Installations et prise en main de ErgoJr (Simulation ou réel)

Sommaire

Des liens pour commencer (...), p1
Installation pour simulations, p2
Utiliser un simulateur, p3
Faire bouger le robot réel, p4

poppy
Photo de poppy

ErgoJr est un robot composé de 6 servomoteurs pilotés par une carte Raspberry Pi.

Programmable en Python, avec Snap ou Scratch, il est parfait pour l'éducation.

L'idée générale étant de paratger les activités et les ressources créées, vous trouverez un forum de discussions, d'aide et d'échanges pédagogiques :
https://forum.poppy-project.org/

Exemple en vidéo :

Le prix étant assez élevé (300 euros), vous pouvez commencer par utiliser un robot virtuel. Certains groupes d'élèves peuvent donc utiliser le robot réel, tandis que d'autres groupes utilisent un simulateur. Il existe deux types de simulations :

- Le visualisateur (ou Web Viewer). En ligne avec un navigateur type Firefox, Chrome ou Chromium. Il faudra attendre encore un peu avant d'avoir des mouvements proches de la réalité (Septembre 2016).
- Un simulateur : V-REP. Ce logiciel fonctionne très bien, les mouvements sont réalistes et fluides. Préférez ce dernier...

Exemple en vidéo :

Attention !! Il faudra installer des outils pour utiliser ces deux types de simulations.
Si vous avez déjà installé Miniconda (avec Pyzo par exemple), vous avez déjà effectué une grosse partie de l'installation.

DES LIENS POUR COMMENCER :

Présentation du projet : https://www.poppy-project.org/en/robots/poppy-ergo-jr

La documentation (en anglais) : http://docs.poppy-project.org/en/

Un forum de discussions, d'aide et d'échanges pédagogiques : https://forum.poppy-project.org/

Un simulateur : V-REP : http://www.coppeliarobotics.com/downloads.html

Exemple Jupyter/VREP : https://github.com/poppy-project/community-notebooks/blob/master/tutorials-education/poppy-humanoid_poppy-torso__vrep_installation%20et%20prise%20en%20main/poppy%20simul%C3%A9/Ergo_simulation%20prise%20en%20main.ipynb

Un visualisateur : http://simu.poppy-project.org/

Page suivante : Installation pour simulations : Linux/Windows

1 2 3 4

vendredi 2 septembre 2016 par wlaidet

Installations et prise en main de ErgoJr (Simulation ou réel)

INSTALLATION POUR SIMULATIONS : LINUX/WINDOWS

Sommaire

Des liens pour commencer (...), p1
Installation pour simulations, p2
Utiliser un simulateur, p3
Faire bouger le robot réel, p4

Il existe deux façons de simuler votre robot :

- Le visualisateur (ou Web Viewer). En ligne avec un navigateur type Firefox, Chrome ou Chromium. Il faudra attendre encore un peu avant d'avoir des mouvements proches de la réalité (Septembre 2016).
- Un simulateur : V-REP. Ce logiciel fonctionne très bien, les mouvements sont réalistes et fluides. Préférez ce dernier...

Dans les deux cas il faut installer quelques modules avant.

Un lien vers une doc complète de l'installation :
http://docs.poppy-project.org/en/installation/install-poppy-softwares.html

Voici un résumé de la procédure :

1. Téléchargez/Installez Miniconda (*Environnement Python et plus encore*)
2. Ajoutez des librairies de calcul scienfique
3. Installez Jupyter (*Editeur python et bien plus encore, parfait pour créer des cours !*)
4. Ajoutez les librairies *pypot* et *poppy-ergo-jr* spécifiques au robot.

Les installations ont été testées sous Linux Mint17 64 bit, Ubuntu 12.04 32 bit et Windows10.

1) Miniconda :

Miniconda permet d'installer et de faire évoluer simplement un environnement python.

Téléchargez l'installateur qui correspond à votre système ici : http://conda.pydata.org/miniconda.html

Python3.5 fonctionne bien et il faut bien avancer...

Installez ensuite Miniconda :

Sous Windows, il s'agit d'un *.exe*. Il s'installe par défaut dans *C :\Users\Votre_nom_dutilisateur*.
Si vous n'êtes pas à l'aise avec tout cela, vous pouvez définir cet environnement par défaut en cochant les deux cases correspondantes.

http://docs.poppy-project.org/en/installation/install-poppy-softwares.html#install-python

Sous linux, il s'agit d'un *.sh* donc, ouvrez un terminal, allez dans le dossier où est situé le fichier récemment téléchargé. Il faut ensuite rendre exécutable ce fichier puis le lancer. Ce qui donne (avec modifications si nécessaire) :

```
cd Téléchargements/
chmod +x Miniconda3-latest-Linux-x86_64.sh
```

```
./Miniconda3-latest-Linux-x86_64.sh
```

Télécharger

Il s'installe par défaut dans */home/Votre_nom_dutilisateur*.
Si vous n'êtes pas à l'aise avec tout cela, vous pouvez définir cet environnement par défaut.

