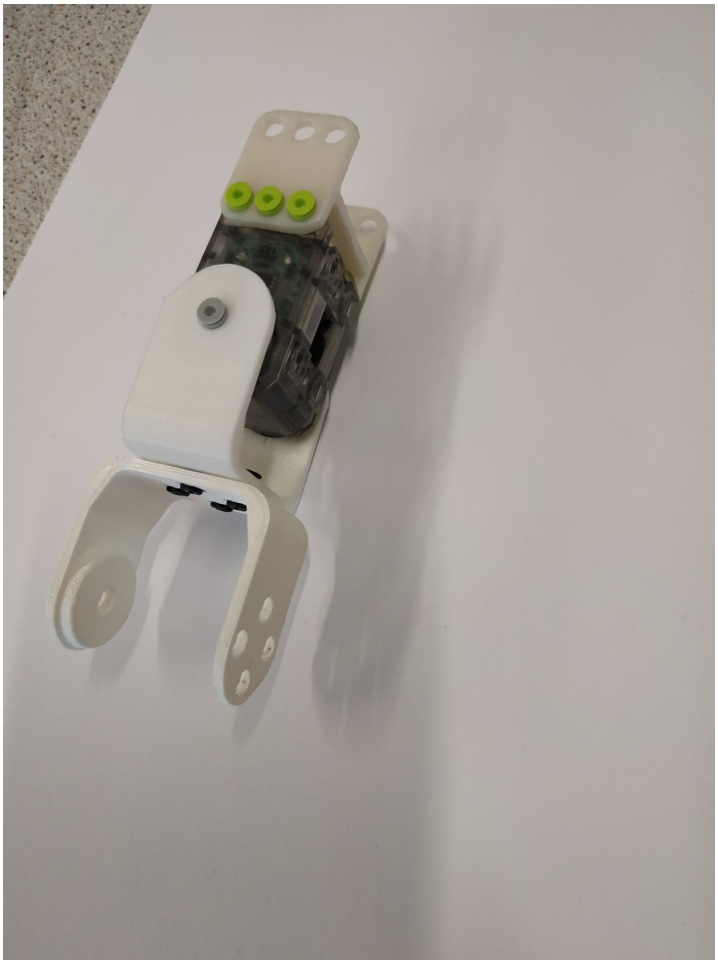

FIGURE 6 – Etape 3

Étape 4 : Fixation du deuxième moteur sur le deuxième U Fixez le deuxième moteur comme indiqué sur la figure 7.

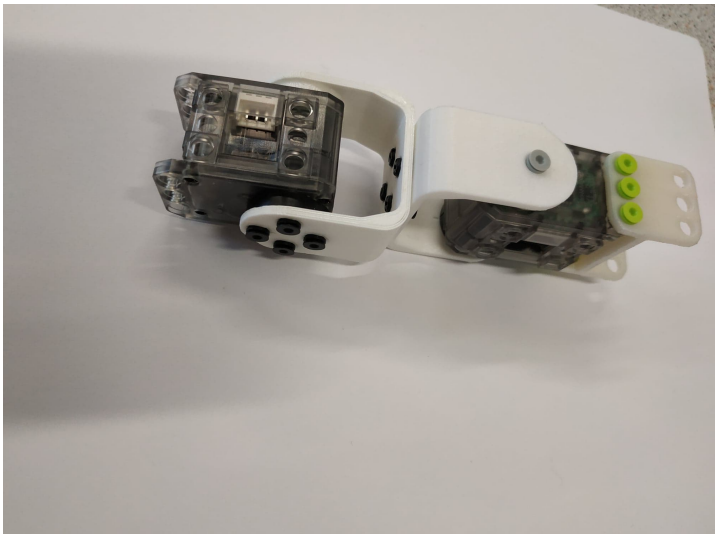

FIGURE 7 – Etape 4

Étape 5 : Fixation du long H sur le deuxième moteur Le long H doit être fixé sur les faces où le U n'a pas été fixé. Ceci est important pour laisser du jeu au pied de la patte. (Voir Figure 8)

FIGURE 8 – Etape 5

Étape 6 : Fixation du troisème moteur sur le long H Attention à ce que les parties noires tournantes des moteurs deux et trois soient dans du même côté (Voir Figure 9).

FIGURE 9 – Etape 6

Étape 7 : Fixation du pied sur le troisième moteur Attention à bien fixer le pied dans l'alignement (Voir Figure 10).

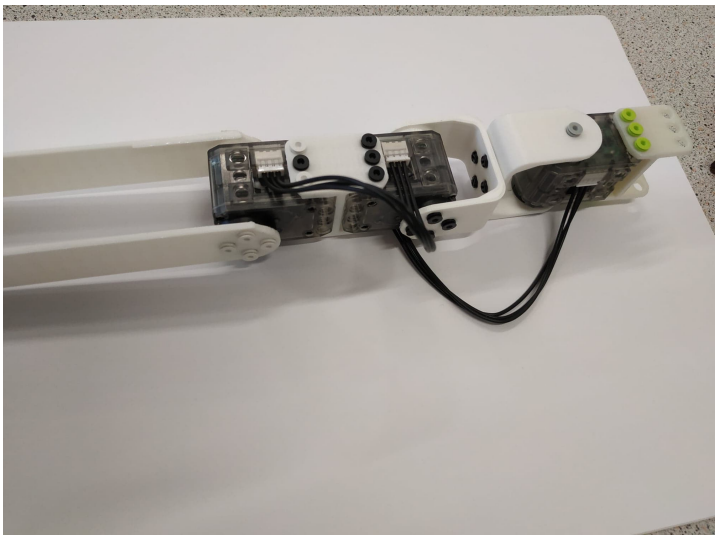

FIGURE 10 – Etape 7

4.2.2 Configuration des moteurs

La configuration des moteurs nécessite que vous ayez réalisé sur votre Raspberry les étapes d'installation des programmes du Diplo décrits dans la section "Programmes supplémentaires Raspberry".

Les moteurs du robot sont numérotés comme suit en partant de la base vers le pied :

— Avant gauche : 7 8 9
— Arrière gauche : 10 11 12
— Arrière droit : 13 14 15
— Avant droit : 16 17 18

FIGURE 11 – Numerotation des moteurs

Pour les configurer suivez la même manipulation que présentée dans l'assemblage de l'Ergo Jr : https://docs.poppy-project.org/fr/assembly-guides/ergo-jr/motor-configuration.html

Attention rappel : lors de la configuration des moteurs il faut que seul celui en cours de configuration soit connecté à la Raspberry et qu'il ne soit relié à aucun autre moteur. Si vous ne respectez pas cela il faudra recommencer la configuration et reconfigurer tous les moteurs qui étaient liés, un à un.

4.2.3 Assemblage final

Pour faire l'assemblage il vous faudra :

— Les deux deux plaques carrées imprimées en 3D
— Les 4 petites cales
— Un hub pour 6 cables moteurs
— La Raspberry
— Les deux batteries
— Le bras robotique Ergo jr monté
— Les 4 pattes montées
— 18 longs rivets
— 16 petits rivets

Raspberry et batteries Munissez-vous d'une plaque, de la Raspberry et de deux supports de batteries, fixez les comme présenté sur l'image. Pour fixer la Raspberry vous pouvez utilier la partie "vis" de 4 longs rivets en les insérant dans les coins. Les socles à batteries se fixent chacun avec 3 longs rivets (gris sur la Figure 12) et 4 petits rivets (blancs sur la Figure 12).

