PROTIS

Projet : robot holonome piloté à distance

Gonzague Bonin, Thadek Ferrand

Lola Guengard-Morineau, Julie Massé

Nous attestons que ce travail est original, que nous citons en référence toutes les sources utilisées et qu'il ne comporte pas de plagiat.

Table des matières

Vidéo de démonstration du projet : https://youtu.be/ipa-E9LYDjU

Introduction et contexte

La société SOLEC nous a contacté afin de mettre en œuvre un robot piloté à distance, pour travailler dans leurs ateliers de production. À partir du plan fourni par la société, notre objectif a été de répondre à leur appel d'offre en proposant une solution technique. Elle permet à un utilisateur non expérimenté c'est-à-dire qui n'a pas reçu de formation préalable de contrôler à distance le déplacement d'un robot. Pour cela elle a choisi notre équipe d'experts en programmation, systèmes embarqués et électronique.

Dans un premier temps, nous allons rappeler le cahier des charges requis. Puis nous allons décrire la démarche utilisée pour aboutir au prototype final. Ensuite, nous allons établir un bilan de l'équipe et des compétences acquises. Enfin, nous expliquerons les fonctionnalités pouvant encore être améliorées.

Descriptif du cahier des charges

Le robot utilisé est un robot holonome fonctionnant avec une batterie (Figure 1). Le robot doit pouvoir se déplacer en ligne droite, dans les couloirs de l'usine et entrer et sortir des différentes pièces, en fonction des besoins. Pour ce faire, il doit également pouvoir effectuer des rotations, d'angles de 90°, 180° ou -90°.

En cas de panne de batterie, le robot doit pouvoir se déplacer dans une base de rechargement prévue dans l'atelier.

L'atelier a une cartographie connue à l'avance et un utilisateur (employé de SOLEC sans compétences techniques particulières) doit pouvoir lui transmettre les ordres par l'intermédiaire d'un ordinateur à distance.

Le robot est basé sur une plateforme holonome. Le robot a donc trois roues à galets tangentiels, qui permettent des rotations “sur place” du robot ou d’avancer en contrôlant la rotation de deux des trois roues.

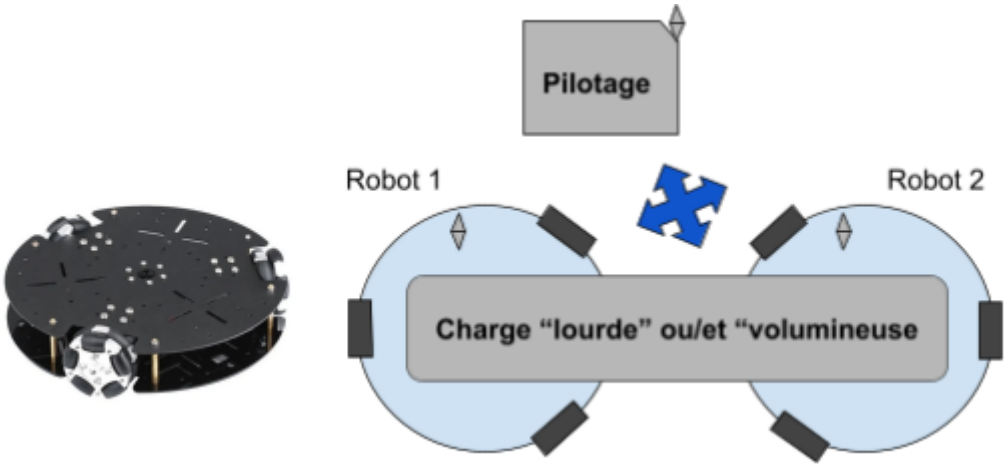

Figure 1 :Robot holonome

Certaines performances sont par ailleurs attendues. Le robot doit être **rapide** : il doit pouvoir avancer à une vitesse comprise entre 10 et 30 cm/s. Il doit être **fiable** : une erreur maximale de 2 cm est tolérée sur la position et une erreur de 3° est tolérée sur l’angle. Il doit, de plus, être **autonome** : il doit pouvoir réaliser un parcours de 1 km sans que ses batteries ne soient rechargées. Enfin, il doit être **ergonome** : l’interface Humain-Machine permettant de transmettre les ordres de parcours doit pouvoir être utilisée sans formation préalable. Les données du trajet doivent pouvoir être collectées et affichées en fonction du temps.

Démarche utilisée pour aboutir au prototype final

Organisation du projet

Notre projet a été découpé en deux phases distinctes : d’un côté l’interface graphique et la détermination du trajet que doit emprunter le robot par l’ordinateur, et de l’autre le pilotage du robot. Notre équipe s’est scindée en deux binômes afin de mettre en place ces phases avant de les relier.

