POPPY ERGO JR

INFORMATION

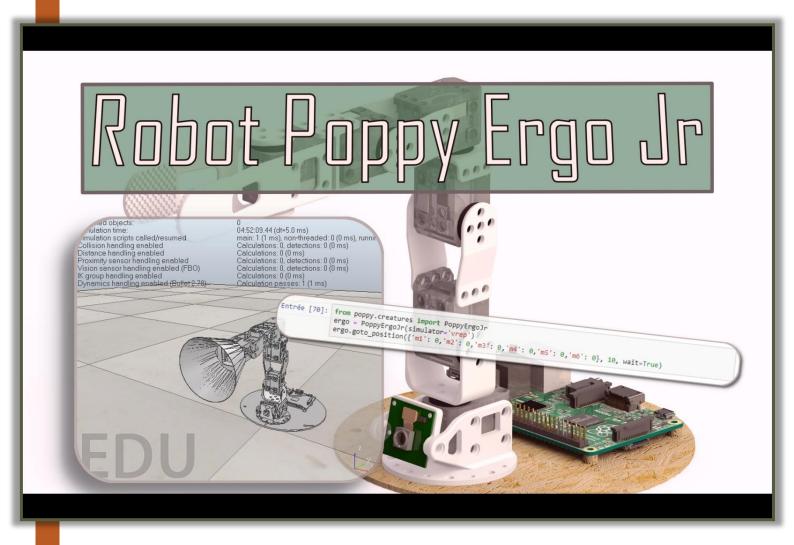


TABLE DES MATIERES

Introduction	
Objectifs	
Consignes	
Le robot Poppy	
Simulation	
Démarrage de la simulation	3
Premiers tests et analyse des servomoteurs	
Programmation	6

INTRODUCTION

OBJECTIFS

Le but de cette activité est de comprendre le fonctionnement informationnel du robot Poppy Ergo Jr en programmant les servomoteurs qui le composent.

CONSIGNES

Suivre ce document.

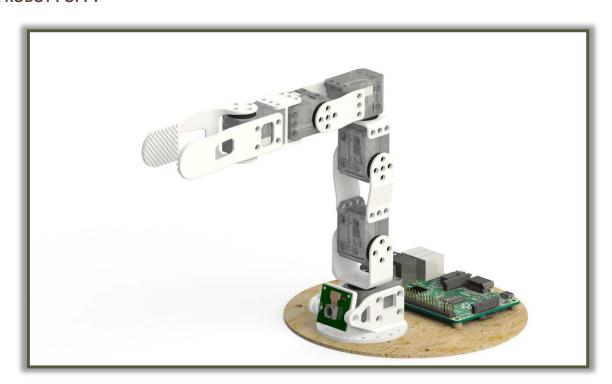
A chaque encadré comme celui-ci se trouve une manipulation à effectuer ou une question à répondre.

Il faudra répondre sur un document Word soigneusement présenté et intitulé

« NOM_Poppy_Information ».

Créer un document Word de réponse et l'enregistrer dans l'espace personnel.

LE ROBOT POPPY



Poppy Ergo Jr est un bras robotisé, composé de **6 servomoteurs** permettant des mouvements presque naturels, ainsi que d'éléments imprimés en 3D. Il est contrôlé par une carte Raspberry Pi, et il est muni d'une caméra pour interagir avec le monde.

Tout dans le robot est laissé transparent et accessible pour promouvoir une approche éducative et une technologie interactive.

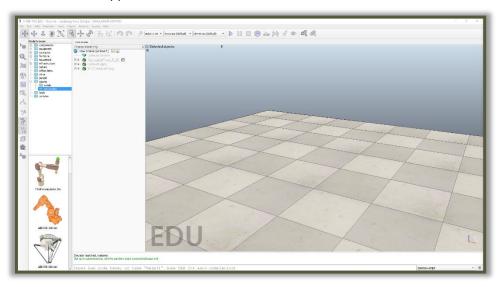
SIMULATION

DEMARRAGE DE LA SIMULATION

Les robots actuels sont de plus en plus sophistiqués et les développeurs de ces robots cherchent à leur faire faire des actions de plus en plus complexes. Afin de limiter les coûts, les premiers tests sont faits sur simulateur.