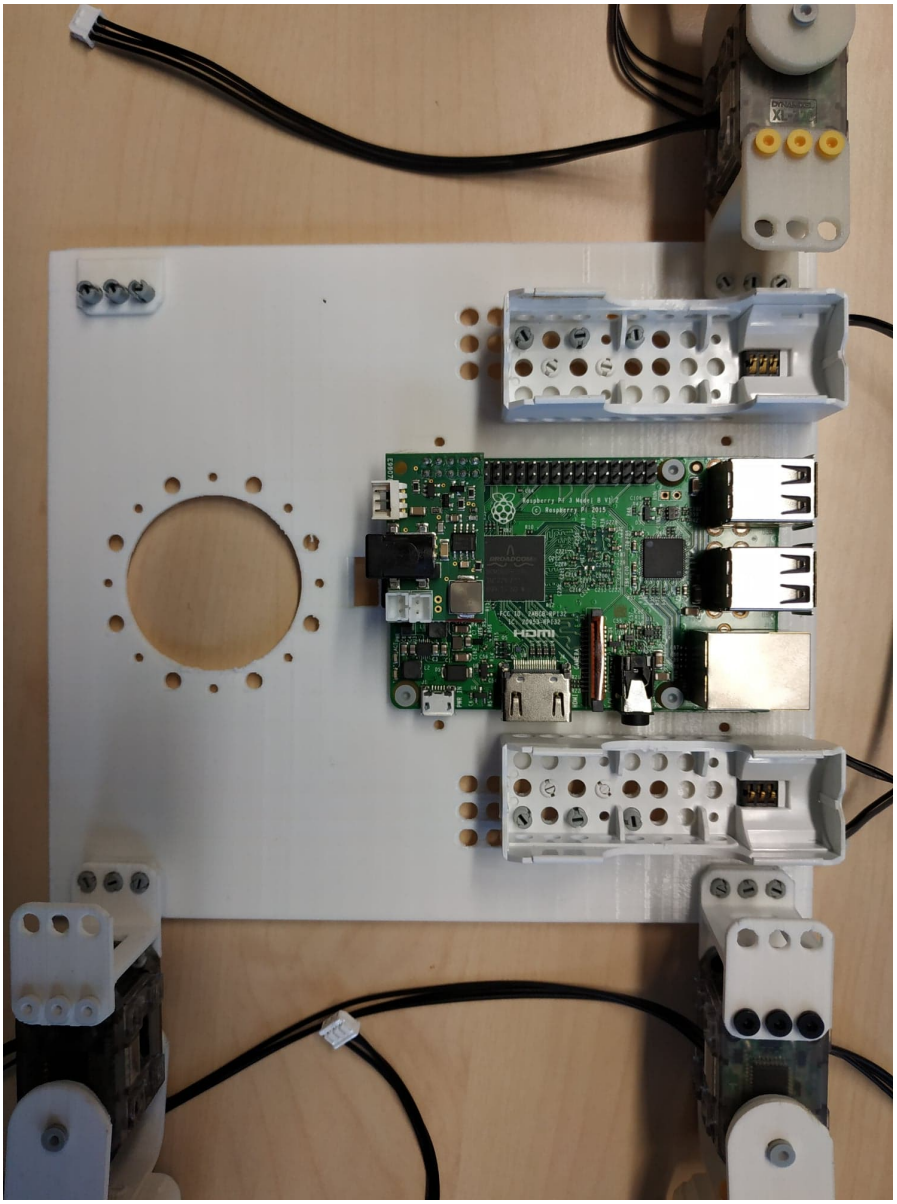

FIGURE 12 – Fixation Raspberry et moteurs

Fixation des pattes Pour fixer les pattes à la première plaque comme montré sur l'image précédente vous aurez besoin de 3 longs rivets par patte. En-dessous de chaque patte vous devrez fixer une des 4 petites cales. Il est compliqué de fixer les 3 rivets en même temps à une patte, n'hésitez donc pas à en mettre deux puis le dernier. Pour fixer les pattes réfèrez-vous au code couleur et à la figure 13.

— Avant gauche : blanc - moteurs : 7 8 9
— Arrière gauche : noir - moteurs : 10 11 12
— Arrière droit : jaune - moteurs : 13 14 15
— Avant droit : vert - moteurs : 16 17 18

Partie supérieure de la base Munissez-vous du bras Ergo Jr et de la deuxième plaque imprimée en 3D. Faites passer la connectique de la caméra du bras sous la plaque (par le gros rond) et fixez le bras Ergo Jr comme affiché sur la figure 13 avec quelques petits rivets. Préparez les rivets de fixation des pattes comme affiché sur l'image en-dessous, essayez de respecter le code couleur pour plus de faciliter :

— Avant gauche : blanc
— Arrière gauche : noir
— Arrière droit : jaune
— Avant droit : vert

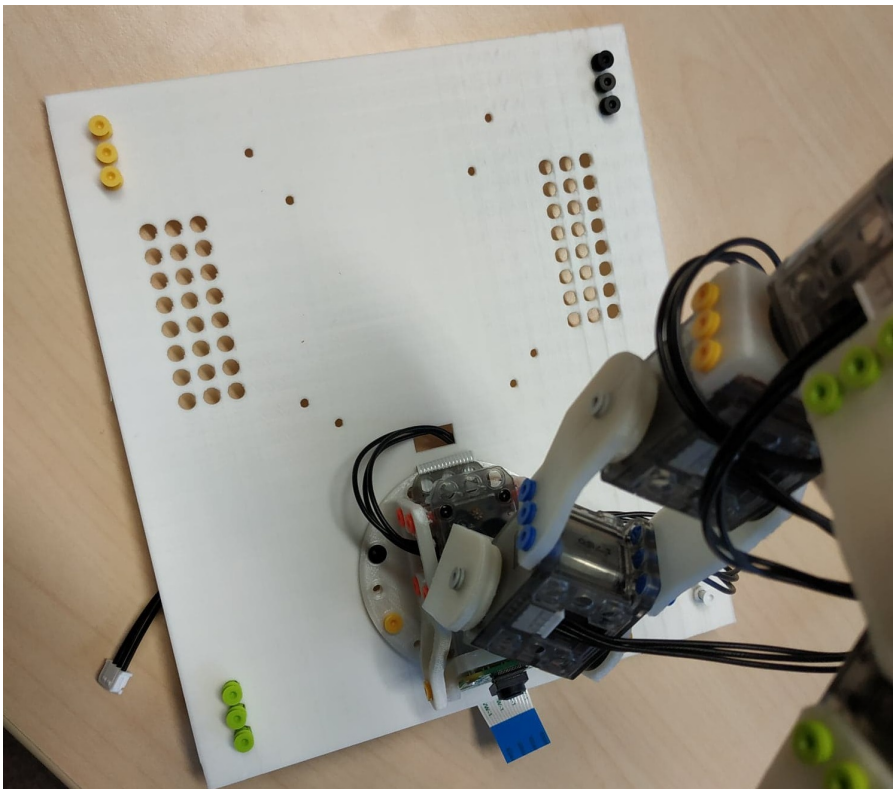

FIGURE 13 – Partie superieure base

Fixation de l'ensemble Avant de fixer les deux plaques ensembles placez les batteries dans leurs socles si ce n'est pas déjà fait, attention à ce que la connectique soit du bon côté (côté extérieur).

Pour l'utilisation des batteries n'oubliez pas que vous ne devez pas les utiliser pendant qu'elles rechargent et de les débrancher si vous ne les utilisez pas.

1. Branchez les petits cables des batteries sur la Raspberry sur les deux connecteurs prévus à cet effet.
2. Branchez 5 cables moteurs (3 courts et 2 long) sur le hub (le sixième sera pour le bras et sera fixé après).
3. Branchez un des cables courts lié au hub sur la Raspberry.
4. Branchez la connectique de la caméra du bras sur la Raspberry, référez-vous à la partie caméra du guide d'assemblage de l'Ergo Jr : https://docs.poppy-project.org/fr/assembly-guides/ergo-jr/mechanical-construction.html
5. Passez le cable relié au moteur 1 du bras (le plus proche de la base) dans le petit trou prévu à cet effet sur la partie supérieure de la base (petit rectangle situé au-dessus du gros rond). Puis fixez le cable sur le dernier port restant du hub.
6. Fixez la partie supérieure des pattes à la partie supérieure de la base avec les petits rivets que vous avez normalement placés dans la partie "Partie supérieure de la base".
7. Branchez les cables du hub sur les premiers moteurs des pattes, les deux longs cables pour les pattes arrières, les deux courts pour les pattes de devant. (Vous avez normalement relié l'un des deux derniers à la Raspberry et l'autre au bras Ergo Jr).
8. Si ce n'est pas fait reliez les moteurs entre eux comme visible sur la figure 10 et 14.

