



## EA08 – Mise en Œuvre de Systèmes Mécatroniques

TP Pixy

SALGUEIRO Alexandre, CORREIA André, AUZARY Frédéric

Printemps 2023

## Introduction

On observe depuis la révolution industrielle dans les années 1760, une hausse du nombre de robots et autres machines mécatroniques dans l'industrie. Ces ajouts permettent de remplacer la main d'œuvre humaine dans des tâches à faible valeur ajoutée et parfois pénibles ou difficilement réalisables par l'homme. Ces changements ont permis une hausse de la productivité en usine impliquant une baisse des coûts de production et par conséquent, des coûts de vente aussi. Au fur et à mesure des révolutions industrielles (industrie 2.0, 3.0 ou encore 4.0) l'automatisme a pris de plus en plus de place au sein des chaînes de production ou des outils de création. C'est pour ces raisons que le métier d'ingénieur a dû s'adapter. Un ingénieur mécanique doit maintenant avoir des connaissances en électronique pour être capable de concevoir non plus de simples systèmes mécaniques mais de vrais systèmes mécatronique fonctionnant automatiquement grâce à un code implémenté sur un microcontrôleur à bord du système.

Dans le cadre de l'UE EA08, nous devons réaliser un projet de 6 mois portant sur un système mécatronique. Le projet est à réaliser en groupe de 3 et à la suite du travail réalisé par les étudiants ayant fait EA08 au dernier semestre. Le but de ce projet est de se familiariser avec l'utilisation de micro contrôleurs, d'utiliser les connaissances acquises en automatisme et en asservissement à un projet concret ainsi que d'apprendre à appliquer nos compétences en programmation afin de faire réaliser au robot ce que l'on souhaite. Le projet est découpé en deux parties majeures, la compréhension du robot et du travail réalisé par les anciens étudiants et l'ajout de nouvelles fonctionnalités. Afin de pouvoir ajouter des fonctionnalités, nous devons comprendre quelles fonctionnalités sont déjà implémentées, et comment elles l'ont été. C'est donc l'objet des premières séances de travaux pratiques jusqu'à au moins la moitié du projet. Dans un premier temps, nous vous présentons le robot en général ainsi que le travail réalisé par les anciens groupes. Puis nous vous détaillerons les changements apportés au robot et l'ajout des nouvelles fonctionnalités décidées par notre groupe.

