Rapport du projet d'ingénierie robotique

Maxence Ahlouche Maxime Arthaud Korantin Auguste
Martin Carton Thomas Forgione Thomas Wagner

9 décembre 2013

Table des matières

1 Présentation de l'équipe

Cette équipe a été menée par Maxime Arthaud, assisté de son Responsable Qualité Thomas Forgione. Les autres membres de l'équipe sont Martin Carton, Maxence Ahlouche, Korantin Auguste et Thomas Wagner.

2 Méthodes de programmation du robot

2.1 Programmation graphique

Au début du projet, nous avons commencé à programmer le robot comme indiqué dans le sujet, c'est à dire en utilisant le logiciel de programmation graphique disponible sur Windows. Seulement, ce logiciel, bien qu'accessible à tout le monde, nous a semblé très peu pratique à utiliser, si bien que nous n'avons pas été très productif lors des premières séances.

En effet, dès que nous souhaitions faire autre chose que des instructions de base (par exemple, faire tourner le moteur à une vitesse dépendant de la valeur d'un capteur), nous passions beaucoup trop de temps à essayer de comprendre comment fonctionnait l'interface.

De plus, nous avons rencontré un problème que nous n'avons pas réussi à résoudre : notre robot, avançait de manière saccadée. Bien que nous ayons quelques hypothèses sur l'origine de ce problème (bug ou documentation non à jour), nous n'avons pas réussi à le résoudre avec le logiciel fourni.

À cause de ces désagréments, nous avons décidé d'essayer d'autres méthodes pour programmer le robot.

2.2 Librairie Python

Un des membres de notre équipe a trouvé sur Internet une librairie en Python, qui permet de contrôler un robot branché en USB. Cette librairie (qui s'appelle `nxt-python`) nous a permis de régler notre problème d'avancement saccadé.

Elle nous a également permis de programmer notre robot de manière très simple, et de gagner beaucoup de temps par rapport au logiciel de programmation graphique.

Elle présente toutefois un inconvénient majeur : elle ne permet pas de télécharger le programme sur le robot (le python étant interprété sur l'ordinateur) ; par conséquent, le robot devait rester branché à l'ordinateur lors de l'exécution du programme, et nous devions le suivre avec l'ordinateur. Ce qui n'est évidemment pas pratique.

2.3 Langage NXC

Finalement, nous avons décidé de programmer le robot en utilisant le langage NXC (*Not Exactly C*), développé spécifiquement pour les robots NXT.

Le compilateur que nous avons trouvé nous permet également de télécharger le programme sur le robot, donc nous avons résolu le problème posé par la librairie en Python, ainsi que ceux posés par le logiciel de programmation graphique.

3 Suivie de murs

Nous avons réalisé un programme qui permet au robot de longer un mur, grâce à NXC.

Nous avons fait le choix, par manque de capteurs, de ne longer que les murs à gauche du robot.

3.1 Suivie d'un mur

Afin de longer un mur, nous avons posé un capteur à ultrasons sur le côté gauche de notre robot. Lorsque nous détectons que nous sommes trop éloignés du mur, nous tournons légèrement à gauche.

Cette méthode donnant des résultats peu probants si la direction initiale du robot n'est pas exactement parallèle au mur, nous avons décidé d'ajouter un autre capteur à ultrasons à l'avant du robot. Ainsi, notre robot ne fonce plus dans les murs, et tourne à droite quand le capteur de devant détecte quelque chose.

Ainsi, nous pouvons suivre un mur droit de manière assez régulière.

3.2 Lissage du mouvement du robot

Afin d'éviter d'avoir un mouvement erratique lorsqu'on n'est pas exactement parallèle à la paroi que l'on doit longer (mouvements de "rebondissement" sur le mur, par exemple), nous avons décidé d'améliorer notre algorithme via une astuce très simple : le changement de direction dépend de la distance du mur. Ainsi, si nous nous éloignons légèrement du mur (à cause d'une trajectoire non parallèle au mur, par exemple), alors nous tournerons très légèrement à gauche, sans ralentir. Ainsi, nous nous retrouverons presque parallèle au mur.

Dans le cas d'un virage à gauche du mur, le capteur détecte un mur à l'infini ; dans ce cas là, nous tournons autant que possible, jusqu'à retrouver un mur. Dès que le capteur détecte un mur, on repasse dans le cas précédent ; par conséquent, notre robot ne se retrouve quasiment jamais face au mur qu'il devait longer. De plus, ceci présente l'avantage que notre robot est désormais capable de faire demi-tour autour d'un mur qui s'arrête d'un coup, sans virages (comme dans le cas des planches fournies par notre professeur pour tester le robot).

Dans le cas d'un virage à droite du mur, notre algorithme ne change pas.

Ainsi, nous avons un robot capable de longer les murs de manière très fiable, peu importe l'angle des murs.