Ouvrir le logiciel V-REP PRO EDU permettant de simuler via une interface 3D des robots programmables.

La fenêtre ci-dessous devrait apparaître.



Ouvrir ensuite une page Jupyter Notebook en allant dans le menu démarrer, chercher dans la liste le dossier « Anaconda2 (64-bit) » puis choisir « Jupyter Notebook ».

Une page Web s'ouvre avec l'interface de programmation Jupyter.

Ouvrir un nouveau programme Python en cliquant sur « New » puis « Python ».



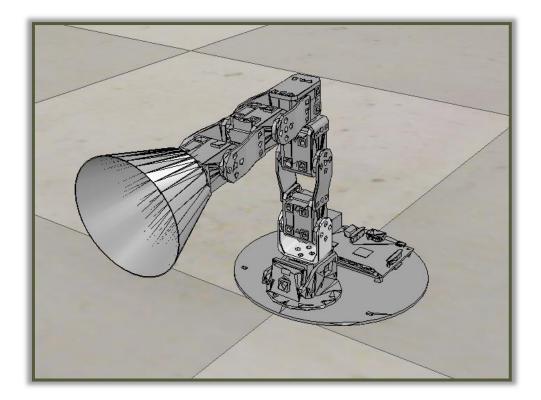
Copier-coller le code suivant dans l'interface de programmation :

from poppy.creatures import PoppyErgoJr

ergo = PoppyErgoJr(simulator='vrep')

Exécuter le code en appuyant sur « Alt » + « Enter ».

Sur la fenêtre du logiciel VREP le robot Poppy Ergo Jr doit alors apparaître.



La programmation du robot se fera sur la fenêtre avec le Jupyter Notebook et le résultat en simulation se fera donc sur la fenêtre VREP. Il est donc recommandé d'avoir en visuel les deux fenêtres en même temps.

Les lignes de code exécutées font appel à des notions de programmation orientée objet (POO).

from poppy.creatures import PoppyErgoJr

Ici on importe dans notre programme l'objet « PoppyErgoJr » depuis la bibliothèque « poppy.creatures » qui est une bibliothèque créée par un développeur permettant de répertorier des fonctions utilisant les robots Poppy. L'objet « PoppyErgoJr » est alors déjà paramétré avec les différentes pièces et paramètres qui le composent.

ergo = PoppyErgoJr(simulator='vrep')

Là, on définit dans notre programme la variable « ergo » comme étant un objet « PoppyErgoJr ». On lance le tout dans le simulateur VREP pour avoir l'interface 3D de l'objet.

PREMIERS TESTS ET ANALYSE DES SERVOMOTEURS

Il est maintenant temps de commander les servomoteurs du robot sur VREP en les programmant.

Dans l'interface de programmation, sur une nouvelle ligne, copier-coller le code suivant :

ergo.m4.goal_position = 90

Exécuter le code en appuyant sur « Alt » + « Enter ».

Observer le résultat sur la fenêtre VREP.

Exécuter la même ligne de code en changeant le 90 par 180, puis 360 et enfin 1500.

Exécuter la même ligne de code en remplaçant cette fois « m4 » par « m1 » puis « m6 ».

Question 1 : Expliquer les lignes de code précédemment exécutées.

Le robot Poppy Ergo Jr est composé de 6 servomoteurs Dynamixel XL-320 avec des caractéristiques bien précises. Toutes les informations et caractéristiques sur ces servomoteurs sont disponibles sur le fichier PDF « XL-320 User guide » mis à disposition sur le site du professeur.

Question 2 : Grâce au guide utilisateur du servomoteur Dynamixel XL-320, indiquer :

- La résolution angulaire du servomoteur.
- La plage angulaire admissible (angle minimal et angle maximal).
- La vitesse de rotation si le servomoteur est alimenté avec 7,4 Volts.

Question 3 : Toujours dans le guide utilisateur, grâce au chapitre « Goal position », indiquer combien de valeurs angulaires le servomoteur peut prendre.

Question 4 : Indiquer sur combien de bit l'angle de position du servomoteur est-il codé. Vérifier alors par un calcul que la précision angulaire est bien de 0,29°.

