Rapport sur le Projet de Robotique

Création d'un projet avec le robot e-puck 2

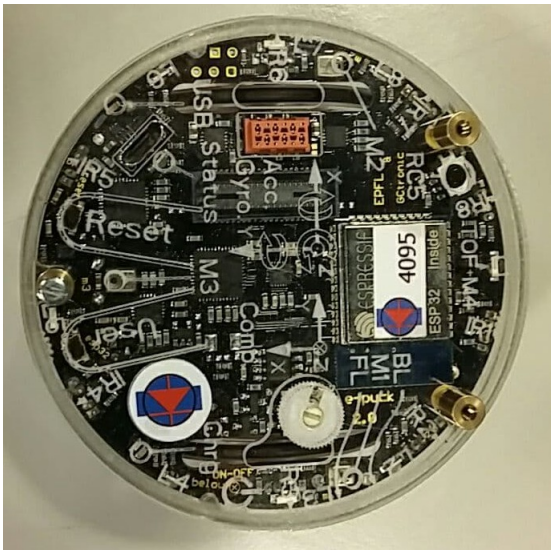

FIGURE 1 – E-puck vu depuis le dessus, l'avant du robot se trouve en haut de la photo.

Jonathan Henry 287725
Louis Sinss 273022

Lausanne
École Polytechnique Fédérale de Lausanne

EPFL

Table des matières

1 Introduction

Dans ce projet, nous avons décidé de faire un robot skieur de Super G. Pour cela il devra descendre un plan incliné et slalomer entre des piquets symbolisés par des blocs de bois. Chaque piquet aura un symbole différent suivant si le robot doit passer à droite ou à gauche de ce piquet et ainsi slalomer jusqu'en bas de la piste.

2 Comportement du robot

Le robot part depuis la première porte. Avec les capteurs IR3 et IR6 (voir Figure 1) qui se trouvent au dessus des roues, il détecte les deux piquets de la porte. Il se met alors à regarder avec la caméra pour détecter le piquet suivant et s'en rapproche jusqu'à ce qu'il soit à environ 5 cm de distance. A cet endroit, il va déterminer si c'est le piquet gauche (qu'on appellera piquet 1 pour la suite du rapport) de la porte ou celui de droite (qu'on appellera piquet 2) à l'aide des symboles dessinés dessus (Figure 2).

FIGURE 2 – La porte tel que le e-puck la voit, donc le piquet 1 avec la ligne noire entourée de deux bandes blanches se trouve à gauche, et le piquet 2 avec une bande blanche entourée de deux bandes noires se trouve à droite.

Pour la suite, s'il a détecté un piquet 1, il tournera à droite, avancera un peu puis tournera à gauche pour se retrouver face à la porte suivante. Bien sûr, s'il détecte un piquet 2, il tourne d'abord à gauche puis à droite pour se retrouver face à la prochaine porte. Il utilisera ensuite ses capteurs de proximité pour être bien au milieu de la porte. Et bien sur, quand il franchi la porte, les capteurs IR3 et IR6 détecte une nouvelle fois que la porte est franchie et le cycle recommence.

Une fois que le e-puck a franchi la dernière porte, il arrive en bout de piste et va détecter un changement de gravité selon l'axe Z, et va donc s'arrêter et effectuer une petite danse de la victoire.

3 Fonctionnement du programme

L'utilisateur doit commencer par placer le robot entre les deux piquets de départ, puis allumer le robot. Une fois le robot allumé, le robot se déplacera tout seul jusqu'à la fin du parcours.

3.1 Démarrage

Le robot va commencer le parcours à l'aide de ses capteurs de proximité, c'est pourquoi il est important de bien le positionner au début, entre les deux piquets de départ. Le détail du passage et re-centrage entre les piquets est décrit plus loin.

