Robots LEGO EV3 : premiers pas

1 Présentation

Dans ce premier travail, vous allez réaliser des premiers programmes en Python pour commander un robot LEGO EV3. L'objectif est de se familiariser avec la programmation Python pour les robots LEGO.

1.1 Matériel

Le robot utilisé est un robot Lego Mindstorms EV3 (illustré sur la figure 1). Chaque robot possède les équipements suivants :

— un capteur de couleur ;
— un capteur ultra-son ;
— un gyroscope ;
— deux moteurs (roue droite et gauche) ;
— un dongle Wifi.

Le capteur couleur distingue 8 différentes couleurs et peut mesurer l'intensité lumineuse perçue, il est orienté vers le bas pour observer la couleur ou le niveau de gris du sol permettant au robot de se localiser par rapport au centre du chemin sur lequel il se déplace. Le capteur à ultrasons permet de mesurer la distance (principe du sonar) aux obstacles situés devant le robot. Le gyroscope (capteur de position angulaire) mesure le mouvement de rotation du robot et les changements d'orientation.

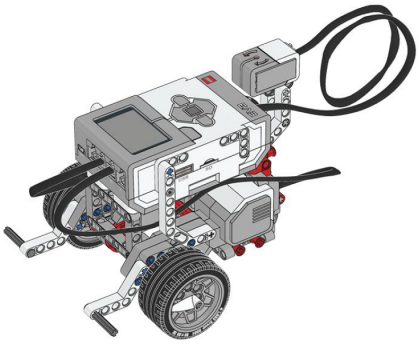

FIGURE 1: Le robot LEGO EV3

1.2 Logiciels : installation

Pour la programmation, nous utilisons l'environnement open-source ev3-dev2. Les sources et la documentation de cet environnement est disponible sous : `https://github.com/ev3dev/ev3dev-lang-python`. Le site web officiel est sous : `https://www.ev3dev.org/`.

La document de l'API python de ev3-dev est disponible sous ce lien : `https://python-ev3dev.readthedocs.io/en/ev3dev-stretch/index.html`. Un autre source de documentation pour utiliser cette bibliothèque est le site web suivant : `https://sites.google.com/site/ev3devpython/`

1.3 Connection au robot et transfert des programmes

Vous allez écrire des programmes Python sous l'éditeur Pyzo ou simplement avec un éditeur Notepad++ sur les machines de la salle et les exécuter ensuite sur les robots. **Dans votre éditeur de code, il faut activer l'option d'édition LF au lieu de CRLF. Cette option est disponible dans votre éditeur en bas à gauche de la barre d'état.**

Pour exécuter vos scriptes python dans la brique LEGO, il faut les transférer sur le robot, ensuite les exécuter depuis le robot via un terminal ou le FileBrowser disponible dans le menu de la brique LEGO.

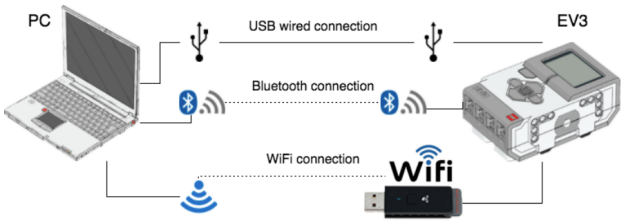

FIGURE 2: Connectivité du robot LEGO EV3.

Les robots LEGO sont accessibles via le USB, WiFi or Bluetooth.

Chaque robot possède une adresse IP sur le réseau WiFi ENSEMSAMI, affichée sur l'écran du robot après son démarrage. Néanmoins, nous pouvons aussi transférer les programmes via le cable USB du robot à connecter à la machine. La procédure de connexion par le cable USB est décrite ci-dessous. Les détails sont disponibles sous ce lien https://www.ev3dev.org/docs/tutorials/connecting-to-the-internet-via-usb/.

— Connecter le robot à votre machine avec le cable USB.

— Maintenant, dans l'écran principal du robot, choisissez l'option "Wireless and Networks", "All network connections" et "Wired". Choisissez ensuite l'option "Connect". Le robot affiche "Configuring" ensuite une adresse IP qui commence par 169 est affichée sur l'écran du robot.

Pour se connecter sur un robot et avoir accès à un terminal, nous utilisons l'outil *PuTTY* pour Windows. Sous Linux, l'outil ssh est disponible en ligne de commande. Il faut lancer l'outil *PuTTY* et saisir l'adresse IP du robot sur lequel vous voulez se connecter. L'adresse IP est affichée sur l'écran du robot. Vous pouvez aussi utiliser *ev3dev* comme nom de host dans PuTTY pour se connecter au robot.

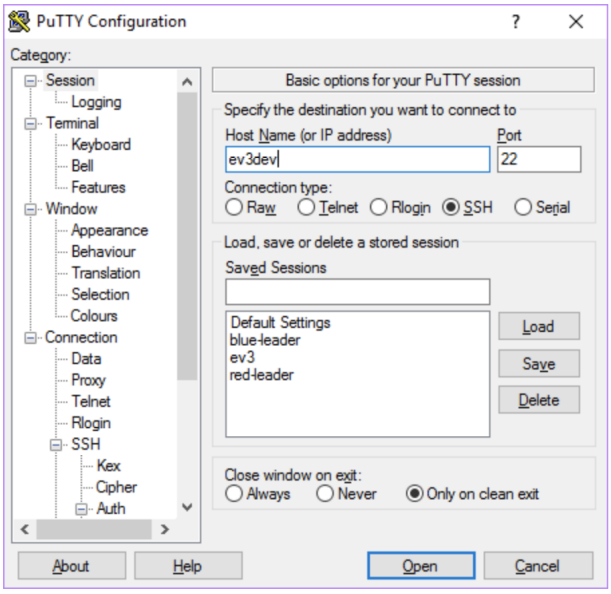

FIGURE 3: Connexion au robot LEGO via PuTTY.