2) Librairies scientfiques :

Une fois Miniconda installé, vous pouvez ajouter des librairies de calcul scientifque.

Sous Linux ou Windows : Ouvrez un terminal. (Tapez *cmd* dans la recherche logiciels du menu démarrer de Windows).

Si vous avez défini Miniconda comme environnement Python par défaut, tapez simplement :

```
conda install numpy scipy matplotlib
```

Sinon, si vous n'avez pas installé Miniconda comme environnement par défaut, il vous faudra naviguer vers le dossier Miniconda avant de taper les commandes précédentes. Le but étant d'ajouter ces packages dans l'environnement créé par Miniconda...

3) Jupyter :

Jupyter est un éditeur particulier. Il fonctionne dans un navigateur et permet d'ajouter du contenu autour des lignes de codes (des titres, des images, du cours....). C'est génial ! Il possible ensuite d'exporter les pages en *.html* ou en *.pdf* pour créer des cours.

Vous pouvez le tester en ligne sans rien installer pour voir : ici

Pour notre robot, il est préférable de l'installer.

De la même façon que précédemment :

```
conda install notebook jupyter
```

Si vous voulez aller plus loin dans l'utilisation de jupyter, je vous conseille aussi les packages *pandas* et *pyqt*.

4) Librairies poppy :

Vous pouvez regarder ici : http://docs.poppy-project.org/en/installation/install-poppy-softwares.html#install-poppy-software

En résumé, on installe d'autres packages spécifiques au robot comme précédemment, non plus avec *conda* mais avec *pip* :

```
pip install pypot
pip install poppy-ergo-jr
```

Télécharger

(Si vous n'avez pas défini Miniconda par défaut, faites comme précédemment, naviguez dans le bon dossier pour utiliser *pip*.)

Il est aussi possible d'utiliser :

```
easy_install pypot
```

C’est prêt !

Page suivante : Utiliser un simulateur

1 **2** 3 4

vendredi 2 septembre 2016 par wlaidet

Installations et prise en main de ErgoJr (Simulation ou réel)

UTILISER UN SIMULATEUR

Sommaire

Des liens pour commencer (...), p1
Installation pour simulations, p2
Utiliser un simulateur, p3
Faire bouger le robot réel, p4

Une vidéo (désolé pour le son) :

Après avoir installer toutes les librairies :

Commencez par ouvrir V-Rep.

Ouvrez jupyter-notebook *(dans miniconda ou miniconda/bin)*.

Exécutez les lignes suivantes : *(Ctrl+Entrée pour exécuter une cellule)*

```
#Import des librairies et creation du robot
from poppy_ergo_jr import PoppyErgoJr
creature = PoppyErgoJr(simulator='vrep',use_snap=True)#vrep ou poppy-simu
```

Télécharger

```
# un exemple de mouvements : Poppy dit oui
import time
for i in range(2):
    creature.m6.goal_position = -20
    time.sleep(1)
    creature.m6.goal_position = +20
    time.sleep(1)
creature.m6.goal_position = 0
```

Télécharger

```
#un autre exemple de mouvement :
creature.m6.goto_position(+20,1,wait=True)
```

Télécharger

Page suivante : Faire bouger le robot réel

1 2 **3** 4

vendredi 2 septembre 2016 par wlaidet

Installations et prise en main de ErgoJr (Simulation ou réel)

FAIRE BOUGER LE ROBOT RÉEL

Sommaire

Des liens pour commencer (...), p1
Installation pour simulations, p2
Utiliser un simulateur, p3
Faire bouger le robot réel, p4

Sous Snap, vous avez vu la vidéo d'introduction de cet article.

En python :

Connecter votre robot à votre PC.

Ouvrez un navigateur puis allez à l'adresse : *http://poppy/local*

Si vous rencontrez un problème de connexion, il est possible que vous deviez changer le type de connexion (partagée avec d'autre ordinateurs au lieu de DHCP auto).

Ouvrez Jupyter puis : *(Pour exécuter une ligne : Ctrl + Entrée)*

```
1. from poppy.creatures import PoppyErgoJr
2. #Pour importer les librairies
```

Télécharger

```
1. poppy = PoppyErgoJr()
2. #Pour créer un robot appele poppy
```

Télécharger

```
1. for m in poppy.motors:
2.     m.compliant = True
3. #Pour fixer les moteurs
```

Télécharger

Ensuite, vous pouvez définir une position initiale :

```
1. for m in poppy.motors:
2.     m.goto_position(0, 1, wait=True)
```

Télécharger

Ou seulement bouger un moteur :

```
1. poppy.m1.goto_position(50, 1, wait=True)
```

vendredi 2 septembre 2016 par wlaidet

connecter | Plan du site | Mentions légales | RSS 2.0 | Haut de page | SPIP |