Votre Diplo est prêt, amusez-vous bien.

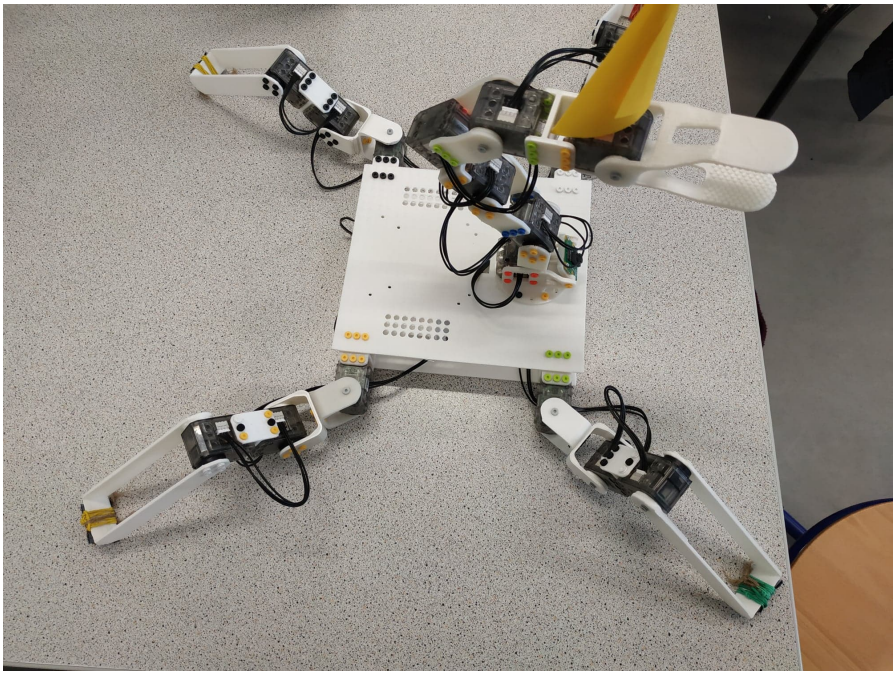

FIGURE 14 – Diplo complet

8.3 Annexe 3 : Cahier des charges

CAHIER DES CHARGES

Poppy

Référence	CDC_POPPY_V0
Projet	POPPY
Tuteur	**Elèves**
Thibault Desprez	Vincent LECONTE
	Nina DOCTEUR
	Alexis JUVEN

Table des matières

I. Introduction

Ce projet s'articule autour du projet Poppy. Plusieurs kits basés sur ce projet ont été développés, dont le kit ErgoJr qui permet de construire son propre bras robotique à partir de moteurs, de pièces imprimées en 3D et d'une carte Raspberry.

Afin d'agrandir la collection de robots Poppy, le but de ce projet est de construire un nouveau robot qui soit capable de se déplacer, avec pour base 3 kits ErgoJr. Cela permettrait aux enseignants possédant des kits ErgoJr de fabriquer une toute nouvelle créature sans avoir à acheter un kit entier.

I.1. Contexte du projet

Le projet Poppy est un projet Open Source visant à développer des robots faciles à construire et à utiliser dans le cadre éducatif mais aussi artistique et scientifique. Grâce à cela, des professeurs ont dans leur classe des kits ErgoJr afin d'enseigner à leurs élèves les bases de la programmation et plus généralement de la robotique.

I.2. Pré-existant

Les kits ErgoJr sont des kits complets comportant chacun 6 moteurs (les XL-320 de dynamixel), des pièces imprimées en 3D permettant d'articuler les moteurs entre eux et formant la pièce opérationnelle du robot (pince, porte crayon ou abat-jour), une petite caméra et une carte Raspberry permettant de contrôler le robot via la plateforme Poppy.
Un état de l'art a été réalisé préalablement à ce cahier des charges.

II. Description technique des besoins client

II.1. Objectif du projet

Le projet a pour but la création d'un robot mobile intégré à la plateforme Poppy à partir de 3 kits ErgoJr, et la mise à disposition d'un guide expliquant comment reproduire et utiliser le robot avec les ressources nécessaires.

II.2 Description des produits attendus

À l'état de l'art déjà réalisé s'ajouteront trois livrables:

- le robot mobile réalisé pendant le projet
- les ressources nécessaires à sa création (modèle des pièces 3D, code s'exécutant sur le robot)
- un guide permettant de recréer un tel robot à partir des ressources fournies

II.3 Exigences techniques et fonctionnelles

Le livrable devra respecter certain critères.
Afin de minimiser le coût de création du robot pour des professeurs possédant déjà des kits Ergo Jr., le

robot devra être constitué le plus possibles de pièces des Ergo Jr.
Le guide de construction et les fichiers disponibles permettront à n'importe quel professeur ayant accès à l'impression 3d et la découpe laser de recréer le robot.
Une fois monté, il sera possible de contrôler et déplacer le robot via des blocs Snap!, permettant une prise en main et une utilisation rapide du robot.

III. Exigences portant sur la conduite du projet

III.1 Durée du projet

L'état de l'art du projet a commencé le 19.10.2018 et a été clôturé le 30.11.2018 par une soutenance.
Le projet se finira complètement le 29 janvier par la remise d'un rapport suivi le 1er février par une soutenance.

III.2 Critères d'acceptation du produit

Le projet sera considéré comme achevé s'il y a un robot mobile accompagné d'un guide pour le reproduire.
La mobilité devra être facilement expérimentable via l'utilisation de blocs Snap! .

III.3 Structuration du planning

Le planning présenté ici démarre au 07 janvier 2019, date de première remise du cahier des charges.

- Guide : réalisation en continue
- Structure du robot : design du robot et des pièces permettant de passer à la partie développement en ayant la possibilité de tester
  - Design du robot : structure générale
  - Design des pièces : à partir de la structure globale modélisation des différentes pièces en vue d'une impression
  - Impression 3D : obtenir une structure utilisable
- Modèles pour le déplacement : modèles mathématiques permettant la mise en place des déplacements
  - Modèle Géometrique Direct
  - Modèle Géometrique Indirect
- Commandes de déplacement
  - Vitesse
  - Rotation
  - Intégration plateforme Poppy
- Déplacements élémentaires
  - Avancer
  - Tourner
  - Intégration plateforme Poppy
- Rapport : réalisation en continue à partir du 21 janvier
- Préparation soutenance

Nom	Date de début	Date de fin
Guide	07/01/19	29/01/19
Structure du robot	07/01/19	21/01/19
Design du robot	07/01/19	14/01/19
Design des pièces	07/01/19	14/01/19
Impression 3D	21/01/19	21/01/19
Modèles pour le déplacement	14/01/19	21/01/19
MGD	14/01/19	14/01/19
MGI	21/01/19	21/01/19
Commandes de déplacement	22/01/19	25/01/19
Vitesse	22/01/19	24/01/19
Rotation	22/01/19	24/01/19
Intégrations plateforme Poppy	24/01/19	25/01/19
Déplacements élémentaires	25/01/19	29/01/19
Avancer	25/01/19	28/01/19
Tourner	25/01/19	28/01/19
Intégrations plateforme Poppy	28/01/19	29/01/19
Rapport	21/01/19	30/01/19
Preparation soutenance	31/01/19	31/01/19

8.4 Annexe 4 : État de l'art

Projet robotique autonome : Poppy

État de l'art

Nina Docteur, Alexis Juven, Vincent Leconte

ENSEIRB-MATMECA - ENSC

Troisième année option Robotique

30 janvier 2019

Tuteur de projet : Thibault Desprez

Table des matières

Table des figures

1 Introduction

Ce projet s'articule autour du projet Poppy, un projet Open Source visant à développer des robots faciles à construire et à utiliser dans le cadre éducatif mais aussi artistique et scientifique. Plusieurs kits basés sur ce projet ont été développés, dont le kit ErgoJr qui permet de construire son propre bras robotique à partir de moteurs, de pièces imprimées en 3D et d'une carte Raspberry. Afin d'agrandir la collection de robots Poppy, le but de ce projet est de construire un nouveau robot qui soit capable de se déplacer, avec pour base 3 kits ErgoJr. Cela permettrait aux enseignants possédant des kits ErgoJr de fabriquer une toute nouvelle créature sans avoir à acheter un kit entier.