## Sommaire

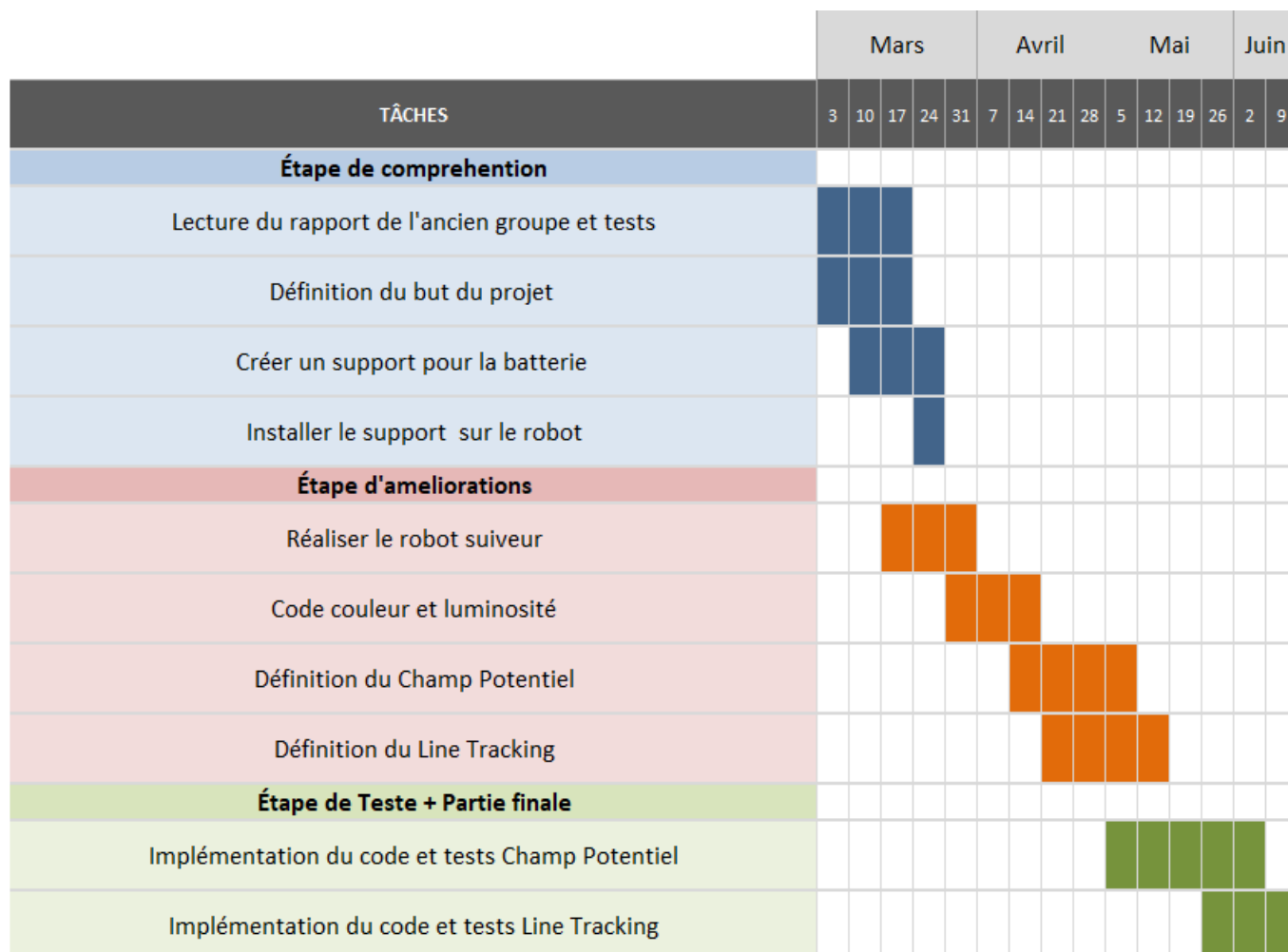
### Table des matières

Introduction.....	2
Sommaire .....	3
Organisation.....	3
Présentation du Robot .....	4
Développement du Projet.....	6
Mise en Marche du Robot .....	6
Orientation de la Caméra .....	7
Importation des Informations de la Caméra .....	7
L'Équilibre dynamique et l'accéléromètre .....	7
Axes d'Amélioration et Ajout de Fonctionnalités.....	8
Réalisation d'un porte batterie en impression 3D .....	9
Code Couleurs et Luminosité.....	13
Contour des Obstacles.....	13
Suivie de ligne .....	17

## Organisation

Nous avons décidé dans un premier temps de nous familiariser avec le robot en équipe de trois. Nous avons analysé et modifié le code pour faire fonctionner le robot ensemble avant de nous diviser en plusieurs groupes pour faire avancer plus rapidement le projet. Un étudiant s'est occupé de la partie mécanique du projet, un autre de l'ajout de l'algorithme du contournement d'obstacles et un troisième sur la partie du line tracking. Bien-sûr, lorsqu'un de nous avait un problème sur sa partie, les autres pouvaient lui venir en aide.

Nous avons également réalisé un diagramme de Gantt afin d'en savoir plus sur l'avancée de notre projet et d'en déduire si nous fournissons un travail suffisant. Vous pouvez trouver ci-dessous une capture d'écran du diagramme de Gantt complété :



## Présentation du Robot

On nous a présenté pendant la première séance de travaux pratiques, le robot sur lequel nous devons travailler, tous les trois, durant le semestre. Il s'agit d'un robot composé de deux roues de vélos reliés par une armature en profilé aluminium avec des plaques d'aluminium sur lesquelles sont fixés différents composants électroniques. On y retrouve une carte Arduino, deux plaquettes de connexion électronique, ainsi qu'un réseau de câbles électroniques assez vaste. On observe sur l'avant du robot et dans un système mécanique complexe, monté sur un profilé : une caméra Pixy. Une Pixy est un capteur compatible avec des micro-contrôleurs comme Arduino ou Raspberry pi. Le capteur ne détecte que les

images dont on a besoin et nous renvoie les données capturées sur 6 canaux différents : UART, SPI, I2C, USB, digital et analogique. La Pixy fonctionne sur le principe de détection des couleurs (jusqu'à 7 couleurs différentes) et des formes. Sur les roues sont montés deux moteurs à courant continu reliés à la carte Arduino ainsi que deux codeurs incrémentaux. Les roues, les codeurs et les moteurs sont reliés entre eux par une courroie dentée. Une place est disponible pour accueillir la batterie mais une batterie rechargeable est utilisée et n'a pas de place attribuée.

La fonctionnalité principale du robot lorsqu'on l'a récupéré était de suivre un opérateur grâce à la caméra Pixy. On pourrait imaginer que le robot puisse tracter des charges et les transporter en suivant une cible, permettant à un ouvrier de porter beaucoup de matériel d'un point à un autre par exemple. Le robot doit également être capable de retourner à sa position d'origine de façon autonome lui permettant de se ranger tout seul après une utilisation par exemple.

Le robot n'ayant que deux roues, il doit être capable de se maintenir en équilibre dynamique lorsqu'il est à l'arrêt et doit pouvoir corriger son inclinaison afin de ne pas basculer. Il doit rester droit lorsqu'on lui applique une force d'un côté ou de l'autre. De plus, afin d'assurer une certaine stabilité, nous voulons que le robot ne fasse pas d'à-coup lorsqu'il est en fonctionnement. En effet, nous pouvons imaginer que le robot transporte quelque chose de léger et fragile, s'il y a trop de secousses et de changements de vitesses brutales, la cargaison pourrait tomber. Pour réaliser cela, un asservissement précis sur les codeurs incrémentaux ainsi que sur un accéléromètre est nécessaire.

On sait que la pixy est utilisée pour suivre un opérateur mais nous ne savions pas comment elle fonctionnait réellement. Il s'avère qu'elle a été programmée pour reconnaître une couleur, le rouge par exemple, et de se rapprocher de cette couleur. On imagine donc un opérateur muni d'un gilet de couleur rouge, reconnu par la pixy, permettant à cette dernière de commander les roues afin que le robot puisse le suivre. Nous avons remarqué par ailleurs que la pixy avait beaucoup de mal à gérer la luminosité ambiante. Les lumières de l'éclairage de la halle J1 étant assez puissantes, couplées à la luminosité extérieure, il arrive que la pixy se déconnecte de temps en temps. Ce problème contraint l'observateur à marcher lentement lui faisant perdre du temps.

## Développement du Projet

Notre premier objectif avec le robot était d'être capable de le faire fonctionner correctement, selon le code des étudiants précédents. Pour réussir dans cette tâche, on a étudié le code et essayé de le mettre en opération.

Nous avons repris le code de l'ancienne équipe chargée du projet mais il nous a paru incomplet. Nous avons investi du temps à la compréhension des différentes fonctions présentes dans le code et comment étaient pilotés les deux moteurs. Nous avons réalisé des tests avec le code de l'ancien groupe, si ces tests étaient concluants, nous essayerons de nous les approprier, autrement nous repartirons de zéro.

Nous avons choisi d'étudier chaque fonction du robot indépendamment afin de bien comprendre comment elles fonctionnent et de pouvoir, par la suite, les relier et les faire fonctionner simultanément. Nous nous sommes intéressés dans un premier temps au fonctionnement des moteurs à courant continu pilotant les roues de vélos, ainsi qu'aux codeurs incrémentaux servant à asservir ces moteurs.

