



# RAPPORT DE MODAL

## PHY43M06EP: Robotique

---

FRAMBOURT MATEÏS



INSTITUT  
POLYTECHNIQUE  
DE PARIS

# TABLE DES MATIÈRES

---

<b>Introduction .....</b>	<b>1</b>
<b>1 - Base commune .....</b>	<b>2</b>
1.1 - Base mécanique .....	2
1.2 - Encodeurs .....	3
1.3 - PID .....	3
1.4 - Roues holonomes .....	4
<b>2 - Robot suiveur .....</b>	<b>5</b>
<b>3 - Robot tank .....</b>	<b>6</b>
<b>Conclusion .....</b>	<b>7</b>
<b>Bibliographie .....</b>	<b>8</b>
<b>Annexe A - Fiche d'évaluation du stagiaire .....</b>	<b>9</b>

# INTRODUCTION

---

Je suis arrivé dans ce modal un peu par hasard. Au départ je m'étais inscrit pour le ModAL FPGA car c'était quelque chose qui me semble très versatile mais un peu obscur pour le découvrir en auto-didacte. C'est donc avec surprise que j'apprend la disparition de ce ModAL dans une fusion avec le ModAL de Robotique. Au final, ça ne m'a pas trop gêné car c'était mon deuxième vœu. J'avais une légère expérience en robotique. En effet, j'ai participé de 2012 à 2021 aux compétitions de robotique junior (robot radioguidés) de planètes sciences et j'aide encore marginalement dans l'équipe familiale. J'avais donc une certaine connaissance des composants et de leurs utilisations mais je ne connaissais presque rien dans le contrôle et la commande haut niveau. Mon objectif dans ce cours était donc d'apprendre cette partie pour pouvoir ensuite l'appliquer au Binet X-Robot qui est en manque crucial de RH et de compétence.

D'un point de vue plus académiques, l'objectif final est de réaliser un robot « tank » pour pouvoir travailler sur l'anticipation de mouvement de l'adversaire. L'objectif était donc de réussir à pointer une cible sur l'adversaire tout en se déplaçant dans un environnement encombré. Pour cela nous avions à notre disposition :

- Roues holonomes
- Moteurs avec encodeurs
- Raspberry pi 4
- Cartes moteurs I<sup>2</sup>C Groves
- Caméra grand angle 5MP
- Servomoteurs

Cependant, avant de réaliser le projet final il est nécessaire de réaliser un suivi de ligne fonctionnel. Dans ce sens j'ai passé (malheureusement) la majeure partie des séances (8/10) sur ce projet initial.

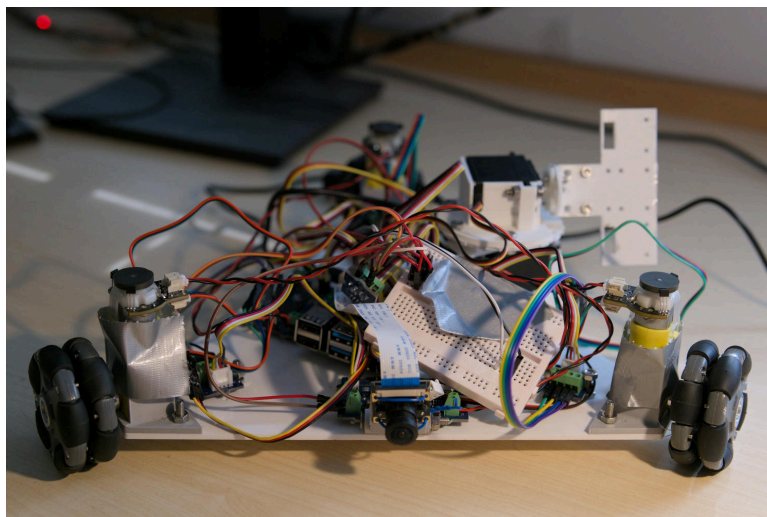


Fig. 1. – Vue d'ensemble du robot final

# 1

## BASE COMMUNE

### 1.1 - BASE MÉCANIQUE

Dans l'optique de pouvoir réutiliser un maximum de composants (tant mécaniques que logiciels), J'ai commencé par modéliser précisément les composants à ma disposition. Ensuite j'ai dessiné les supports des moteurs pour avoir une emprise au sol minimale (ils ont ensuite été imprimés au DrahiX). Enfin j'ai dessiné la plaque principale qui devait permettre d'avoir tous les composants à plats avec l'ensemble des trous déjà percés (elle a été découpée au laser du DrahiX). Cependant, la breadboard et la batterie ont été oubliées ce qui a mené à un montage relativement complexe à suivre et à modifier. L'objectifs n'a donc pas complètement été rempli.