Le mot de passe de connexion sur le robot est *maker* et le login est *robot*. Un tutoriel est disponible sous ce lien https://www.ev3dev.org/docs/tutorials/connecting-to-ev3dev-with-ssh/. Pour transférer les programmes depuis votre machine Windows vers le robot, vous utilisez le programme *WinSCP*. Si le programme n'est pas installé sur les machines de la salle, il faut le télécharger depuis ce site et l'installer : https://winscp.net/eng/index.php.

1.4 Premières commandes

Une fois que vous avez obtenu, un terminal sur le robot. Nous allons effectuer quelques tests. Dans le terminal du robot, taper la commande *python3*. Un terminal python s'ouvre. Dans ce terminal python taper les commandes suivantes pour tester le moteur connecté sur le port B du robot :

```
from ev3dev2.motor import OUTPUT_B, LargeMotor
m = LargeMotor(OUTPUT_B)
```

```
m.run_timed(speed_sp=300,time_sp=1000)
```

Ce petit programme démarre le moteur branché sur port B de votre robot pendant une seconde avec une vitesse de 300 degrés/seconde.

2 Mes premiers pas

Vous allez maintenant réaliser les exercices suivants en Python pour programmer votre robot LEGO.

2.1 Afficher un message sur l'écran

Dans un premier exercice, vous allez afficher simplement le message "Bonjour robot" sur l'écran LCD de LEGO.

La bibliothèque ev3dev offre la classe *Display* pour l'affichage sur l'écran sur le robot : https://python-ev3dev.readthedocs.io/en/ev3dev-stretch/display.html. Un exemple d'utilisation de cette classe est le programme ci-dessous. Ce programme affiche le message "Bonjour robot!" sur l'écran au coordonnées (25,50), ensuite il attend que l'utilisateur appuie sur le bouton central du robot. Sans cette boucle d'attente, le programme affiche rapidement le message puis il s'arrête.

```
#!/usr/bin/env python3
from ev3dev2.display import Display
from ev3dev2.button import Button
import time

lcd = Display()
lcd.clear()
lcd.draw.text((25,50),'Bonjour Robot!')
#Create a button
button = Button()
#Check if 'enter' button is pressed
while not button.enter:
        lcd.update()
        time.sleep(0.01)
```

L'écran LCD du robot contient 178x128 pixels. Le pixel en haut à gauche est le (0,0) et celui à droite en bas est le (177,127). Taper le code de ce programme dans un éditeur. Sauvegarder ce programme dans un fichier nommé, par exemple : monPremierProgramme.py. Ensuite avec WinSCP, transférer le fichier monPremierProgramme.py sur le robot. Ouvrez un terminal sur le robot avec *PuTTY*. Dans le terminal du robot, taper la commande suivante :

```
python3 monPremierProgramme.py
```

Nous pouvons aussi exécuter le programme via le menu "FileBrowser" du robot. Mais, il faut avant rendre le programme exécutable. Dans le terminal du robot, taper la commande suivante :

```
chmod +x monPremierProgramme.py
```

Ensuite, sélectionner sur le robot "FileBrowser", ensuite "PremierProgramme.py" et appuyer sur le bouton central pour exécuter le programme.

2.2 Avancer le robot de 20 cm

Dans cet exercice, vous allez écrire une méthode pour avancer le robot de 20 cm et qu'il affiche sur l'écran "J'avance". La documentation d'utilisation de moteurs est disponible sous ce lien : https://python-ev3dev.readthedocs.io/en/ev3dev-stretch/motors.html Une première solution est d'utiliser la méthode *run_timed* de la classe *LargeMotor*. Il faut aussi indiquer la vitesse et le temps de roulage du robot avec les variables respectives *speed_sp* en dégrés par seconde et *time_sp* en millisecondes. Vous pouvez définir la vitesse d'un moteur avec d'autres unités (RPM, RPS, DPS, DPM) en utilisant les fonctions disponibles dans la classe motor : https://python-ev3dev.readthedocs.io/en/ev3dev-stretch/motors.html#units.

Une deuxième solution est d'utiliser la classe *MoveDifferential* et sa méthode *on_for_distance* pour avancer le robot de la distance souhaitée.

La documentation de cette classe est disponible sous : https://python-ev3dev.readthedocs.io/en/ev3dev-stretch/motors.html#move-differential. Un exemple de code pour la première solution est le suivant :

```
from ev3dev2.motor import OUTPUT_B, OUTPUT_C, LargeMotor

# Connect two large motors on output ports B and C
lmotor, rmotor = [LargeMotor(address) for address in (OUTPUT_B, OUTPUT_C)]
for motor in (lmotor, rmotor):
            motor.run_timed(speed_sp=600, time_sp=500)
# Wait 0.5 seconds while motors are rolling
sleep(0.5)
```

2.3 Reculer le robot de 20 cm

Dans cet exercice, vous allez écrire une méthode pour reculer le robot de 20 cm et qu'il affiche sur l'écran "Je recule". Vous programmez deux solutions : une solution avec les objets moteurs et une autre avec la classe *MoveDifferential*.

2.4 Tourner à droite, tourner à gauche

Pour tourner le robot à droite ou à gauche, on utilise soit les objets moteurs avec une vitesse positive sur une route et une vitesse négative sur l'autre, soit la classe *MoveDifferential* et ses méthodes $turn\_right$ et $turn\_left$.

2.5 Faire un carré

Maintenant, que vous avez appris comment avancer et tourner le robot, vous allez coder un programme pour que le robot fasse un carré de côté 20cm.

2.6 Faire une spirale

Enfin, la spirale.