### Mise en Marche du Robot

L'objectif, c'était de découvrir les commandes pour faire tourner les moteurs. Malheureusement, le code des étudiants de l'année précédent ne marchait pas, et on a dû faire de la recherche pour trouver les bonnes configurations du pont-H double Sabertooth utilisé, bien comme la configuration correcte de leurs DIP switches ainsi que sur les librairies Arduino nécessaires pour faire la bonne communication entre l'Arduino et le pont-H.

On a aussi adapté le code programmé en cours pour le traitement des codeurs incrémentaux, de manière à pouvoir retrouver les informations données.

Après ces changements, le groupe a réussi à faire fonctionner les moteurs, contrôler leur vitesse et sens de rotation. A partir de ce moment-là, les fonctions pour le déplacement du robot étaient déjà prêtes.

Il fallait maintenant programmer les fonctions de suivi de la cible et la partie de vision computationnelle, responsable d'envoyer les consignes aux moteurs pour réaliser les mouvements attendus.

Pour ce faire, il fallait être capable de commander la caméra.

## Orientation de la Caméra

La caméra du robot Pixy est fixée sur deux servomoteurs sur la partie avant du robot. Chaque servomoteur était fixé à 90° de l'autre, de telle façon que l'un était responsable de faire un mouvement horizontal de la caméra et l'autre au mouvement vertical.

On a pu récupérer le code des étudiants de l'année dernière, et avec peu d'amélioration et de changements, on a pu programmer les fonctions qui permettent à la caméra de diriger la caméra vers la direction de la cible grâce à l'utilisation des servomoteurs. Vu cela, l'objectif était de pouvoir diriger la caméra vers le centre de la cible qu'elle détectait.

## Importation des Informations de la Caméra

On a rapidement compris le code permettant de piloter la pixy, nous avons réussi à lui faire reconnaître la couleur rouge et à la faire pivoter pour suivre la cible grâce aux servomoteurs de la pixy. Cependant nous rencontrons plusieurs problèmes probablement dus à la luminosité inconstante de l'environnement de travail.

## L'Équilibre dynamique et l'accéléromètre

Nous avons voulu utiliser l'accéléromètre présent sur le robot afin d'asservir les moteurs pilotant les roues. L'exploitation du gyroscope nous permettrait d'éviter les oscillations créées lors du démarrage du robot ou encore les à-coups créés par des informations erronées envoyées par la pixy. Le code fonctionnait en changeant la vitesse de base des moteurs, de manière à forcer une accélération artificielle contraire à celle mesurée par l'accéléromètre.

Une fois le code complet, nous avons mis en relation les différents composants du robot afin de réaliser les premiers tests. Nous avons relié la pixy aux moteurs pilotant les roues de sorte que si la pixy détecte une couleur venant de la droite, la roue gauche accélère afin de centrer le robot sur la couleur détectée. Les résultats des premiers tests n'étaient pas parfaits mais pas décourageants non plus. La pixy

faisaient des allers-retours de droite à gauche de façon rapide lorsqu'elle perdait la vision de la couleur ce qui entraînait des problèmes de commandes importants. Il semblait également que la pixy avait de gros problèmes pour détecter la couleur. Le robot ne savait pas corriger les oscillations car le gyroscope n'avait pas été bien calibré dans le code général.

Après plusieurs modifications et d'autres tests, le robot suiveur remplissait sa fonction. Il arrivait à suivre une couleur et à corriger les oscillations qu'elles soient créées au sein du robot ou extérieure comme lorsqu'on appliquait une force sur l'avant ou l'arrière du robot. Le robot était déjà fonctionnel, mais ne réalisait aucune fonction spécifique au-delà de suivre la cible.

## **Axes d'Amélioration et Ajout de Fonctionnalités**

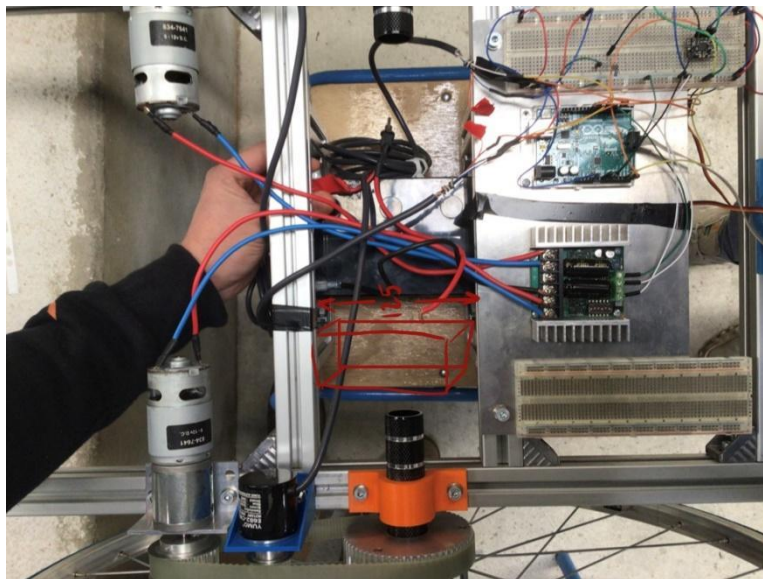
Nous avons identifié certains axes d'amélioration étant plus ou moins complexes et nous avons également réfléchi à l'ajout de nouvelles fonctionnalités. Nous avons pour objectif d'installer un emplacement proposé pour la batterie rechargeable qui était jusqu'à la, scotchée au robot. Nous aimerions également régler le problème de luminosité qui intervient sur la pixy, bien que ce ne soit pas choses faciles car les étudiants du groupe d'avant se sont déjà posé la question sans trouver de réelles solutions pour pallier ce problème.