3.2 Déplacement vers le piquet

Le robot va ensuite se déplacer jusqu'au premier piquet en face de lui en utilisant la caméra et un algorithme de détection de lignes adapté du code du TP4. Grâce à régulateur proportionnel, le robot va réguler sa position et régler sa distance avec le piquet. Tout au long de cette phase, le robot aura les leds 3,5 ou 7 allumée, indiquant le piquet qu'il détecte. Il allumera la led 3, située à droite du robot, lorsqu'il détectera le piquet 2, la led 7 (située à gauche) pour le piquet 1 et enfin la led 5 (située à l'arrière du robot) lorsqu'il ne détecte aucun des deux.

3.3 Pivotage et centrage

Une fois que la distance, fixée préalablement pour avoir une certaine marge de manoeuvre, est atteinte, le robot va tourner d'un quart de tour à droite ou à gauche suivant le piquet qu'il a détecté. Il va ensuite avancer en ligne droite d'environ 5 cm pour se mettre entre les deux piquets et tourner à nouveau d'un quart de tour dans le sens opposé, et enfin avance encore en ligne droite de 7 centimètres, afin d'être bien centré entre les deux piquets.

3.4 Passage entre les piquets

Cette phase se passe à l'aide des capteurs de proximité, qui vont réguler le déplacement et la position du robot jusqu'à ce qu'il ait passé les deux piquets, et reprendre le contrôle par la caméra. Le robot va faire des petits ajustements de sa vitesse pour se retrouver le plus en face possible du piquet suivant.

3.5 Arrivée

Ce cycle va se répéter tout seul jusqu'au bas de la piste. Puis une fois en bas, le robot utilise son accéléromètre pour détecter un changement d'accélération sur l'axe vertical (axe z) et va arrêter la caméra et les capteurs de proximité. Il va effectuer une petite danse de victoire pour célébrer sa réussite.

4 Utilisation des ressources

4.1 Capteur de proximité

Les capteurs de proximité sont utilisés pour se déplacer à travers la porte, entre les deux piquets. Nous prenons les valeurs des capteurs IR3 et IR6, situés respectivement à droite et à gauche du robot. Les valeurs transmises nous permettent de corriger notre trajectoire et de donner l'information au robot que les blocs ont été dépassés, et qu'il peut recommencer à utiliser la caméra.

4.2 Caméra

La caméra est utilisé pour capter la lumière rouge et reconnaître les symboles utilisés sur les piquets. Pour cela, nous sommes parti du TP4, CamReg [1] mais nous avons modifié la reconnaissance de la ligne. En effet, nous avons remarqué qu'en absence de zone blanche suffisamment grande, le e-puck a tendance à voir du noir autour de deux pics séparés par une "vallée" (voir Figure 3). Par conséquent nous avons modifié le code pour qu'il puisse détecter les deux symboles dans même étant entourés de bruits.

Le programme va parcourir l'image de gauche à droite et relever s'il y a des variations importantes et soudaines des valeurs des pixels, des montées ou des descentes. Pour le piquet 1 (voir Figures 2, 3), il faut que le programme détecte une montée suivi d'une descente et d'une autre montée. Pour éviter les bruits parasites qui peuvent faire des pics soudains et qui fausseraient le programme, il faut que lors d'un pic, la valeur du pixel situé à une certaine distance horizontale soit également à une valeur haute. Pour le piquet 2 (voir Figures 2, 4), il faut que le programme détecte une montée puis une descente qui aient une certaine largeur afin que ça soit considéré comme une ligne blanche entourée de deux lignes noires.