Question 5 : Justifier alors le fait que sur le système réel, la ligne de commande suivante ne serait pas possible.

ergo.m4.goal_position = 500

Pour le robot Poppy Ergo Jr, dans la bibliothèque « poppy.creatures », des fonctions sont déjà créées pour manipuler les servomoteurs ensembles suivant des lignes de commande fonctionnant en même temps. Par exemple, la fonction « dance » permet de faire bouger les 6 servomoteurs en même temps de façon à faire danser le robot. On peut exécuter cette commande avec la méthode start de cette façon :

ergo.dance.start()

Exécuter la commande précédente et observer le résultat.

Cette fonction est exécutée en boucle.

Trouver un moyen d'arrêter le fonctionnement de cette fonction.

Question 7 : Indiquer sur le compte-rendu la commande permettant d'arrêter la fonction.

Il est possible de contrôler tous les servomoteurs ensemble grâce à la méthode « goto_position() ». Voici un exemple :

```
ergo.goto_position({'m1': 0,'m2': 45,'m3': 45,'m4': 90,'m5': -90,'m6': 45}, 5, wait=True)
```

Question 8 : Tester cette commande et indiquer la fonction du deuxième argument de cette méthode.

PROGRAMMATION

Le but est maintenant de pouvoir contrôler les déplacements du robot grâce à la souris.

Voici un nouveau programme :

```
import Tkinter as tk
import time
app = tk.Tk()
a = app.winfo_pointerx()
b = app.winfo_pointery()
c = app.winfo_pointerxy()
print ("la position de votre souris suivant l'axe x :",a)
print ("la position de votre souris suivant l'axe y :",b)
print ("la position de votre souris avec les coordonnées x et y en même temps :",c)
```

Exécuter ce programme dans le notebook Jupyter et **observer** le résultat.

Ce programme permet de récupérer les coordonnées du pointeur de votre souris à l'écran.

Question 9: Tracer un rectangle qui symbolisera l'écran d'ordinateur et indiquer les coordonnées des 4 coins.

Question 10 : Tracer le repère avec les axes X et Y dans le rectangle.

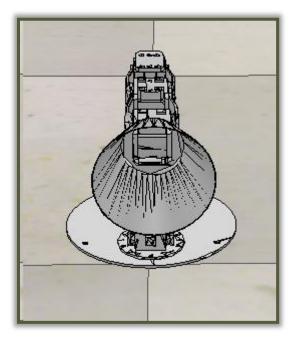
Compléter le programme en ajoutant les lignes suivantes :

```
d = app.winfo_screenwidth()
e = app.winfo_screenheight()
print (d)
print (e)
```

Question 11 : Indiquer quelles informations nous donnent les variables « d » et « e » et compléter le tracé de l'écran en indiquant ces 2 valeurs.

Placer le robot dans sa position de départ et **positionner** la caméra pour observer le robot de face comme sur l'impression écran ci-dessous.

L'objectif est maintenant de bouger le servomoteur m1 en fonction de la coordonnée X du pointeur de la souris.



Contraintes:

- Il faut évidemment prendre en compte que les servomoteurs sont pilotables que sur un certain angle (question 2). Si le pointeur de la souris est à l'extrême gauche de l'écran le servomoteur sera alors à son angle minimum. Si le pointeur de la souris est à l'extrême droite de l'écran alors le servomoteur sera à son angle maximum.
- L'angle 0 est la position centrale du servomoteur. Soit A l'amplitude angulaire totale du servomoteur, alors la plage angulaire accessible est $\left[-\frac{A}{2};\frac{A}{2}\right]$.

Créer un programme permettant de satisfaire l'objectif désiré en prenant en compte les contraintes.

Question 12: Copier-coller le programme sur le compte-rendu.

Compléter le programme en ajoutant un contrôle du servomoteur m5 grâce à la coordonnée Y du pointeur de la souris. Il est impératif de restreindre l'amplitude angulaire pour éviter les collisions !

Question 13: Copier-coller le programme sur le compte-rendu.

Enregistrer votre compte rendu sur

« Ma classe / Restitution de devoirs / Poppy_Information ».