Enfin nous avons réfléchi à l'ajout de nouvelles fonctionnalités faisant évoluer le robot lui permettant de réaliser certaines tâches plus complexes. Nous avons comme idée, dans un premier temps, de faire une sorte de « robot-serveur » avec plusieurs postes de livraison différenciés par couleurs et une zone de détection de pièces sur le robot. De telle sorte, le robot serait capable de reconnaître la couleur de la pièce, et de la livrer au poste correspondant à la couleur de l'objet. Une fois le travail effectué, le robot reviendra à sa position d'origine afin de venir chercher une autre pièce. Finalement, notre but a évolué vers un robot capable d'identifier une cible et de s'y rendre en évitant un ou plusieurs obstacles qui interviendraient sur son chemin et de faire le chemin inverse sans recopier le chemin aller mais en calculant l'itinéraire à emprunter. De telle sorte, si un obstacle est sur le chemin retour du robot mais n'était pas présent au premier passage du robot, il va pouvoir l'éviter plutôt que de rentrer dedans.

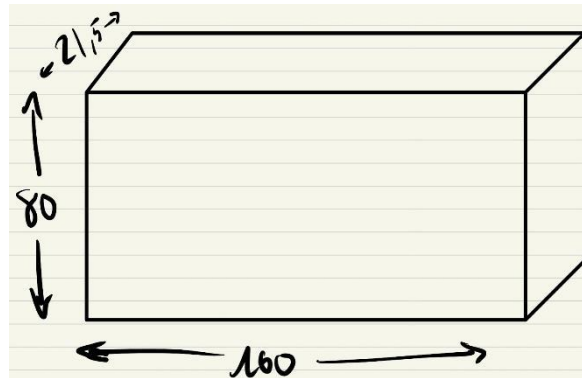


## Réalisation d'un porte batterie en impression 3D

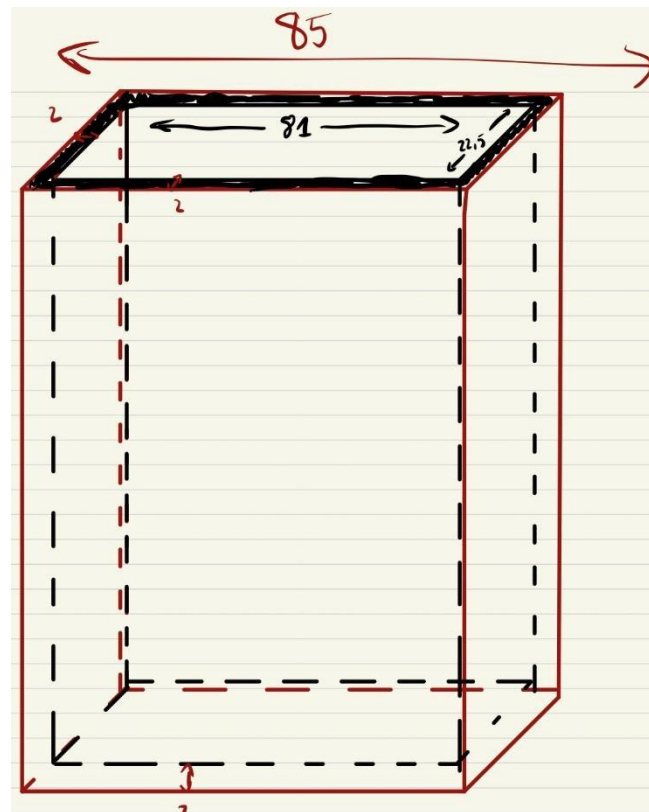
Comme je l'ai expliqué plus tôt dans ce rapport, nous avons vite remarqué la nécessité de créer un emplacement prévu pour placer la batterie rechargeable au sein du robot. Les ports USB de sorties doivent être facilement accessibles et la batterie doit tenir en place malgré les oscillations réalisées par le robot. Dans un premier temps, nous nous sommes interrogés sur l'emplacement de cette dernière sur le robot. Nous avons imaginé que plus cette dernière était proche du centre de gravité du robot, moins elle serait soumise à l'inertie créée lors des oscillations et de la mise en mouvement, et moins elle aurait d'impact sur l'équilibre dynamique. Nous avons donc choisi de l'installer proche de la batterie principale, comme il est visible sur l'image ci-dessous.

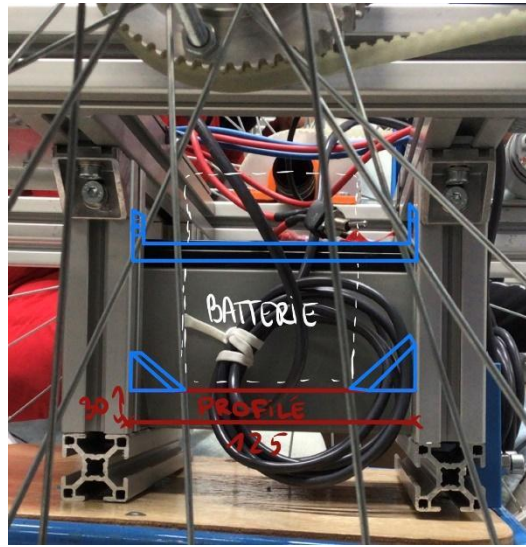
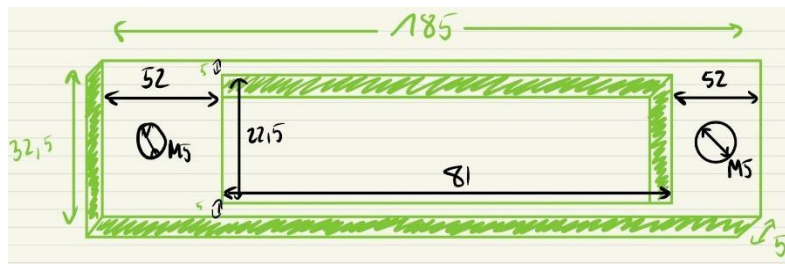


Nous avons mesuré la batterie grâce à un pied à coulisse mais afin d'être plus précis, nous avons recherché sur internet les spécificités de la batterie grâce aux références inscrites dessus (Ansmann 1700-0068) afin de connaître précisément les dimensions de cette dernière. Ces dimensions sont représentées par le schéma ci-dessous. Ces spécificités nous ont permis de dimensionner le support de la batterie afin qu'elle soit bien fixée et d'éviter le jeu.

