TP - Robotique - L1
Déplacement du Thymio et Simulation

November 12, 2015

Ce TP est structuré en deux parties. Dans la première partie, on mettra au point des fonctions pour faire se déplacer le Thymio, et lui faire repérer les obstacles qui l'entourent. Ensuite, il lui faudra cartographier son environnement.

1 Déplacement et Capteurs de distance du Thymio

1.1 Moteurs

Dans le cours, on a vu que le code suivant permettait de faire avancer la roue gauche de 3000 unités de déplacement.

```
import pythymio, time, random

with pythymio.thymio([],[]) as Thym:
    x= 0.0          # position a 0.0
    t= time.time()

    # Faire pendant 4 secondes
    while time.time() - t < 4.0:
        e = 3000.0 - x     # difference entre destination et position courante

        u = max(-500,min(500,e/3))
        Thym.set('motor.left.target',[ u ])

        v = Thym.get("motor.left.speed")  # vitesse du moteur
        x += v                            # mise a jour de la position

        time.sleep(0.1)                   # attendre 1/10 seconde


    Thym.set('motor.left.target',[0])
    time.sleep(0.5)
    Thym.stop()
```

1. Adaptez le code précédent pour qu'il fasse avancer simultanéement les deux roues de 3000 unités.

2. Creez une fonction `avance(x)` qui fasse avancer le Thymio de `x` centimètres. Lancez plusieurs fois cette fonction et estimez la marge d'erreur du déplacement (par exemple, `avance(20)` a une marge d'erreur de ± 2 cm).

3. Créez une fonction `tourne(a)` qui fasse tourner les deux roues en opposition (donc le Thymio devra tourner sur lui-même). Le paramètre `a` devra représenter l'angle en degrés. Si $a > 0$, le Thymio tournera vers la droite, sinon la gauche.

1.2 Capteurs

1. La fonction `Thym.get("prox.horizontal")` renvoie une liste de 5 valeurs indiquant la proximité des obstacles détectées par les 5 diodes infrarouges sur le devant du Thymio. Ces diodes ne peuvent rien détecter au delà de 8 cm.
   Si ces valeurs sont égales à 0, le Thymio ne détecte pas d'obstacles
   Lorsque l'obstacle se rapproche, la valeur augmente jusqu'à atteindre 4000 à peu près.

2. Ecrivez un programme qui affiche cette liste 10 fois par secondes. Ensuite, utilisez la fonction `max` pour afficher le maximum de cette liste 10 fois par seconde.

3. La fonction `Thym.get('prox.ground.delta')` permet de savoir si le "nez" du Thymio est au dessus d'une table ou du vide. Lancez cette fonction et regardez ce qu'elle affiche, selon que le Thymio est sur une table ou sur du vide.

1.3 Evitement d'obstacles

Ecrivez un programme qui fasse ceci:

1. En s'inspirant du code de `avance(a)`, écrivez une fonction `toutdroit()` qui fasse avancer le Thymio jusqu'à ce qu'il détecte un obstacle ou qu'il atteint le bord de la table. Cette fonction doit renvoyer le nombre de centimètres parcourus.

2. Ecrire une fonction `quartTourAlea()` qui fasse faire au robot un quart de tour sur lui-même, soit à droite, soit à gauche, choisi aléatoirement. Note: pour tirer un nombre aléatoire parmi 0 et 1, faire `random.randint(0,1)`

3. Le programme principal doit appeler `avance`, puis `quartTourAlea`

2 Simulateur de Robot

Le simulateur de robot peut être appelé avec les commandes suivantes:

```
from robosim import *
init('vide')
```

On avance le robot de x pixels en faisant `av(x)` et on le fait pivoter à droite par exemple, avec `tournedroite(a)` où `a` est un angle en degrés. Le robot peut aussi dessiner: la fonction `pendown()` lui fait dessiner une trace lors de ses déplacements.

1. Faites dessiner au robot un octogone (huit faces)
2. Implémentez (et testez) la fonction suivante, qui dessine un cercle rouge juste en face du robot.

```
from math import pi,cos,sin

def dessine_rond():
    x,y=position()
    a=orientation()*pi / 180.0
    circle(x+25*cos(a),y+25*sin(a),r=12)
```

3 Robot Thymio et Simulateur

Maintenant, on veut reprendre le programme d'évitement d'obstacles codé sur le Thymio précédemment, et l'implémenter sur le simulateur en parallèle.

1. Reprendre le code du Thymio, et rajouter pour chaque déplacement du Thymio une commande `av(x)` ou `tournedroite(a)` qui reproduise en simulation les mouvements du robot.
2. Dès que le robot détecte un obstacle, appeler la fonction `dessine_rond()`. Ainsi, la carte des obstacles se dessine peu à peu.
3. ****Question bonus niveau expert**** Etant donné que les relevés du Thymio ne sont pas très fiables, on décide de ne garder que les 10 derniers ronds dessinés à l'écran. Pour cela, à chaque fois, on efface les ronds précédents avec la commande `efface()`, et garde une liste contenant les coordonnées des 10 derniers cercles à tracer, et on les trace.