Interface graphique et trajet emprunté par le robot (Thadek et Lola)

Nous devons déterminer le trajet du robot dans l'usine du plan suivant (figure 2):

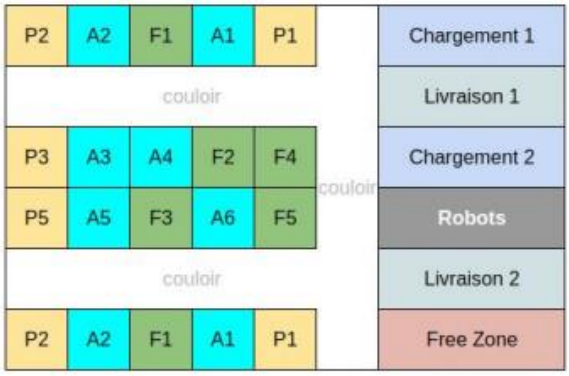

Figure 2 : Plan de l'usine SOLEC

La première étape est de construire l'interface graphique de l'usine. Pour cela, nous utilisons la bibliothèque *tkinter* de Python. Nous commençons par représenter le plan de l'usine par une matrice de 6 lignes et 7 colonnes. Nous définissons 42 fonctions (6x7) qui codent chacune une case de l'usine. (figure 3) Les fonctions couloirs retournent "zone interdite" dans la console et les autres fonctions codent une case. La bibliothèque tkinter permet de coder des boutons. Nous reconstruisons ainsi le plan de l'usine. Ensuite, l'utilisateur doit cliquer sur un premier bouton qui correspond à la case de départ et sur un second bouton qui correspond à la case d'arrivée.

Pour un cerveau humain, trouver le trajet entre deux points en passant seulement par les couloirs est un problème très simple. Pour un ordinateur, c'est plus compliqué car il faut écrire un programme avec de nombreuses conditions. Nous avons choisi Python comme langage de programmation pour élaborer cet algorithme. L'usine comporte cinq blocs qui peuvent être codés de la même façon. Un bloc de colonne et quatre blocs de ligne (figure 3). Il y a ensuite deux cases spéciales correspondant à des carrefours (figure 3). Ce sont les cases 2;6 et 4;6.

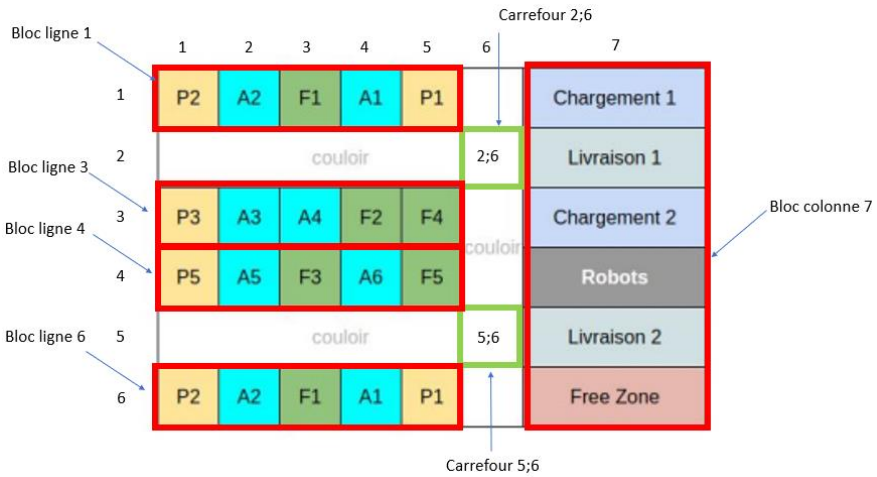

Figure 3 : Plan de l'usine SOLEC sous forme de matrice

Le programme permettant le calcul du chemin récupère d'abord une liste contenant les coordonnées du point de départ et du point d'arrivée. Si le point de départ et le point d'arrivée sont sur les mêmes blocs ou sur des blocs placés en face à face (comme ligne 1 et ligne 3 ou ligne 4 et ligne 6), c'est assez facile. Il suffit de dire au robot de sortir de la case, de tourner dans le bon sens selon le signe de la différence entre les premières coordonnées des cases de départ et d'arrivée, de retourner dans le bon sens, puis d'avancer jusque dans la case. Il faut faire attention car le robot rentre dans une case en marche avant. Il en sort donc en marche arrière. Si le point de départ et le point d'arrivée ne sont pas dans les mêmes blocs (ou en face à face) alors il faut amener le robot jusqu'à la case de carrefour la plus proche. Puis reprendre la procédure avec les blocs qui n'ont pas été testés.