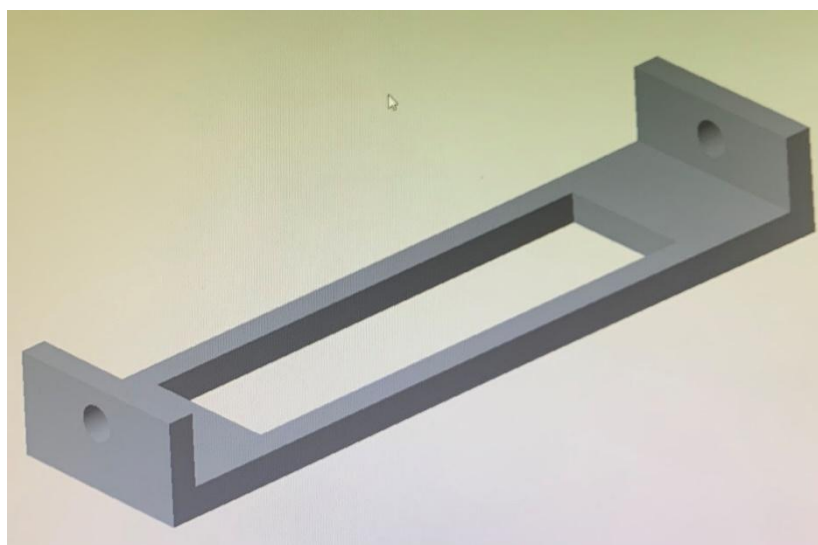


N'ayant pas d'imprimante 3D à notre disposition, nous avons fait appel au MindTech pour créer notre pièce. Nous avons imaginé le MindTech comme un réel fournisseur demandant plus d'argent et de temps pour une pièce plus complexe et plus grosses. Nous avons ensuite proposé plusieurs solutions permettant de porter la batterie et nous avons retenu la plus simple et la plus économique permettant de nous faire gagner du temps et de l'argent. Nous avons choisi la solution 2 en y ajoutant du profilé aluminium à notre disposition pour soutenir la batterie rechargeable plutôt que de créer la pièce numéro 1 demandant beaucoup d'impression.





Une fois notre pièce imaginée ainsi que l'installation de cette dernière aient été imaginée, nous avons dessiné la pièce sur un logiciel de CAO (ici CREO) afin d'en tirer un fichier .stl destiné aux imprimantes 3D.



Nous avons envoyé un mail au MindTech en précisant les paramètres d'impressions voulu tel que l'épaisseur du filament, le taux de remplissage, ou encore le matériel à utiliser. Nous avons pris de la marge au niveau du trou destiné à guider la batterie donc nous n'avons pas demandé de paramètre trop précis comme une épaisseur de filament faible car nous n'en n'avions pas besoin. De plus, la pièce n'étant pas soumise à des contraintes mécaniques trop importantes, le taux de remplissage n'avait pas besoin non plus d'être trop important. Durant l'impression de la pièce, nous avons découpé un bout de profilé aluminium à la dimension voulu. Pour ce faire, nous avons utilisé une des machines de découpe industrielle présente dans la halle J1. Après une présentation réalisée par M. Noailles, nous avons pu utiliser la machine seule afin de découper notre morceau de profilé de façon précise. Puis nous l'avons installé sur le robot grâce à deux équerres fournies par le professeur. Une semaine plus tard, la pièce avait été imprimée et nous l'avons installée sur le robot. Nous avons réalisé le test final en y installant la batterie et ce test fut plutôt concluant. C'est ainsi que nous avons déjà amélioré le robot en y ajoutant notre touche personnelle. Cette tâche n'a pas demandé de compétences en électronique particulières mais certaines compétences de mécaniciens ont été utiles.



## Code Couleurs et Luminosité

Notre premier objectif a été de faire détecter à la pixy plusieurs couleurs différentes, comme un code couleur. Dans un premier temps, nous voulions passer d'une couleur de détection à deux. On s'est vite heurtés à un problème sérieux : la pixy n'arrivait pas à différencier les couleurs en fonction de la luminosité changeante. Lors de l'initialisation, la pixy arrivait à faire la différence entre les deux couleurs distinctes mais lorsqu'on se déplaçait dans des zones plus sombres ou plus claires, la détection ne fonctionnait plus.

Pour pallier ce problème, nous avons eu pour idée d'installer sur le robot une lampe faisant office de phare seulement notre unique source d'alimentation était la batterie rechargeable reliée par un port USB. Après quelques recherches sur internet, nous avons vite compris que pour que la lampe ait un effet et éclaire malgré l'éclairage de la salle et de l'extérieur, il nous faudrait une lampe bien trop puissante qui ne se branche pas en USB. En effet, les lampes que nous avons trouvées étaient aux alentours des 420 lumens alors qu'un néon comme ceux utilisés dans la halle industrielle éclairent autour des 3350 lumens. Nous avons donc pensé à utiliser une membrane optique en prenant exemple sur les appareils photos numériques qui règlent la luminosité en fonction de l'angle d'ouverture mais nous n'avions pas pu trouver sur internet de membrane optique automatique destiné à la vente. Nous avons donc réfléchi à d'autres solutions comme un pare-soleil, un asservissement au posemètre ou encore un asservissement selon la pigmentation. Nous avons aussi réfléchi à revoir l'ajout de notre fonctionnalité et à réfléchir à une nouvelle fonctionnalité à ajouter.

