TP Semaine 10 - Modélisation et Robotique

November 30, 2016

1 Introduction

Cette semaine, nous allons utiliser le module `pythymio2` vu en cours. Pour rappel, voici un petit programme utilisant `pythymio2`, qui vous montre comment utiliser ce module (inutile de le taper sur votre ordi).

```
from pythymio2 import *

def monprogramme():
    # le robot avance pendant 1 seconde,
    # tourne un peu sur lui-meme dans un sens
    # puis dans l'autre sens, et s'arrete
    av()
    yield "sleep 1"
    td()
    yield "sleep 0.5"
    tg()
    yield "sleep 0.5"
    arrete()
    # on verifie les boutons du Thymio
    b = yield "buttons"
    if b[1]==1:
        print('le bouton gauche est appuye')
    # on verifie les capteurs de distance du Thymio
    p = yield "prox"
    if p[2] > 1000:
        print('obstacle en vue')

runThymioControl(monprogramme)
```

Comme indiqué en cours, votre fonction `monprogramme` ne peut pas prendre de paramètre.

2 Evitement d'obstacle

Programmez le Thymio pour qu'il aille tout droit. Lorsqu'il rencontre un obstacle, il doit tourner sur lui même jusqu'a ne détecter plus aucun obstacle avec ses 5 capteurs frontaux. Alors il se remet à avancer dans sa nouvelle direction.

3 Imprécision de la mesure du temps

La commande `yield ''sleep 1''` n'est pas précise: le robot attendra *environ* une seconde. Pour savoir exactement combien de temps le robot a attendu, utilisez la fonction `time()` du module `time`, qui renvoie le nombre de secondes écoulées depuis le 1ier Janvier 1970. Codez le programme suivant sur votre ordinateur, et lancez-le.

```
from pythymio2 import *
from time import *

def tempsQuiPasse():
    tempsDebut = time()
    yield "sleep 1"
    tempsFin = time()
    print('Le sleep a duré exactement ',tempsFin-tempsDebut,' secondes')

runThymioControl(tempsQuiPasse)
```

Maintenant, modifiez ce programme pour qu'il affiche la durée **moyenne** d'une instruction `yield ''sleep 1''`. Pour calculer cette durée moyenne, vous devrez exécuter cette instruction plusieurs fois, mesurer le temps à chaque fois et afficher la moyenne de ces mesures.

4 Distance d'un bout à l'autre

Programmez le Thymio pour qu'il aille droit devant lui jusqu'à détecter un premier obstacle. A ce moment, le robot devra faire demi-tour et repartir dans l'autre sens jusqu'à détecter un deuxième obstacle. Alors le robot devra s'arrêter, et afficher sur l'écran la distance entre les deux obstacles.

Remarque : il n'y pas de moyen précis pour mesurer des distances ou des angles avec un Thymio. Donc pour faire demi-tour, vous devrez faire tourner sur lui-même le Thymio pendant un certain temps (déterminez lequel). De même, vous n'avez aucun moyen direct pour calculer la distance entre les deux obstacles. En revanche, vous pouvez mesure le temps que mets le Thymio pour aller d'un obstacle à l'autre, et multiplier ce temps par sa vitesse. Cela vous donnera une estimation de la distance.

5 Joystick

Programmez le Thymio pour qu'en temps réel, on puisse le controler avec ses boutons. Plus précisément:

- si on appuie sur le bouton droite, le robot doit pivoter vers la droite
- si on appuie sur le bouton gauche, le robot doit pivoter vers la gauche
- si on appuie sur le bouton du centre et que le robot est immobile, le robot doit se mettre à avancer
- si on appuie sur le bouton du centre et que le robot est en mouvement, le robot doit s'arrêter

6 Evitement et Réel et en Simulé

Reprennez le code de l'évitement d'obstacle des questions précédentes. A l'aide de l'enrironnement `robosim`, vous devrez déplacer le robot à l'écran afin qu'il imite les déplacements du Thymio.

Pour placer le robot simulé au milieu d'un environnement vide, et pour faire en sorte que ce robot simulé dessine une trace derrière lui, vous taperez au début de votre code:

```
from robosim import *
init('vide')
pendown();av()
```