1. Donnée du TP4, consultée le 06.05.2021

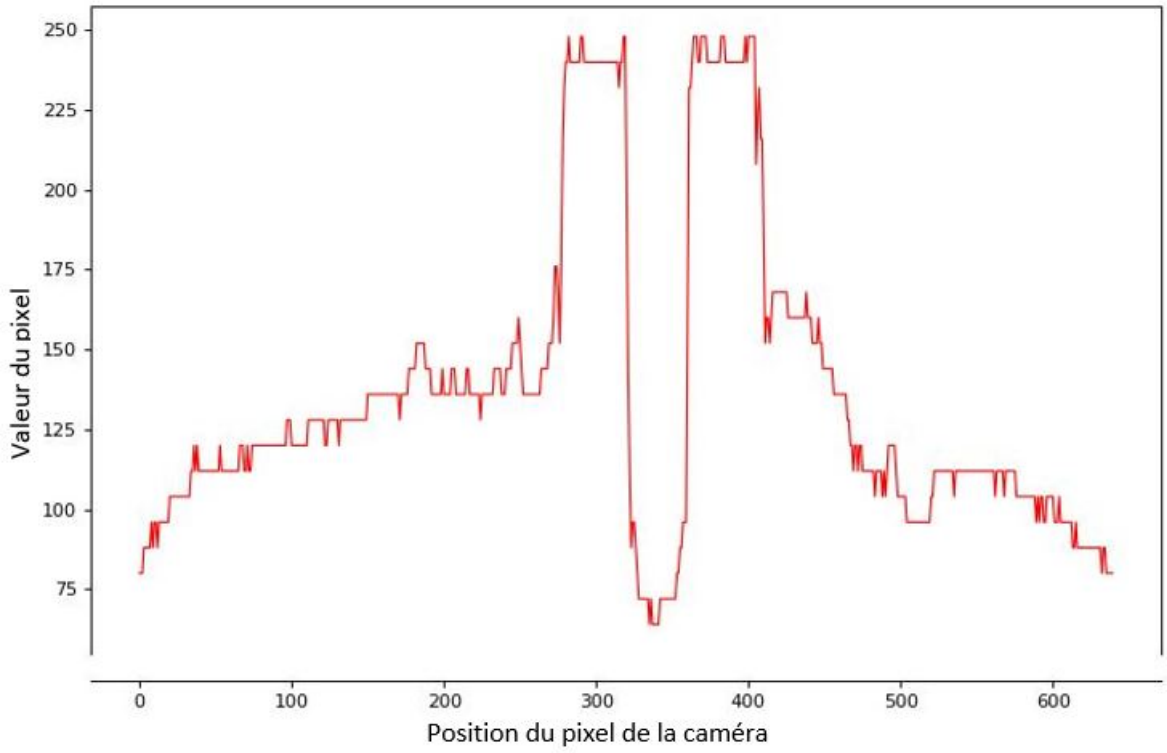

FIGURE 3 – Ce graphe représente la valeur moyenne de trois lignes contiguës pour une même position selon l'axe x des pixels. On observe bien le bruit autour du piquet 2, les deux "vallées" pour la valeur du pixel correspondent aux deux bandes noires et le pic au milieu correspond à la bande blanche au milieu des deux bandes noires.