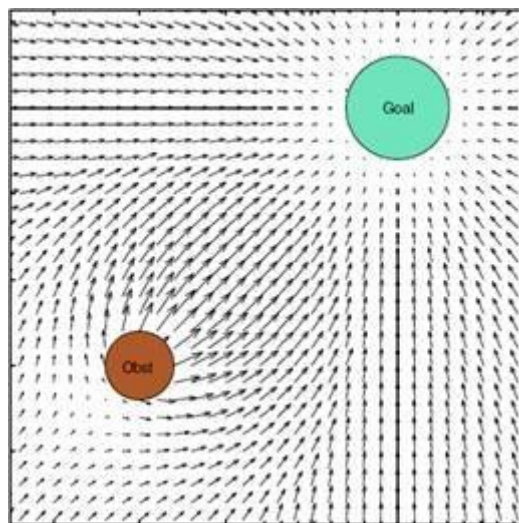
## Contour des Obstacles

Le groupe de l'année dernière avait proposé un robot capable de suivre des obstacles mobiles, comme un ouvrier portant d'un gilet de couleur.

L'objectif pour ce sujet était d'ajouter la fonctionnalité de contourner les obstacles. Pour cela, on a choisi de changer la cible d'un point mobile à un point fixe (bien que l'algorithme développé soit également valide pour les cibles mobiles, il était plus simple pour les tests de supposer une cible fixe pour valider le suivi d'une cible mobile) et mettre dans le chemin de la cible, des obstacles, dont la position devrait être évité par le robot.

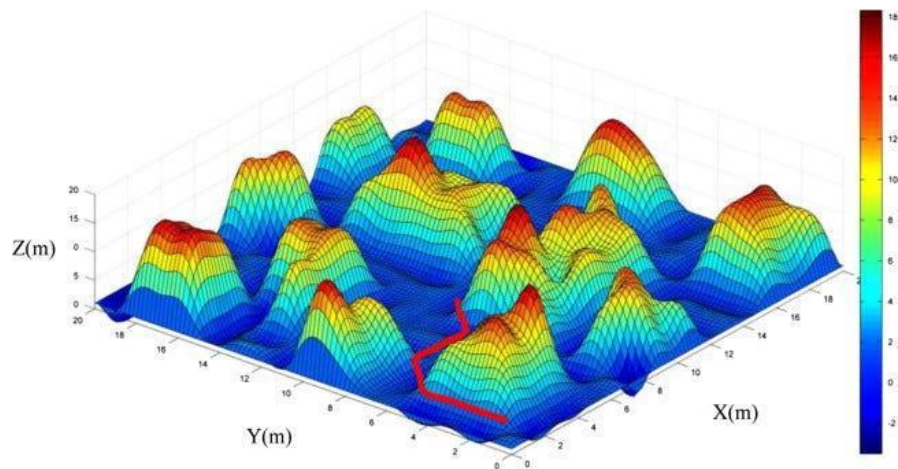
Ce type de défi exige des modifications dans le code original. Il était important d'ajouter la capacité de la Pixy d'identifier deux objets et les classer comme cibles ou obstacles et utiliser leur position pour le calcul de la trajectoire. Pour se faire, l'algorithme de champ potentiel a été implémenté.

L'algorithme de champ potentiel est un algorithme de planification de trajectoire qui a pour but de faire se comporter le robot comme une particule, soumise à un champ potentiel (comme un électron dans un champ potentiel électrique, par exemple). L'algorithme attribue à chaque point de l'espace une valeur de « potentiel » et le robot a pour objectif trouver le chemin que lui permet de minimiser ce potentiel. Les cibles sont donc mathématiquement définies comme des points de bas potentiel, et les obstacles comme des points de haut potentiel. Si les fonctions qui décrivent les potentiels générés par les cibles et obstacles sont des fonctions bien continues et rapidement assimilables, c'est possible de trouver le gradient du champ potentiel à partir du point occupé par le robot.



A partir de ce moment-là, le robot doit suivre le négatif de ce gradient, de manière à se conduire au minimum local de « potentiel ». Une façon de visualiser cet algorithme est d'imaginer un relief accidenté, dans lequel une balle est déposée. Au lieu des obstacles, il y a des collines, et au lieu des cibles, il y a des vallées. La balle sera naturellement attirée vers les vallées.





Dans le cas de notre robot, le déplacement est toujours dans une espace plane, de manière qu'on peut définir la position du robot en deux dimensions.

Dans notre cas, il n'y a qu'une cible et un obstacle. Donc, le formalisme mathématique est le suivant :

Soit la position de la cible  $C(x_{cible}, y_{cible})$  et de l'obstacle  $O(x_{obs}, y_{obs})$

Soit la fonction potentiel de la cible  $V_{cible} = +a\sqrt{(x - x_{cible})^2 + (y - y_{cible})^2}$ , soit la partie supérieure du cône de centre dans la cible,  $a$  une constante de choix de projet.

Soit la fonction potentiel de l'obstacle  $V_{obs} = b \exp(h((x - x_{obs})^2 + (y - y_{obs})^2))$ , soit une gaussienne de paramètres  $b$  et  $h$  choix de projet.