Ensuite, il faut convertir la liste avec les déplacements obtenus pour qu'elle soit facile à lire pour le programme en C. Nous créons une nouvelle liste de la taille du double+1 de la liste précédente. Le premier élément de cette nouvelle liste correspond au double du nombre de déplacements. Ensuite, nous ajoutons chaque déplacement de l'ancienne liste un à un dans la nouvelle liste sous la forme d'une lettre, puis d'un nombre. La lettre correspond au déplacement ("a" pour avancer, "r" pour reculer, "t" pour tourner dans le sens trigonométrique et "h" pour tourner dans le sens horaire) puis le nombre de pas. Pour les instructions pour tourner, le nombre de rotations à 90° est indiqué. Par exemple pour aller de la case1;2 à la case 1;7 la liste retournée sera la suivante:

[14,"r", 1,"h", 1, "a", 4,"t",1, "a",1, "h",1, "a", 1]

Enfin, nous établissons la liaison Bluetooth entre la carte Bluetooth de l'ordinateur et celle du robot. La carte de l'ordinateur doit être configurée en mode maître et celle du robot en mode esclave. Les instructions sont alors transmises une par une au robot.

Pilotage du robot (Gonzague et Julie)

Le robot holonome est piloté par une carte Nucléo, qui communique avec l'ordinateur au moyen d'un module Bluetooth RN42. La Nucléo embarque un programme codé en langage C, qui permet l'interprétation et l'exécution du trajet fourni par l'ordinateur et transmis *via* la liaison Bluetooth.

Après avoir préalablement établi la connexion entre le module Bluetooth du robot et celui de l'ordinateur, la première étape est de recevoir les caractères envoyés par le programme Python par la liaison Bluetooth un par un, et de reconstruire un tableau de caractères dans le programme en langage C. Ce tableau stocke alors la suite d'instructions nécessaire à l'accomplissement du parcours souhaité par le robot. La seconde étape est la lecture linéaire du tableau afin d'exécuter les instructions l'une après l'autre.

Le robot a été calibré afin de connaître la correspondance entre la consigne que l'on doit envoyer au robot pour le faire avancer d'une distance donnée. Le robot holonome étant muni de moteurs à courants continus, on utilise la donnée issue de l'incrémenteur afin de connaître le nombre de tours d'axe de moteur effectivement réalisés. Ainsi, la calibration du robot revient à connaître le nombre de tours de moteur à effectuer pour que le robot avance d'un mètre, et tourne de 90°.

Schéma fonctionnel

Il est possible de résumer les procédés mis en jeu sur le schéma fonctionnel suivant (figure 4):