Le robot ainsi créé a donc un but pédagogique. Une question se pose alors sur ce qu'est la robotique pédagogique, ses cibles et ses besoins, les types de robots pédagogiques qui existent et les langages utilisés dans ce cadre. De plus, il existe bien des manières pour un robot de se déplacer et le choix de locomotion entraîne une forme particulière au robot. Comment choisir entre tous les types locomotions possibles et quels sont-ils ? Pour pouvoir se déplacer dans un certain environnement correctement, par exemple sans provoquer de collisions entre le robot et des obstacles, il est nécessaire de capter des informations sur son environnement, mais aussi sur le robot lui-même. Quelles informations utiles au déplacement existent et peuvent être captées par un robot et comment les utiliser ? Enfin, grâce à ses informations, le robot peut agir sur son environnement. De quelle manière et par quel biais un robot interagit-il avec son environnent ?

Il existe plusieurs exemples dans la littérature sur la possibilité d'utiliser un robot comme outil pédagogique. De nombreux robots sont développés et des études ont été réalisées pour tester l'amélioration de la transmission de connaissances dans ce cadre. Au niveau des différentes formes de locomotion, la question a déjà été beaucoup traitée en particulier au niveau des robots terrestres, les différents modes de locomotion sont donc sous-catégorisés (nombre de roues, nombre de pattes, manière de se déplacer...), ces catégories sont étudiées dans des travaux de recherche faisant varier en particulier des cas et contextes d'utilisation. Il existe également une grande littérature sur l'interaction entre le robot et son environnement appliquée à divers capteurs et actionneurs, transmetteurs.

De nombreuses solutions présentées dans cet état de l'art sont appliquées à des contextes particuliers et sont dépendantes de certains aspects matériels. Ces travaux déjà réalisés sont donc à synthétiser et adapter au projet. Il faut en particulier adapter les différentes solutions existantes sur le plan pédagogique pour qu'elles se prêtent aux lycéens. Il y a également un travail à réaliser sur l'adaptation de la structure et de la locomotion du robot aux contraintes de matériels et de coûts faibles.

L'état de l'art commence ainsi par les recherches sur l'aspect pédagogique que peut avoir la robotique, quels sont les attentes et besoins qui caractérisent ce contexte, quel est l'existant et l'importance du langage d'apprentissage. Il est ensuite question des formes et déplacements du robot, les différents types de locomotions qui peuvent s'appliquer à notre projet ainsi que la variété, l'utilité et l'intérêt des simulateurs 3D. L'état de l'art se clôture sur la partie interactions avec l'environnement dans le contexte matériel de notre projet, les possibilités en terme de captation externe et interne de l'environnement et les actions réalisables avec les outils du Kit Ergo Jr.

2 Robotique pédagogique

La robotique pédagogique est une robotique bien particulière. Elle a des caractéristiques qui la distinguent des autres types de robotique. Elle cible une catégorie de la population qui n'est pas professionnelle du domaine. Cela peut être des adultes mais aussi en grande partie des enfants. Cela engendre des besoins qui lui sont bien spécifiques. Pour réaliser les buts qui lui sont fixés, différent types de robots pédagogiques existent, avec, pour chacun, un fonctionnement différent. De plus, de part le fait que la cible de la robotique pédagogique ne sait pas forcément utiliser les langages de programmations utilisés par les informaticiens, il faut trouver un moyen simple d'utiliser les robots et de les programmer.

2.1 Cible(s) et besoins spécifique

La robotique pédagogique est à distinguer de la robotique éducative [4], la robotique éducative regroupe l'ensemble des utilisations de robots dans le milieu de l'éducation. Tandis que la robotique pédagogique ne concerne que l'utilisation des robots à des fins d'enseignement de la programmation, de l'algorithmique et autres connaissances se rapprochant de l'informatique.
La robotique pédagogique peut être utilisée de la maternelle à l'enseignement supérieur. Son utilité est démontrée en 2007 dans le cadre d'une classe d'école élémentaire américaine [1].
La robotique pédagogique soulève un ensemble de contraintes, tout d'abord le robot doit être programmable ou manipulable par les élèves visés de manière à pouvoir transmettre de la connaissance, ça ne peut pas être un simple outil de démonstration avec une boite noîre au niveau du fonctionnement. Cette transmission de connaissance doit être adaptée au niveau d'éducation des élèves, on ne peut pas présenter le même robot à des maternelles, des primaires, des collégiens, des lycéens en espérant le même résultat. Dans le cadre de lycéens, ils doivent être en mesure de programmer le robot, la transmission de connaissance est alors facilitée en particulier par l'observation concrète des conséquences de leurs codes.
Le robot doit ainsi avoir suffisamment de fonctionnalités pour permettre de générer du code intéressant, sans être trop complexe sinon les élèves risquent de ne plus comprendre l'impact de leur travail. Il doit également permettre aux professeurs de mettre en place diverses activités facilement.
Le robot se doit de plus d'avoir des barrières logiciels ou mécaniques empêchant à l'élève de pouvoir blesser quelqu'un ou endommager le robot sans le faire volontairement à minimum, ceci dans une optique de sécurité, de maintient du matériel et de ne pas faire s'en vouloir un élève.
À ces contraintes s'ajoute le besoin d'un prix peu élevé, en effet dans le contexte éducatif actuel l'idée est de faire travailler les élèves par petits groupes. Des petits groupes d'entre 2 et 4 élèves impliquent la présence d'entre 16 et 9 robots dans une classe de lycée, ce qui impose un budget non négligeable pour la plupart des lycées.

2.2 Types de robot

En fonction de l'âge de l'élève cible, et donc des objectifs pédagogiques de son enseignant, on trouve différentes manières de programmer un robot.

Les plus petits élèves de maternelle peuvent apprendre d'une première activité consistant à parler au robot pour qu'il leur obéisse. C'est ce qui est par exemple possible avec le robot blue-bot, doté d'instructions concernant son déplacement (avancer, tourner à gauche, etc.) et qui permet d'associer à chaque action un son, afin que les enfants puissent ordonner oralement au robot ce qu'il doit faire. Cela permet aux enfants de mieux comprendre les liens entre cause et conséquence, et ainsi d'anticiper les mouvements du robot avant même de lui avoir donné une suite d'ordres.

Pour des enfants un peu plus âgés, on peut utiliser un support physique sur lequel ils vont placer une suite d'instructions qu'un robot effectuera. L'utilisation d'un tel support physique pour la programmation du robot est indispensable pour les plus jeunes, pour qui l'utilisation d'une interface numérique reste une barrière. On permet ainsi à de jeunes enfants de se représenter l'espace et le temps et de concevoir de premiers algorithmes pour résoudre un problème de déplacement. Par exemple, le robot blue-bot [5], de manière similaire au robot Primo [6], se programme aussi à l'aide de cartes représentant une action à effectuer. Les enfant insèrent les cartes dans un certain ordre dans un clavier de commande, et observent ensuite le robot réaliser chaque instruction dans l'ordre.

La programmation par bloc est majoritairement présente lorsqu'il s'agit de faire programmer des collégiens et lycéens. Par exemple, des robots tels que mBot ou Thymio sont programmables dans le langage Scratch, développé au Massachusetts Institute of Technology, et ceux de la plateforme poppy le sont en Snap *!*, un autre langage de programmation par bloc développé par l'Université de Californie à Berkeley. Cette manière de programmer possède de nombreux avan-

tages, détaillés en section 2.3.

Enfin, le robot reste un support pédagogique rendant attrayant l'apprentissage d'un langage de programmation, car il permet à un étudiant, programmant le robot puis observant son comportement, d'avoir un retour direct sur ce qu'il a écrit.