J'ai choisi d'utiliser 3 moteurs en triangles pour 2 raisons :

- Il y a 3 degrés de liberté sur le robot donc avoir 3 moteurs pourrait donc nous permettre de facilement faire de la cinématique inverse exacte (ce n'est pas la méthode qui a finalement été utilisée)
- Les placer à 120° permet de avoir une symétrie et donc on peut avoir plusieurs « face avant » du robot et il suffit juste de faire un décalage au niveau des consignes pour utiliser par exemple un autre outil

D'autre part, la caméra est montée sur un pivot car elle possède un grand angle de vue. Il est donc possible de régler l'étendue du champ de vision en inclinant la caméra pour qu'elle ne soit pas perturbée par des objets extérieurs.

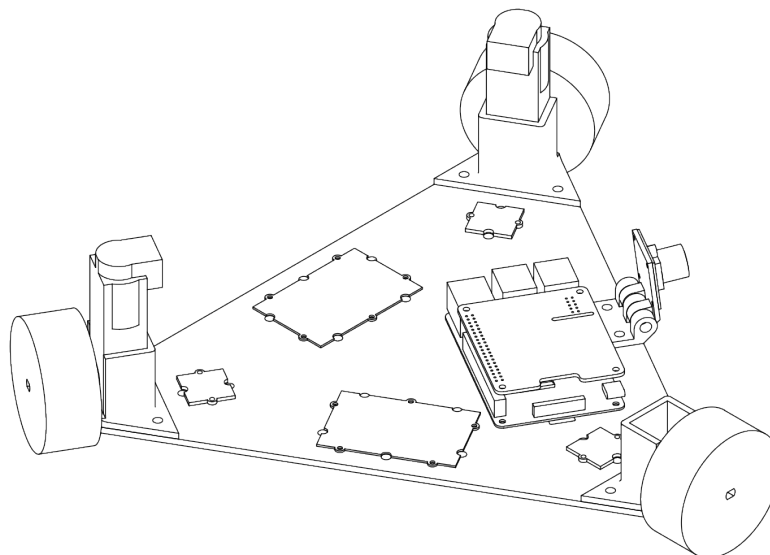


Fig. 2. – Vue isométrique de l'assemblage basique

## 1.2 - ENCODEURS

Rapidement, les encodeurs sont des paires de capteur à effet hall en quadrature de phase autour d'un aimant à 14 pôles radiaux de manière à ce qu'ils donnent 7 périodes de la Fig. 3 par rotation. Pour interpréter le signal, j'ai fait une librairie CMake qui attache une interruption aux 2 pins de l'encodeur et appelle une fonction très simple :

```
uint8_t phase = identifier_phase();
if (phase == (lastPhase + 1) % 4)
{
    pos++;
}
else if (phase == (lastPhase - 1) % 4)
{
    pos--;
}
lastPhase = phase;
```

et pour identifier la phase on fait appel à une table de correspondance :

```
uint8_t const LUT[] = {0, 1, 3, 2};
return LUT[2 * digitalRead(pinA) + digitalRead(pinB)];
```

J'ai essayé pendant plusieurs séances d'implémenter cette librairie de manière conforme à la POO mais je n'arrivais pas à faire une fonction statique qui appelle une méthode. Finalement, j'ai réussi à l'implémenter pour une autre librairie dont je parlerais plus loin.

Globalement, le problème principal de cette mesure est qu'elle donne précisément la position mais pas la vitesse. Il faut ensuite dériver numériquement ce qui ne donne pas de très bon résultat car la position est une fonction en escalier. Or je souhaitais asservir les moteurs en vitesse ce qui rend l'asservissement un peu instable.

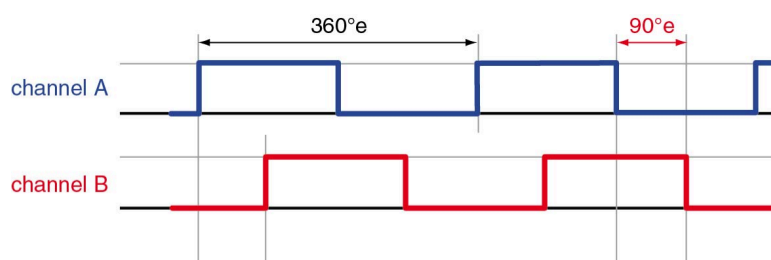


Fig. 3. – Signal en sortie des encodeurs (© Maxon)