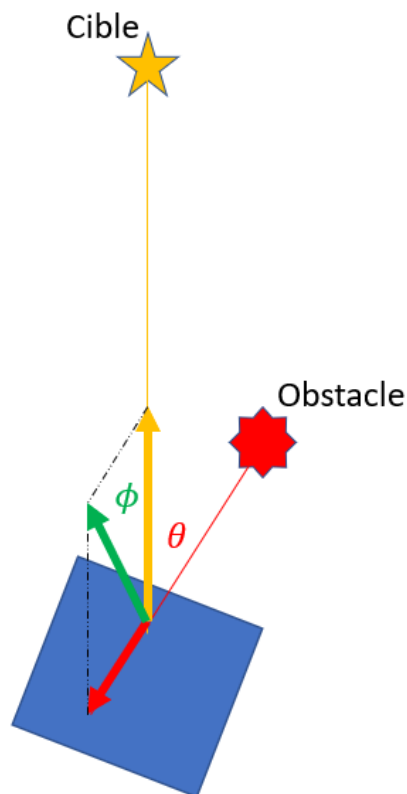
Le potentiel total est  $V_{tot} = V_{cible} + V_{obs}$

Ce qu'on peut voire est que le gradient de la cible a toujours une intensité constante, ainsi que le potentiel de l'obstacle n'a d'effet que quand le robot est proche de l'obstacle. La distance est calculée en sachant que l'obstacle a une taille définie (en pixels) quant à une distance standard (en mètres) de la caméra, et par perspective, paraît plus petit quand il est plus loin et plus grand quand il est plus proche. De cette façon, le robot est toujours attiré à la cible, mais n'est repoussé par l'obstacle que quand la distance est suffisamment petite. Les valeurs des constantes  $a, b, h$  sont responsables pour la calibration des « forces » qui agissent sur le robot.

Dans l'aspect de la programmation, on n'a pas besoin de ce formalisme. Comme par choix de projet, la caméra est toujours dirigée vers la cible, ce qu'il faut faire est savoir où se trouve l'obstacle dans son champ de vision. L'angle entre la direction actuelle du robot et la cible est connu. De cette façon, par la caméra, on peut savoir quel est l'angle entre l'obstacle et la cible.

Avec ses informations, calculant l'intensité des gradients de chaque champ potentiel séparément et faire la trigonométrie nécessaire pour déterminer la « force résultante ». La force résultante n'est pas intéressante par son intensité, mais pour sa direction. On met à consigne du robot (qui normalement serait de s'approcher de la cible dans sa direction) à la direction de l'angle de la « force résultante »

Le diagramme suivant permet de visualiser le comportement du robot :





De cette façon, il est toujours possible d'éviter que le robot se frappe contre les obstacles en chemin, en faisant un tour pour les éviter. La calibration des constantes permet d'augmenter ou diminuer le tour réalisé par le robot.

## Suivie de ligne

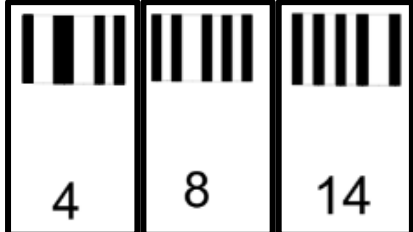
Pour notre deuxième système, la Pixy est utilisée pour l'analyse d'une ligne, cette caractéristique est déterminée avec précision grâce à son système de vision intégré. Contrairement au premier mode où l'on utilisait les servos moteurs pour faire bouger la Pixy, pour ce système elle est fixée et orientée vers le sol, lui permettant de capturer les informations nécessaires pour suivre la ligne avec succès. Notre objectif principal est de permettre au robot de suivre un chemin en fonction à la fois de la ligne tracée, et du code-barre détecté que nous déterminerons. Ce processus utilise un algorithme spécifique appelé Line Tracking, qui analyse en temps réel les données visuelles provenant de la caméra pour prendre les décisions de direction appropriées. Ces données sont des vecteurs générés par la lecture. Grâce à cette caractéristique, on est capable d'effectuer des mouvements précis et contrôlés, c'est-à-dire des mouvements asservis de départ et d'arrêt, et surtout lui permettant de naviguer avec succès sur différents parcours basés sur des lignes et des codes-barres.

Dans un premier temps, on a défini toutes les variables utiles au projet et chaque paramètre d'entrées pour configurer la Pixy pour ce mode (la table ci-dessous montre les paramètres utilisés), et comme dans le premier mode, ce code est basé sur un système asservi. De plus, la partie de l'accéléromètre et de luminosité sont les mêmes pour les deux modes ainsi que la structure de base des actionneurs. Tandis que pour les 3 barcodes utilisés, on a trouvé au travers de quelques tests que ceux pour le 4, le 8 et le 14 étaient les plus précis dans nos conditions. Ainsi, nous avons défini que si la Pixy lit la valeur 4, elle va l'assimiler comme "TURN LEFT", 14 correspond à "TURN RIGHT" et enfin si c'est 8 cela veut dire "STOP",

<b>Edge threshold</b>	<b>40</b>
<b>Maximum line width</b>	<b>180</b>
<b>Minimum line width</b>	<b>10</b>
<b>Edge distance</b>	<b>3</b>
<b>Line extraction distance</b>	<b>10</b>

Maximum line compare	6000
Maximum merge distance	3
Minimum line length	10
Line filtering	10
Intersection filtering	10
Barcode filtering	10
Default turn angle	0