2.3 Langage d'apprentissage

De nombreux langages de programmation existent. Chacun d'eux a ses propres spécificités et sont utilisés dans des domaines pour lesquels ils sont fait. Il existe donc des langages dits de bas niveau qui requièrent une connaissance précise du fonctionnement de la machine et qui permettent de manipuler explicitement la mémoire par exemple. Un exemple de ce type de langage est le langage assembleur. Opposés des langages de bas niveaux, il existe des langages dits de haut niveau. Ces langages ne nécessitent pas une connaissance poussée du fonctionnement de la machine et sont plus proches du langage naturel. Ces langages permettent d'écrire des programmes plus facilement. Un exemple de ce type de langage est le Python. Seulement, tous ses langages requièrent des connaissances particulières telles que les règles de syntaxes et de sémantique ou encore du vocabulaire spécifique. C'est pourquoi les langages de programmation visuels sont en expansion, entre autres dans le milieu de l'éducation. Ces langages se basent sur l'assemblage graphique de boîtes entre elles. Par exemple, le langage choregraph permet l'assemblage de boîtes en les reliant entre elles par des flèches et permet de programmer facilement le robot Nao 1.

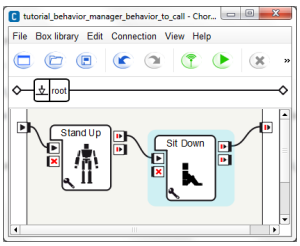

FIGURE 1 – Exemple de code pour Nao grâce à choregraphe.

Un autre exemple de ce type de langage est Blocky ou encore Snap *!*, utilisé sur la plateforme Poppy. Il permet d'assembler des blocs entre eux par le biais de formes qui s'emboîtent. Le nombre de variables devant être utilisé dans une fonction est implicite de part le nombre d'espaces laissé pour cela dans le bloc. Et quelle variable y mettre est aussi implicite de part la forme du trou du bloc. L'avantage de ce type de langage est qu'il a un niveau d'affordance élevé, donc l'utilisateur n'a pas besoin de se demander comment utiliser une boîte car son utilisation est relativement intuitive. De plus, certains de ses langages n'utilisent pas de mots mais des symboles, ce qui permet l'apprentissage de la programmation aux petits élèves qui ne savent pas encore bien lire. Un exemple est le langage Aseba-VPL utilisé par le robot Thymio ou encre le langage Scratch-Junior.

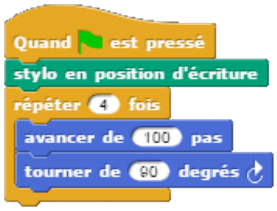

FIGURE 2 – Exemple de code réalisé grâce à snap *!*.

L'utilisation des langages de programmation visuelle ajoute une nouvelle couche par dessus les langages de haut niveau qui sont eux-mêmes une sur-couche des langages de bas niveaux. En effet, le programme visuel doit être traduit en langage de haut niveau (pour Snap *!* c'est le JavaScript) qui lui-même doit être traduit dans un langage compréhensible par la machine. C'est une abstraction supplémentaire sur le fonctionnement d'une machine. Le but ici n'étant pas de comprendre finement le fonctionnement de la machine mais plutôt le fonctionnement des algorithmes, le langage visuel reste une bonne option.

3 Formes et déplacements de robots mobiles

Chaque robot possède une forme qui lui est propre. Cette forme entraîne des contraintes et des possibilités sur le mode de déplacement. Il existe ainsi plusieurs grand types de locomotions. Selon le but que l'on se fixe (ex : être capable d'avancer sur un terrain accidenté) les déplacements du robot peuvent être plus ou moins perfectionnés et le choix du type de locomotion du robot influence ses déplacements. Choisir entre ces différents types de locomotions peut parfois être un vrai casse tête. C'est pourquoi des outils de visualisation 3D permettent parfois de faciliter ce choix en permettant la construction de robots virtuels.

3.1 Types de locomotions

Il existe plusieurs moyens de locomotions utilisés en robotique. Nous nous intéressons, dans le cadre de notre proje,t aux déplacements sur terre, à distinguer des déplacements aériens et marins. Un moyen relativement simple et intuitif est ainsi de faire évoluer le robot sur des roues. Malheureusement, ce moyen peut devenir un peu limitant dans certaines situations. Un autre moyen est donc d'utiliser des pattes. Néanmoins, il existe d'autres types de déplacements comme, par exemple, les roues pattes, une combinaison entre les pattes et les roues, ou encore les chenilles.

3.1.1 Robots à roues

Les robots à roues sont les plus répandus de nos jours. Ce moyen de locomotion possède une grande efficacité dans les environnements créés par l'homme, mais n'est cependant pas adapté aux milieux présentant de grandes irrégularités ou obstacles à franchir.

En comparaison aux robots à pattes, les robots à roues possèdent l'avantage d'avoir un modèle cinématique simple. La configuration des roues reste néanmoins importante pour la manœuvrabilité et la stabilité du robot, comme le montre un papier de l'université de Stasbourg [2].

Il est important pour un robot à roues de se déplacer sans que ses roues ne glissent ou dérapent, ce qui est crucial pour l'odométrie (expliquée à la section 4.1.2). Il faut pour cela que la géométrie du robot impose à tout instant un unique point fixe autour duquel tourne le robot, nommé Centre Instantané de Rotation (CIR). Cette dernière contrainte limite le nombre de configurations possibles à trois catégories principales, schématisées par la figure 3 : les robots de type unicycle, tricycle, et omnidirectionnels.

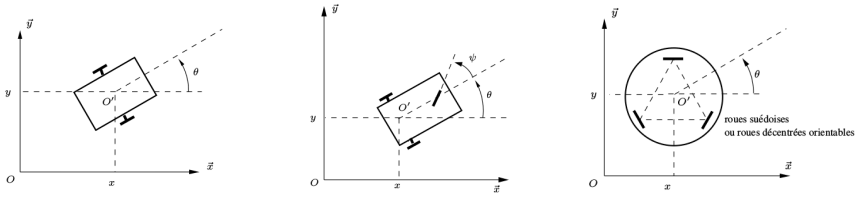

FIGURE 3 – Trois grandes catégories de disposition de roues. De gauche à droite : unicycle, tricycle, et omnidirectionnelle. (source : [2])

Chacune de ces catégories englobe un ensemble de robots pouvant posséder un nombre de roues différent, mais dont la géométrie est équivalente d'un point de vue cinématique. Par exemple, la figure 4 montre l'appartenance des robots de type voiture (à géométrie d'Ackermann) à la catégorie des tricycles.

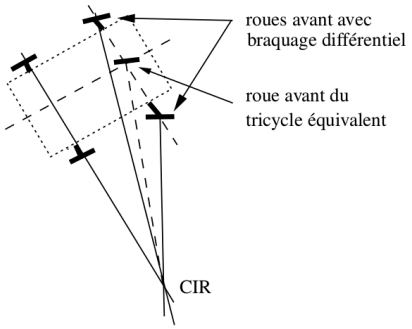

FIGURE 4 – Équivalence cinématique des tricycles et des voitures (source : [2])

La catégorie cinématique contraint les caractéristiques du robot en terme de manœuvrabilité et de stabilité. Ainsi, le choix de la configuration du robot doit être réalisé en fonction de l'usage attendu, chaque catégorie possédant ses avantages et inconvénients.

Robots unicycles : Ce type de robot est très répandu de par sa simplicité cinématique. On impose facilement au robot une vitesse longitudinale et angulaire à partir des moyennes et différences de vitesse de rotation de ses deux roues motrices. Ces robots peuvent poser des problèmes de stabilité, et il est pour cela commun d'ajouter une troisième roue folle en dehors de l'axe des roues motrices.

Robots tricycles : On trouve peu de robots de type tricycles car ils sont très peu stables. En pratique, on utilise des robots à quatre roues de la même classe cinématique : les robots à géométrie de type Ackermann. Comme montré en figure 4, la géométrie Ackermann, celle des voitures, consiste à utiliser deux roues avant pivotantes, mais dotées d'un braquage différentiel, afin que leurs axes de rotation se croisent avec celui des roues arrière en un seul point. On obtient ainsi un robot à grande stabilité mais faible manoeuvrabilité.