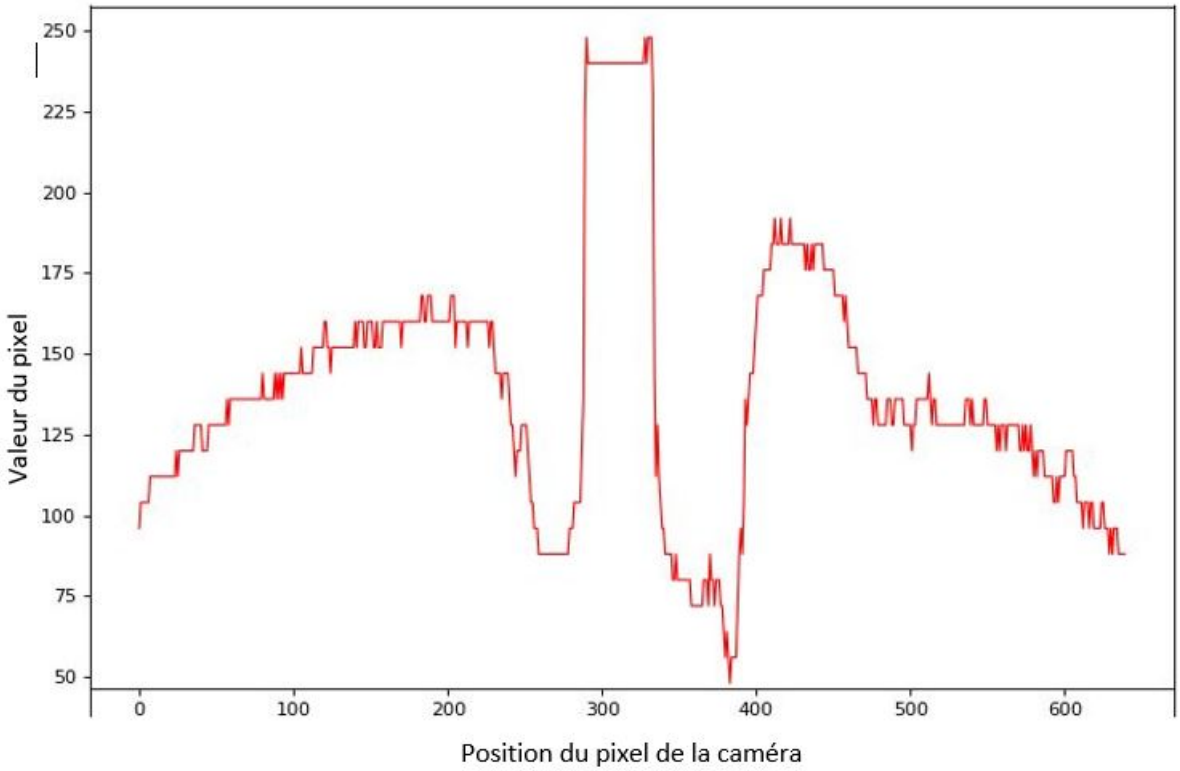

FIGURE 4 – Ce graphe représente la valeur moyenne de trois lignes contiguës pour une même position selon l'axe x des pixels. On observe bien le bruit autour du piquet 1, les deux pics pour la valeur du pixel correspondent aux deux bandes blanches et la "vallée" correspond à la bande noire au milieu des deux bandes blanches.

4.3 IMU

L'IMU est utilisé pour détecter un changement de pente du plan de déplacement du robot. Seule la valeur de l'accélération sur l'axe Z est utilisée. De plus, nous avons choisi la fonction qui permet de récupérer la moyenne sur un échantillon de valeurs pour avoir une meilleure précision. L'IMU ne va donc servir qu'a arrêter le robot à la fin du parcours, nous ne l'avons pas utilisé autrement.

4.4 Moteurs

Les moteurs sont énormément employés dans ce projet. En effet, nous les utilisons en permanence pour le déplacement. Nous avons plusieurs types de commandes de contrôle de la vitesse. Nous avons le premier mode qui est avec un régulateur proportionnel, qui va imposer une vitesse en fonction de la distance avec le piquet en face du robot. Un deuxième est une valeur fixe de vitesse, que nous utilisons pour tourner (en donnant des grandeurs de signe opposé mais de même valeur absolue aux deux moteurs) ou pour se déplacer en ligne droite. Enfin, la troisième méthode est une combinaison des deux autres : La valeur de déplacement en ligne droite est fixée, et il y aura une correction de rotation avec les valeur reçues des capteurs de proximité. Afin d'éviter des problèmes, après presque chaque opération avec les moteurs, nous remettons la vitesse des moteurs à zéro. Le code suivant va imposer sa vitesse puis remettre à zéro, etc. Ceci nous évite des fautes de robot qui continue à avancer alors qu'il devrait être à l'arrêt.