---



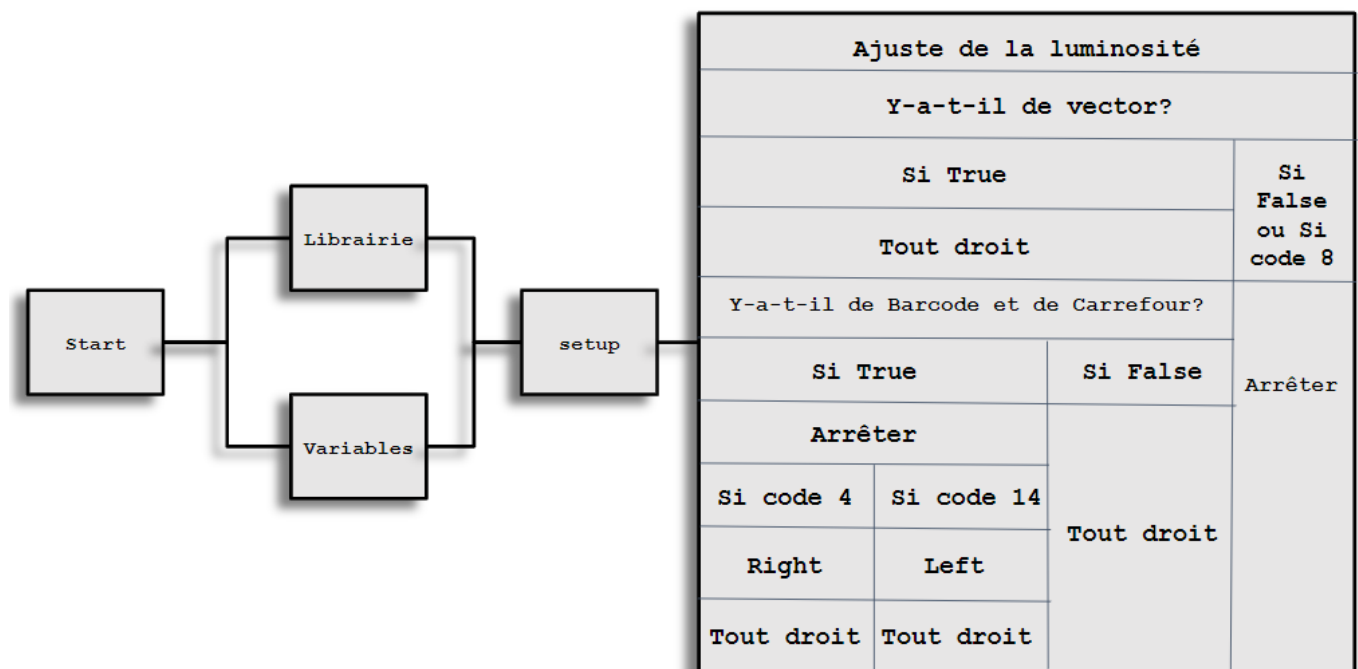
Alors, son principe de fonctionnement est surtout sur l'analyse de direction d'un vecteur. La Pixy utilise la lecture d'un vecteur représentant la différence entre l'extrémité du vecteur et le centre de la fenêtre. Cette différence est enregistrée dans une variable qui détermine la vitesse des roues du robot, permettant ainsi d'effectuer de petites courbes. Il est important de noter que cette variable est ajustée pour obtenir la vitesse souhaitée pour chaque roue.

$$error = (int32t)_{pixy.line.vectors \rightarrow m-x1} - (int32t)_{x-CENTER}$$

Lorsqu'il s'agit de tourner à une intersection, un système spécifique est mis en place. Si la Pixy détecte à la fois une intersection et un code-barres, le moteur du robot se déplace vers la ligne correspondante à la signification du code-barre enregistrée. Cependant, si seule une intersection est détectée, le système continuera à avancer tout droit.

En général, le système d'asservissement fonctionne en fonction du nombre de tours pour atteindre la consigne souhaitée. Il est conçu pour les phases d'arrêt et de démarrage, permettant ainsi un contrôle précis du mouvement du robot.

Ainsi, grâce à ces mécanismes intégrés, la Pixy est capable de suivre les lignes, de tourner aux intersections et de s'adapter en fonction des informations fournies par ses capteurs, assurant ainsi une navigation fluide et précise. Le diagramme ci-dessous est idéal pour mieux comprendre la structure du code.



## Conclusion

En conclusion, ce projet de développement d'un système mécatronique basé sur un robot a permis aux étudiants de se familiariser avec l'utilisation des microcontrôleurs, d'appliquer leurs connaissances en automatisme et en asservissement, et d'acquérir des compétences en programmation. Le projet a été réalisé en équipe de trois étudiants, chacun se concentrant sur une partie spécifique du robot, tout en étant en mesure de collaborer et de s'entraider en cas de besoin.

Le robot lui-même était composé de deux roues de vélo reliées par une armature en profilé aluminium. Il comporte divers composants électroniques, tels qu'une carte Arduino, des plaquettes de connexion électronique et une caméra Pixy. Le robot avait initialement la fonctionnalité de suivre un opérateur à l'aide de la caméra Pixy, ce qui permettait d'imaginer des applications telles que le transport de charges en suivant une cible.

Le développement du projet s'est déroulé en plusieurs étapes. Tout d'abord, l'équipe a étudié le code développé par les étudiants précédents et a réussi à le faire fonctionner correctement. Ils ont ensuite travaillé sur l'orientation de la caméra en utilisant des servomoteurs, permettant ainsi de diriger la caméra vers la cible détectée.

L'importation des informations de la caméra a également été abordée, bien que des problèmes liés à la luminosité ambiante aient été rencontrés. L'équipe a ensuite exploité l'accéléromètre pour asservir les moteurs et assurer un équilibre dynamique au robot. Des tests ont été effectués pour vérifier le bon fonctionnement du robot dans différentes situations.

Enfin, l'équipe a identifié des axes d'amélioration et réfléchi à l'ajout de nouvelles fonctionnalités. Parmi les idées proposées figuraient l'installation d'un emplacement pour une batterie rechargeable, la résolution du problème de luminosité de la caméra Pixy, l'ajout d'une fonctionnalité permettant au robot de reconnaître et de livrer des pièces à des postes de livraison spécifiques, l'ajout d'une fonctionnalité de planification de trajectoire en fonction de l'environnement ainsi que celui d'une fonctionnalité de line tracking. Bien que toutes ces fonctionnalités n'aient pas été implémentées, le travail fourni reste enrichissant.

Ce projet a permis aux étudiants de mettre en pratique leurs connaissances théoriques en mécanique, en électronique et en programmation. Ils ont également acquis des compétences en travail d'équipe, en résolution de problèmes et en gestion de projet. Le développement de ce système mécanique a illustré l'importance croissante de l'automatisation et de l'intégration des technologies dans le domaine de l'ingénierie, et a montré comment les ingénieurs doivent s'adapter aux évolutions technologiques pour répondre aux besoins de l'industrie.

En somme, ce projet a été une expérience enrichissante pour les étudiants, leur permettant d'appliquer leurs connaissances dans un contexte réel et de développer des compétences essentielles pour leur future carrière d'ingénieur mécanique.