Robots omnidirectionnels : Il est possible d'agir indépendemment sur les deux vitesses de translations et sur la vitesse de rotation de ces robots, ce qui leur fourni une très grande manoeuvrabilité. Ces robots ont néanmoins le désavantage d'imposer une plus grande complexité

mécanique, ainsi qu'une odométrie moins fiable.

FIGURE 5 – Exemple de robots unicycles, tricycles, et omnidirectionnels.

L'objectif du projet étant de réaliser un petit robot éducatif à bas coût, il semble que la manoeuvrabilité et la simplicité soient les critères qui importent le plus. Ainsi, les robots tricycles (ou Ackermann) et omnidirectionnels ne semblent pas adaptés, du fait de leur manque de manoeuvrabilité ou de leur trop grande complexité. Il serait cependant tout à fait envisageable de réaliser un robot de type unicycle, en ajoutant une roue folle à l'avant ou l'arrière pour assurer une certaine stabilité.

3.1.2 Robots à pattes

Dans la nature, un grand nombre d'espèces animales se déplacent sur des pattes. Le nombre de pattes sur lesquelles ces animaux se déplacent varie de 1 à un millier. Pourtant, malgré les variabilités entre toutes ses espèces, des allures similaires sont observése. Cela peut laisser à penser que, malgré une variabilité importante, il existe certaines façons de se déplacer avec des pattes qui sont plus efficaces que d'autres.

Lors de la fabrication d'un robot à pattes se pose alors la question de combien de pattes utiliser, comment utiliser chaque patte et comment doivent-elles être conçues afin de pouvoir effectuer des déplacements les plus efficaces tout en restant à un niveau simple d'implémentation. De plus, intuitivement, une patte se constitue de 3 degrés de libertés. Or, il est tout à fait possible de concevoir une patte avec plus de 3 degrés de libertés. Il est également possible de faire des pattes avec moins de 3 degrés de libertés. Par exemple, le mécanisme de Jansen permet de créer une patte avec seulement 1 degré de liberté (voir Figure 6).

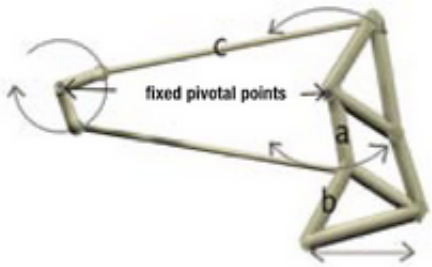

FIGURE 6 – Mécanisme de Jansen pour patte à un degré de liberté

De nombreux robots à pattes existent actuellement. Une grande catégorie des robots à pattes est la catégorie robot bipède dont le robot ASIMO de Honda, ou encore Nao d'Aldebaran font

partie (Voir figure 7). La marche d'un robot bipède est un challenge, entre autres car le nombre limité de pattes (2) engendre des difficultés au niveau de la stabilité. En terrain accidenté, ces difficultés sont largement augmentées.

FIGURE 7 – A gauche : robot ASIMO de Honda - A droite : robot Nao de Aldebaran.

Certains robots sont construits spécialement afin de pouvoir évoluer sur des terrains extrêmement accidentés comme Big dog et Little dog de Boston Dynamics.

FIGURE 8 – A gauche : Big dog - A droite : Little dog de Boston Dynamics.

Un autre parti pris est de concevoir un robot qui sera facilement renversé en évoluant sur des terrains accidentés car l'accent n'est pas mis sur la stabilité. Pour compenser, ils ont la capacité de se relever pour se remettre en mouvement. Ou encore, faire en sorte que le robot soit capable de se déplacer même après la perte d'un de ses membres. D'autres encore sont fabriqués en se basant sur le modèle d'un animal vivant, par exemple bionicAnts de Festo et AIBO de Sony (Voir figure 9). Le but alors n'est pas forcément qu'il soit capable de se déplacer sur tout type de terrain. C'est souvent le cas de robots pédagogiques tels que Metabot. Le robot bionicAnts reproduit le caractère collaboratif entre fourmis et AIBO est quant à lui un robot de compagnie.

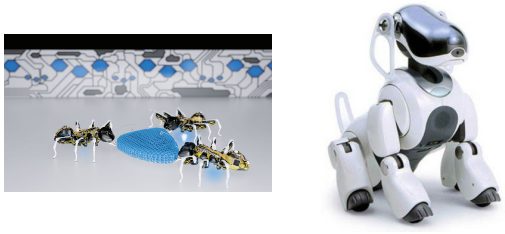

FIGURE 9 – A gauche : trois bionicsAnts de Festo collaborant - A droite : AIBO de Sony.

Enfin, un autre type de robot à pattes qui existe est un robot un peu particulier, le Strandbeest de Theo Jansen, un artiste (Voir figure 10). Ce robot est fabriqué uniquement avec des tubes en plastique jaune et se déplace sur les plages grâce à l'énergie du vent.

FIGURE 10 – Un Strandbeest de Theo Jansen.

Les robots à pattes cités ci-dessus contrôlent leurs pattes de façons différentes. Il en est de même pour la stabilité, même s'il existe des critères génériques. Un premier critère est le critère de stabilité statique qui suppose que les mouvements sont suffisamment lents pour négliger les aspects dynamiques. Un autre est le ZMP (Zero Moment Point), qui est un point associable à chaque configuration du robot et qui doit se trouver dans son polygone de sustentation pour assurer sa stabilité.

Afin de diriger un robot à pattes, chaque robot possède son propre contrôleur qui permet de faire bouger les pattes d'unz certaine manière. Dans sa thèse, Grégoire Passault [8] décrit son contrôleur permettant de diriger les pattes d'un robot à pattes, il teste le contrôleur pour toute une gamme de robots à pattes, en faisant varier le nombre de pattes, l'agencement des degrés de libertés et leur forme.

Dans le cadre du projet, il est tout à fait envisageable de construire un robot à pattes. Étant donné qu'il a des fins pédagogiques, il est préférable que le robot soit robuste et donc stable. L'option d'un robot bipède est donc à éviter. De plus, le but n'étant pas de fabriquer un robot capable d'évoluer en terrain accidenté, il n'y a pas de besoin de faire un contrôleur très évolué. La construction d'un robot avec au moins 4 pattes pour assurer la stabilité et un contrôleur permettant d'évoluer sur un sol plat est complètement envisageable.

3.1.3 Robots à roues pattes

Les robots hybrides «roues-pattes» combinent les deux modes de locomotion à pattes et à roues, dans le but d'obtenir des déplacements à grande mobilité, sur des terrains accidentés semés d'obstacles. [9] En effet, en fonction des propriétés du terrain à parcourir, les robots hybrides peuvent utiliser soit la marche, soit le roulement, mais aussi des modes de locomotions plus complexes, tel que le péristaltisme, consistant en un déplacement de chaque patte, mais sans perte de contact avec le sol. Le franchissement de grands obstacles est aussi possible avec beaucoup plus de facilités que pour des robots à roues, les pattes pouvant franchir une à une l'obstacle tout en maintenant le robot dans une configuration géométrique stable.

FIGURE 11 – À gauche, le robot Hylos (CNRS), et à droite le Handle (Boston Dynamics)

La mise en place d'un tel type de locomotion et de démarches complexes semble nécessiter beaucoup de moyens techniques. Cependant, une telle architecture dans le but de réaliser des

robots pédagogiques présenterait un certain intérêt : de tels robots auraient un aspect attrayant, et pourraient potentiellement permettre une grande diversité d'activités.

3.1.4 Autres robots

La robotique mobile est rapidement confrontée au problème du déplacement sur des surfaces inégales et au franchissement d'obstacles. Les robots à chenilles ont l'avantage de proposer des solutions plus faciles à mettre en place que des robots à pattes pour résoudre ce genre de problèmes, même si plus limités dans leurs usages. Ces robots peuvent prendre différentes formes et avoir différents types de chenilles, Mr Jean-Luc Paillat en a proposé une classification [7]. Ainsi il y a des robots à géométries fixes (figure 12 - gauche) et des robots à géométries variables qui peuvent être soit à chenilles non-déformables (figure 12 - centre) soit à chenilles déformables (figure 12 - droite).