## 1.3 - PID

J'ai ensuite créé une librairie CMake générique pour les PID, c'est-à-dire un contrôleur générique à trois composante (proportionnelle, intégrale, dérivée). Celui-ci devait permettre d'avoir un contrôle précis et réactif mais les premiers moteurs étaient trop peu puissants et ne reposaient presque que sur la composante intégrale. En effet, même en réglant la composante proportionnelle à la limite de l'instabilité, la valeur atteinte

en vitesse était très éloignée de la valeur désirée. D'autre part, l'entrée du moteur étant proportionnelle à sa vitesse, il fallait « intégrer » la commande envoyée au moteur. Dans le cas contraire, dès que l'erreur est nulle, la consigne envoyée au moteur est nulle et donc le moteur s'arrête. Ce n'est bien sûr pas le comportement voulu. Dans ce sens, l'ajout d'une commande constante au moteur permet au PID de compenser l'erreur que celle ci aurait seule. (On approxime donc le moteur à une réponse linéaire à la consigne et on corrige l'erreur d'approximation par PID). Cependant, même avec toute ces modifications, le moteur ne répondait pas correctement à faible commande. En effet, les effets de frictions sèches étaient trop important. Le modèle a donc été adapté pour les prendre en comptes (en augmentant la consigne artificiellement à faible commande). Pour limiter l'inertie de la composante intégrale, l'intégrale sature à partir d'une limite arbitraire.

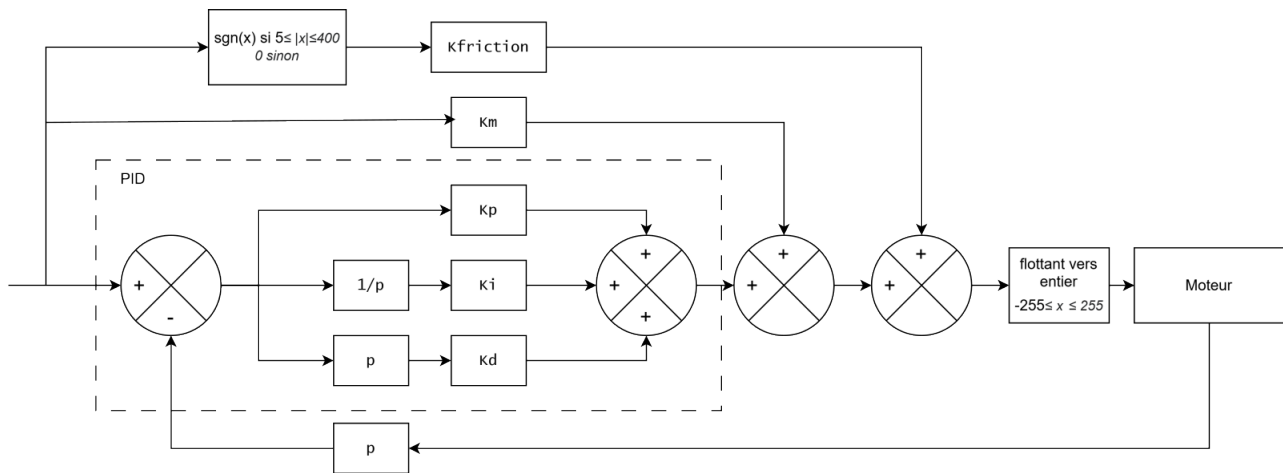


Fig. 4. – Asservissement de chaque moteur

## 1.4 - ROUES HOLONOMES

Les roues holonomes sont des roues qui n'imposent pas de contrainte normale à la roue. C'est dire qu'elle glisse sur la direction normale à la roue. On peut donc considérer uniquement comme un ajout de vecteur (il faut cependant )

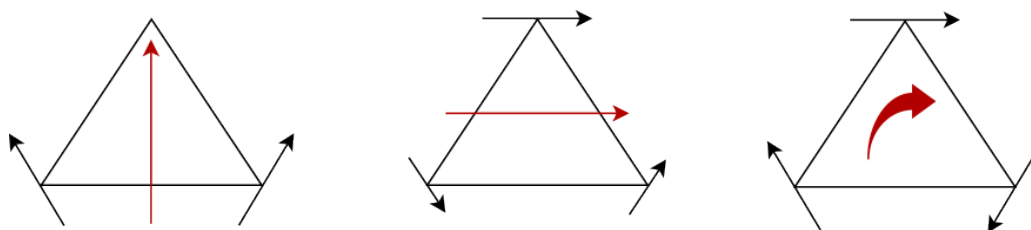


Fig. 5. – Vecteurs considérés pour les roues holonomes

## 2

# ROBOT SUIVEUR

---

# 3

## ROBOT TANK

---

[1]



## CONCLUSION

---

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aequaleam animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

## BIBLIOGRAPHIE

---

- [1] L. Liu, X. Wang, X. Yang, H. Liu, J. Li, et P. Wang, « Path planning techniques for mobile robots: Review and prospect », *Expert Systems with Applications*, vol. 227, p. 120254, 2023, doi: <https://doi.org/10.1016/j.eswa.2023.120254>.

# ANNEXE A

## FICHE D'ÉVALUATION DU STAGIAIRE

---

Yeah j'ai eu que des A partout trop bien, je suis un.e super stagiaire.