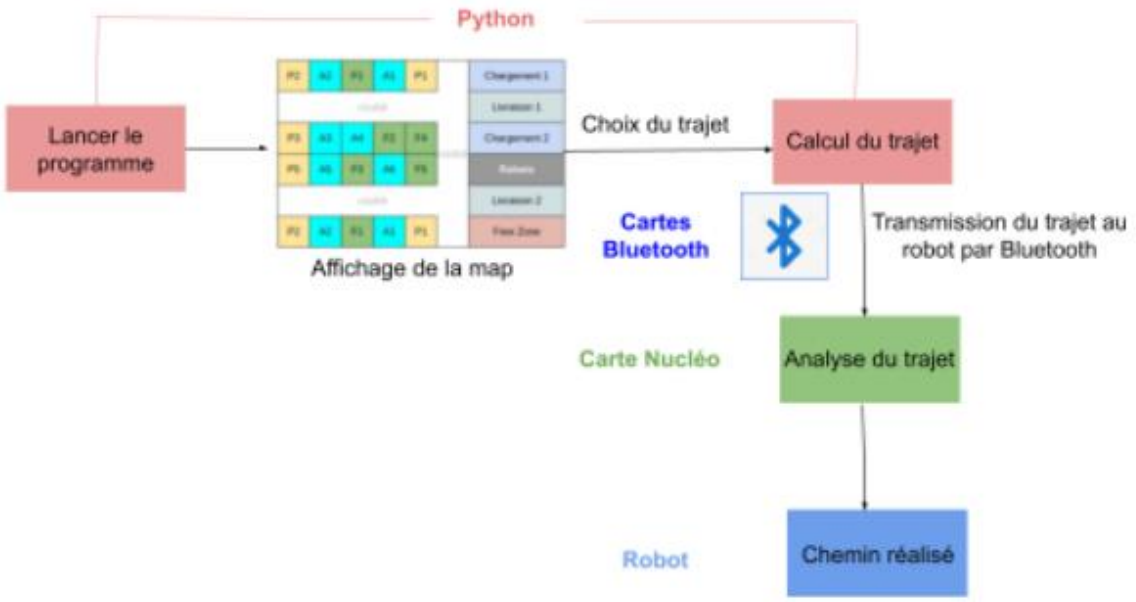

Figure 4 : Schéma fonctionnel résumant les points forts du projet

Bilan de l'équipe d'un point de vue technique et sur les compétences acquises

Ce projet nous a permis d'approfondir nos connaissances tant d'un point de vue de la maîtrise de Python et de la réalisation d'une interface graphique que de celle du langage C pour le programme embarqué sur le robot. Nous avons également acquis des bases sur la mise en place et la connexion de module Bluetooth avec notamment la syntaxe des commandes à envoyer au module pour le commander, le mettre en mode esclave/maître, le connecter à un autre module ou encore pour permettre une authentification par code PIN. Nous avons également dû présenter notre projet aux membres commerciaux de SOLEC ce qui nous a permis d'acquérir des compétences en communication. En effet, nous avons expliqué notre projet de façon claire et accessible à des non-spécialistes ce qui a développé notre esprit de synthèse. La répartition des tâches et le travail d'équipe a été un point clé dans ce projet et notre organisation nous a permis de réaliser le projet selon le planning établi lors du début du projet. Enfin le rapport technique a développé notre communication écrite.

La détermination du trajet devant être emprunté par le robot pour se rendre d'un point à un autre dans l'atelier a été le premier problème théorique rencontré. Il a fallu développer un algorithme fiable et rapide sous Python pour renvoyer une liste contenant la suite des instructions que le robot doit réaliser.

Nous avons ensuite développé une interface graphique avec le module *tkinter* de Python afin de reproduire le plan de l'usine. Cette étape nous a permis d'acquérir une compétence nouvelle, le développement d'interface graphique n'ayant pas été étudié dans les années précédentes.

Du côté de la programmation embarquée du robot, une étape nouvelle a été la communication entre le module bluetooth du robot et celui relié à l'ordinateur. Cette liaison permet l'envoi de la liste d'instructions générée par Python au robot. L'interprétation des informations reçues par le robot caractère par caractère et la construction d'un tableau de caractères en langage C les stockant a été une difficulté du projet. Nous avons cependant pu la résoudre avec l'aide des enseignants et mettre en œuvre une démonstration de l'intégralité du projet lors des deux dernières séances (séance avec les CFA uniquement et séance avec la filière classique uniquement).

Fonctionnalités pouvant être améliorées

Les fonctionnalités du projet pouvant être améliorées sont les suivantes.

Fiabilité: Le cahier des charges spécifie qu'une erreur de 3° sur l'angle et de 2 cm sont autorisées sur le déplacement. Le robot peut être calibré pour ce type de déplacement. Cependant, nous rencontrons deux problèmes. Le premier est que l'un des moteurs semble légèrement défectueux et ne tourne pas tout à fait à la même vitesse que les deux autres. Cela induit une petite erreur, même si la vitesse des moteurs peut s'effectuer séparément. Le deuxième problème est que tous ces calibrages dépendent de la puissance et de l'état de la batterie.

Autonomie: Le robot devrait être capable de retourner à la base après un kilomètre. Pour cela nous avons pensé ajouter un compteur qui compte chaque déplacement élémentaire (avancer d'une case ou tourner de 90°). Par soucis de simplicité les déplacements tourner et avancer seront comptés de la même façon même si avancer fait faire une plus grande distance aux roues que tourner. Une fois que le compteur arrive à une certaine limite, il faudra envoyer le robot vers la zone de chargement.

Suivi du robot: actuellement notre robot peut effectuer un déplacement entre deux points. Le point de départ et le point d'arrivée doivent être donnés par l'utilisateur. Nous partons du principe que l'utilisateur appuie en premier sur la case où se trouve effectivement le robot. Une amélioration serait que le robot sache sur quelle case il se trouve et que l'utilisateur n'ait qu'à appuyer sur la case d'arrivée.

Conclusion

Pour conclure nous avons répondu à l'appel d'offre de la société SOLEC. Notre robot piloté à distance fonctionne parfaitement pour les fonctionnalités de base, c'est-à-dire se déplacer dans l'usine entre deux points. Des fonctionnalités supplémentaires sont en cours de développement afin d'améliorer notre prototype et nous espérons vous fournir des robots pilotés à distance pouvant travailler parfaitement dans cette usine très prochainement.

Codes en annexes

Vous trouverez sur eCampus deux programmes :

- **ordinateur.py** contenant l'interface graphique, l'algorithme de détermination du parcours et le protocole de communication avec le module bluetooth.
- **robot.c** contenant tout le code embarqué sur le robot pour recevoir les données par le module bluetooth, les exploiter et suivre les instructions prévues.