FIGURE 12 – Trois robots à chenilles, de gauche à droite : le TALON de QinetiQ à géométrie fixe, le VGTV d'Inuktun à chenilles non déformables et les chenilles déformables du Viper Robot de Galileo.

D'autres aspects de la locomotion ont été testé, avec par-exemple les robots inspirés par les serpents. Ces derniers sont basés sur une structure hyper-redondante qui leur permet différents modes de déplacements, on retrouve en particulier dans la nature l'ondulation latérale, le mouvement en accordéons et le déroulement latéral. On peut citer par-exemple les robots "modsnake" (figure 13 - gauche) développés par des équipes de la Carnegie Mellon University, capables de monter sur une jambe ainsi que réaliser du déplacement latéral et ondulatoire. Ou plus basiquement les robots de Gavin miller dont voici la deuxième version qui se déplace par ondulation (figure 13 - droite). Ces robots nécessitent des formes et textures particulières si l'on souhaite les voir passer des obstacles. De plus, ils sont beaucoup moins évidents à comprendre en terme de mouvement et de programmation que les robots à roues, chenilles ou pattes.

FIGURE 13 – A gauche un robot de la Carnegie Mellon University. A droite un robot deuxième génération de Gavin Miller.

Dans le cas des robots à chenilles comme des robots serpents, ils nécessitent l'achat ou la fabrication de beaucoup plus de matériel que ce qui est à disposition dans le kit Ergo Jr. De

plus, ils ne permettent pas la richesse de déplacements des robots à patte.

3.2 Simulations modèles 3D

En robotique, il est courant d'avoir recours à des visualiseurs 3D afin de se représenter le robot en cours de développement, et d'étudier son comportement au sein d'une simulation physique, ce qui permet, lors de la conception de nouveaux algorithmes, par exemple, de réaliser des premiers tests de manière rapide et sure, sans risquer d'endommager le vrai robot. C'est ce que fait par exemple Grégoire Passault [8] avec Metabot Studio. Pour ces raisons, il existe de nombreux logiciels permettant de créer un robot (en terme de structures, articulations, capteurs, programmes...) et de le visualiser au sein d'un environnement pouvant être doté d'une physique plus ou moins fidèle à la réalité.

La plateforme Poppy utilise V-REP pour créer les robots et les simuler. Une comparaison détaillée de V-REP [11] à deux autres simulateurs open-sources répandus, Gazebo et ArGoS, montre que V-REP est un simulateur plus complexe et gourmand en ressources, ce qui peut le rendre désavantageux si l'on cherche à simuler le comportement de plusieurs robots en même temps. Cependant, V-REP permet de modéliser de nouveaux robots de manière très détaillée, et de les faire évoluer dans un environnement à la physique développée.
Dans le cadre de notre projet, visant à créer une nouvelle créature, la possibilité de créer en détail les pièces du robot au sein de V-REP rend légitime l'utilisation de ce logiciel.

4 Interactions avec l'environnement

La forme et les déplacements du robot sont à lier à son interaction avec l'environnement. Le robot, pour pouvoir réaliser certaines actions et se déplacer, a besoin de détecter et connaître certains éléments sur son environnement et sur lui-même. Ceci est assuré par des capteurs d'éléments externes et internes. Là où les capteurs permettent de récupérer de l'information sur l'environnement, le robot doit également pouvoir agir sur cet environnement. Ces actions sont réalisées grâce à des actionneurs, des émetteurs et autre composants permettant de modifier l'environnement et le robot lui-même. La recherche est ici portée en particulier sur les éléments présents dans le Kit Ergo Jr, leur intérêt et les limites qu'ils posent.

4.1 Captation de l'environnement

Les capteurs du robot sont séparables en deux catégories, ceux externes qui visent à capter l'environnement qui entoure le robot, et ceux internes qui visent à récupérer des informations sur l'état du robot. Les deux combinés permettent au robot de récupérer les informations nécessaires à la réalisation d'actions.

4.1.1 Externe

Vision : Le kit Ergo Jr est muni d'une caméra, cette dernière permet de détecter l'environnement externe. Le programme de contrôle du robot étant développé sous Python, qui est installé sur la Raspberry PI, l'intérêt est ici porté sur les librairies disponibles pour le traitement d'images dans ce langage. Il existe en Python deux principales librairies pour le traitement d'images.

La librairie Python Imaging Library (PIL), développé depuis 1995, elle ne dispose plus de mises à jours depuis 2011. Mais un fork créé par la suite, qui se nomme Pillow, permet de réaliser des manipulations basiques d'images et continue à être développé.

La deuxième librairie principale est Open Computer Vision (OpenCV). C'est une bibliothèque graphique libre développée depuis 2000 (son support est actuellement assuré par Intel). Elle possède en particulier des outils avancés de traitement d'images, de vidéos et des algorithmes d'apprentissages classiques. Elle est disponible sous C++, Python et Java. Elle est largement utilisée dans le monde de la recherche du fait de ses nombreuses fonctionnalités.

OpenCV a de nombreux avantages face à Pillow, il dispose de plus nombreuses fonctionnalités plus avancées, en particulier pour le traitement de vidéos. OpenCV est également plus rapide pour des tâches égales (en moyenne trois à quatre fois plus rapide) [10]. Mais en contrepartie, OpenCV est plus lourd que Pillow. Ainsi, il convient plutôt d'utiliser Pillow si l'on a besoin uniquement de fonctions de traitement d'images et qu'il est nécessaire d'avoir quelque chose de léger. Dans le cadre du projet, il y a un potentiel besoin d'analyse de vidéo en temps réel, OpenCV est donc plus adapté.

Il existe d'autres capteurs externes qui auraient pu être intéressants à utiliser : capteurs tactiles, capteurs de proximité, capteurs de distance. Ces derniers ne seront pas traités ici car ne sont pas présents dans les kits Ergo Jr.

4.1.2 Interne

Odométrie Le kit Ergo Jr est composé en particulier d'un ensemble de servomoteurs (les XL-320 de dynamixel). C'est grâce à eux que s'effectue l'ensemble des mouvements du robot.

Ces servomoteurs permettent une mesure de leur position avec une résolution d'angle de 0,29 degrés. En plus de cette limite de précision, des incertitudes notables peuvent apparaître, en particulier à cause des glissements du robot.

Ces incertitudes ne sont pas forcément problématiques dans le cadre des déplacements du robot du projet mais si c'était le cas il existe des méthodes pour les compenser.

L'une d'entre elles est explicitée par F. Chenavier, I. Lecoeur Taibi et J.L. Crowley [3], elle est basée sur l'utilisation d'une caméra monoculaire pour réduire les incertitudes . Le principe est d'utiliser du traitement d'images pour estimer la distance à des objets fixes en temps réel et de les combiner aux résultats obtenus par odométrie. Cette technique de combinaison pour réduire les incertitudes est applicable avec d'autres capteurs basés sur la détection d'éléments fixes dans l'environnement.

4.2 Actions sur l'environnement

Grâce à sa capacité à capter son environnement, un robot peut agir dessus en fonction de son but. Il existe plusieurs moyens pour un robot de réaliser des actions sur son environnent. Le premier moyen est par le biais de ses membre qui sont souvent actionnés par des moteurs. Par exemple, actionner une pince pour qu'elle attrape des objets. Il en existe bien d'avantages, notamment grâce à du son ou de la lumière.

4.2.1 Moteurs

Le kit ErgoJr est un kit permettant de construire son propre robot à bas coût. C'est pourquoi les servomoteurs utilisés dans ce kit ne doivent pas avoir un prix trop important. Tout d'abord, les servomoteurs ne sont pas de simples moteurs. Ils contiennent également des engrenages permettant d'augmenter le couple du moteur, un encodeur et un circuit de contrôle qui permettent de maintenir une position.

Pour le kit Ergo Jr, les servomoteurs choisis sont les XL-320. Ils ont la possibilité d'être branchés les uns à la suite des autres afin de faciliter l'assemblage de petits robots. Ils communiquent par liaison série asynchrone Half Duplex. Ils sont légers (16.7g) et peuvent appliquer un couple de 3.98 kg.cm. Ils font partis des servomoteurs à bas coût (autour de 22 euros pièce).