4.5 Gestion des Threads

Dans notre projet, nous utilisons plusieurs threads que nous avons créé ou modifiés pour nos besoins. Les plus importants sont ceux de la caméra (CaptureImage et ProcessImage) et MoveControl, le module de gestion de mouvement. Un thread mineur que nous avons aussi créé est ImuEnding, qui est dédié à l'arrêt du robot une fois les conditions réunies. Les threads de la caméra vont prendre les images et les traiter, puis rendre disponibles les données nécessaires au déplacement, c'est à dire la position de la ligne, la largeur de la ligne ainsi que l'identification du piquet. Le thread de déplacement va ensuite prendre ces informations et agir en conséquence. En interne, le thread possède deux booléen déclarés en static qui vont permettre de choisir le mode de déplacement du robot : caméra ou capteur de proximité. Une fois la tâche d'un des mode accomplie, il va changer l'état de ces deux variables afin de laisser la main à l'autre mode de déplacement. Nous avons choisi de laisser ces deux modes dans le même thread, étant donnée qu'ils sont intrinsèquement liés entre eux, il nous semblait plus facile de les laisser interagir de cette façon. Le dernier thread ne va rentrer en jeu que lorsque le robot va arriver en bas de la piste, il aura donc juste le rôle d'arrêter le déplacement et de faire la petite danse de la victoire.

5 Mode d'emploi

5.1 Environnement

Pour la mise en place, il faut tout d'abord trouver une surface plane horizontale pour placer la piste inclinée. Pour un fonctionnement optimal du programme, il faut également faire attention aux sources de lumières et où elles sont situées. Il faut une luminosité suffisamment grande pour que le e-puck puisse voir les piquets et aussi les symboles dessinés dessus.

Par rapport à la position de la source de lumière, le mieux est de la placer du côté du départ de la piste afin qu'elle éclaire bien les piquets et crée un maximum de contraste entre les bandes noires et les bandes blanches.

Si la lumière est trop frontale et qu'elle rentre dans le champ de l'appareil photo, le e-puck est ébloui (tel un vrai skieur) et ne peut plus voir où se situe la prochaine porte.

Il est tout à fait possible d'utiliser une lumière artificielle pour éclairer le dispositif, même si la lumière du soleil reste la meilleure car il y a moins d'ombres qu'avec les lumières créées par plusieurs lampe dans une salle.

5.2 Mise en route

1. Placer le robot entre les deux premiers piquets.
2. Allumer le robot en appuyant sur son interrupteur situé juste à coté de la roue droite, derrière le capteur IR3 (voir Figure 1).
3. Observer le déroulement de la course.
4. Une fois la course finie et la danse de la victoire effectuée, éteindre le robot puis au choix lui faire refaire une descente, ou alors le ranger dans sa boite jusqu'à sa prochaine utilisation.

6 Conclusion

Le point le plus compliqué selon nous dans notre projet est la détection des blocs et le déplacement jusqu'à ceux-ci. La complexité vient du fait que la caméra voit un environnement variable suivant la position du robot. Nous avons donc mis au point une manière de différencier les deux piquets, en essayant aussi d'éliminer le bruit alentour. Notre code se base sur la détection des lignes blanches sur le bloc, soit une seule sur le bloc de droite, soit deux sur celui de gauche. Pour essayer de réduire le bruit ambiant, nous avons peint en noir notre "piste de ski" car nous avons considéré que l'environnement était plus proche du noir que du blanc. Ceci ne s'est pas révélé tout à fait véridique, et nous a posé quelques soucis. En rétrospective, nous aurions peut-être eu meilleur temps de peindre notre piste en blanc, par souci d'analogie pour commencer, et ensuite peut-être que le bruit aurait été mieux "noyé" dans un environnement blanc que noir.

Un autre point d'amélioration du projet pour que ça ressemble plus à une course de ski, serait de rajouter les quelques signaux sonores signalant le début de la course, dont le dernier a une tonalité plus élevée. Dans cette optique de ressemblance, nous pourrions également rajouter un chronomètre dans le e-puck qui renverrait le temps du robot à l'ordinateur une fois la ligne d'arrivée franchie. Il l'enverrait par Bluetooth à l'ordinateur et on pourrait ainsi organiser des compétitions de e-pucks avec d'autres groupes.