En comparaison, pour le même ordre de prix, il existe également le servomoteur Hitec HS-485HB. Il a un poids bien plus important (45g) et un couple plus important également (4.8kg.cm). Il y aussi le servomoteur à rotation continue de Futaba. Il pèse également 45g et a un couple de 3.4kg.cm.

Le XL-320 a donc l'avantage d'être léger et d'avoir malgré tout un couple satisfaisant. De plus, la possibilité d'en brancher plusieurs en série est utile lors de la fabrication d'un bras robotique. Cependant, son poids le rend plus fragile, ce qu'il faut prendre en compte lors de la conception par des mécanismes logiciels empêchant d'aller au delà des limites par exemple.

Afin d'utiliser facilement des servomoteurs, des librairies ont étés créées. Elles permettent de

parler au servomoteur sans avoir à coder de nouveau des fonctions usuelles et sans avoir a se préoccuper de quel signal envoyer, à quelle fréquence et pendant combien de temps. Une librairie populaire est servo, une librairie d'Arduino. Elle permet de contrôler jusqu'à 12 servomoteurs avec la carte Arduino UNO. Une autre librairie est la librairie PyPot développée par l'équipe Flowers de l'INRIA, conçue spécialement pour les servomoteurs Dynamixel et développée dans le langage Python.
Dans le cadre de l'ErgoJr, la librairie utilisée est donc la librairie PyPot, qui a l'avantage d'être dans le langage de programmation utilisé dans le projet est d'être faite explicitement pour les servomoteurs utilisés.

4.2.2 Autres (Son et lumière)

Un robot peut agir sur son environnement d'une autre manière que grâce à ses membres. Par exemple, il peut agir grâce à du son ou encore de la lumière.
Il est en effet possible de brancher des écouteurs ou des enceintes à la Raspbrry grâce à sa prise jack. Cela donne la possibilité, par exemple, de faire parler le robot ou encore d'émettre des sons ou de la musique.
De plus, les moteurs utilisés dans le kit ErgoJr possèdent 3 LEDs couleur programmables, une de chaque couleur élémentaire (rouge, bleue, verte). L'association des trois permettant de faire afficher une couleur comme le jaune par exemple. En pratique, cela donne une possibilité d'afficher 8 couleurs différentes. Étant donné qu'un ErgoJr possède 6 moteurs, cela lui donne un nombre de combinaisons possibles raisonnable. Les LEDs peuvent, par exemple faire passer un message suivant un code couleur, en indiquant que si une des LEDs d'un moteur s'allume en rouge, c'est que le moteur surchauffe.
Dans le cadre de ce projet, les actions sur l'environnement concernant le son et la lumière sont toutefois minoritaires en comparaison des actions effectuées grâce aux moteurs. Cela n'empêche pas de devoir y réfléchir car, même s'ils ne sont pas une priorité, ils peuvent apporter un plus au robot non négligeable.

La combinaison des capteurs et des actionneurs permet de créer de l'interaction entre l'homme et la machine. Ainsi, selon les séquences pré-programmées dans le robot, les étudiants peuvent et doivent réaliser des codes plus ou moins poussés et exigeants pour mettre en oeuvre des fonctionnalités intéressantes. Il est important de laisser de la liberté dans l'utilisation de ces capteurs et actionneurs pour permettre aux étudiants et professeurs de créer des fonctionnalités et de s'approprier le robot. Dans le même temps, il est important de ne pas laisser tout accessible trop facilement car les composants ont des limites qu'il ne faut pas dépasser qui sont liées à eux-mêmes, et à l'interaction entre eux et dans leur environnement.

5 Conclusion et perspectives

L'ensemble des éléments traités dans cet état de l'art a permis de déterminer la forme que prend le robot de ce projet, à la fois au niveau de l'aspect pédagogique, hardware et software.

Il ressort des recherches que l'interface la plus adaptée, pour permettre aux lycéens d'apprendre à programmer en manipulant le robot, est l'utilisation d'un langage de programmation graphique.

Au niveau de l'aspect mécanique et électronique du robot, il est mis en avant la nécessité d'avoir un robot à bas coût dans le cadre du milieu éducatif visé.

Parmi l'ensemble des moyens de locomotion terrestres traités, c'est celui basé sur des pattes qui est le plus à même de correspondre au projet de robot mobile. Celui-ci permet la création d'un robot basé sur 3 kits Ergo Jr avec peu d'impressions de pièces 3D supplémentaires. De plus, ce moyen de locomotion est attrayant et permet la production de multiples activités par les professeurs.

Les capteurs sur le robot sont déjà fixés, néanmoins la comparaison des librairies pour réaliser le traitement de l'image a montré qu'il est préférable d'utiliser OpenCV dans le cadre du projet.

Il est aussi possible de coupler les informations sur la position des moteurs au traitement des images de l'environnement pour obtenir des déplacements précis.

Enfin, le robot peut agir sur son environnement au travers de l'utilisation des servomoteurs qui sont contrôlés à l'aide de la librairie PyPot. Il serait également possible de transmettre de l'information à la fois sonore et visuelle aux étudiants.

6 Bibliographie

Références

[1] Bradley S. BARKER et John ANSORGE. "Robotics as Means to Increase Achievement Scores in an Informal Learning Environment". In : *Journal of Research on Technology in Education* 39.3 (mar. 2007), p. 229–243. ISSN : 1539-1523, 1945-0818. DOI : 10.1080/15391523.2007.10782481. URL : http://www.tandfonline.com/doi/abs/10.1080/15391523.2007.10782481 (visité le 23/11/2018).

[2] Bernard BAYLE. *Robotique mobile*. URL : http://eavr.u-strasbg.fr/~bernard/education/3a_robmob/3a_robmob_poly.pdf (visité le 25/11/2018).

[3] F. CHENAVIER, I. LECOEUR TALBI et J.L. CROWLEY. *Estimation de la Position d'un Robot par Odométrie et Vision Monoculaire*. URL : https://docs.poppy-project.org/fr/ (visité le 26/10/2018).

[4] "La robotique éducative - Institut Français de l'éducation". In : (2017), p. 25. URL : http://ife.ens-lyon.fr/ife/recherche/numerique-educatif/robotique-educative/robotique-educative-1/dossier-de-capitalisation-robotique-educative-decembre-2017.

[5] ORNA. *Blue-Bot*. URL : http://inshea.fr/sites/default/files/fichier-orna/EG_Blue-Bot_0.pdf (visité le 25/11/2018).

[6] ORNA. *Primo*. URL : http://www.inshea.fr/sites/default/files/fichier-orna/EG_Orna_Primo_0.pdf#overlay-context=fr/users/jramatchandran (visité le 25/11/2018).

[7] Jean-Luc PAILLAT. "Conception et contrôle de robots à géométrie variable : applications au franchissement d'obstacles autonome". In : (), p. 154.

[8] Gregoire PASSAULT et al. "Metabot : A Low-Cost Legged Robotics Platform for Education". In : *2016 International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. 2016 International Conference on Autonomous Robot Systems and Competitions (ICARSC). Bragança, Portugal : IEEE, mai 2016, p. 283–287. ISBN : 978-1-5090-2255-7. DOI : 10.1109/ICARSC.2016.52. URL : http://ieeexplore.ieee.org/document/7781990/ (visité le 26/10/2018).

[9] Jarrault PIERRE. "Optimisation des capacités de franchissement des robots mobiles hybrides "roues-pattes"". In : (), p. 137.

[10] *PIL vs Opencv — Kaggle*. URL : https://www.kaggle.com/vfdev5/pil-vs-opencv (visité le 26/11/2018).

[11] Lenka PITONAKOVA et al. "Feature and Performance Comparison of the V-REP, Gazebo and ARGoS Robot Simulators". In : *Towards Autonomous Robotic Systems*. Sous la dir. de Manuel GIULIANI, Tareq ASSAF et Maria Elena GIANNACCINI. Lecture Notes in Computer Science. Springer International Publishing, 2018, p. 357–368. ISBN : 978-3-319-96728-8.