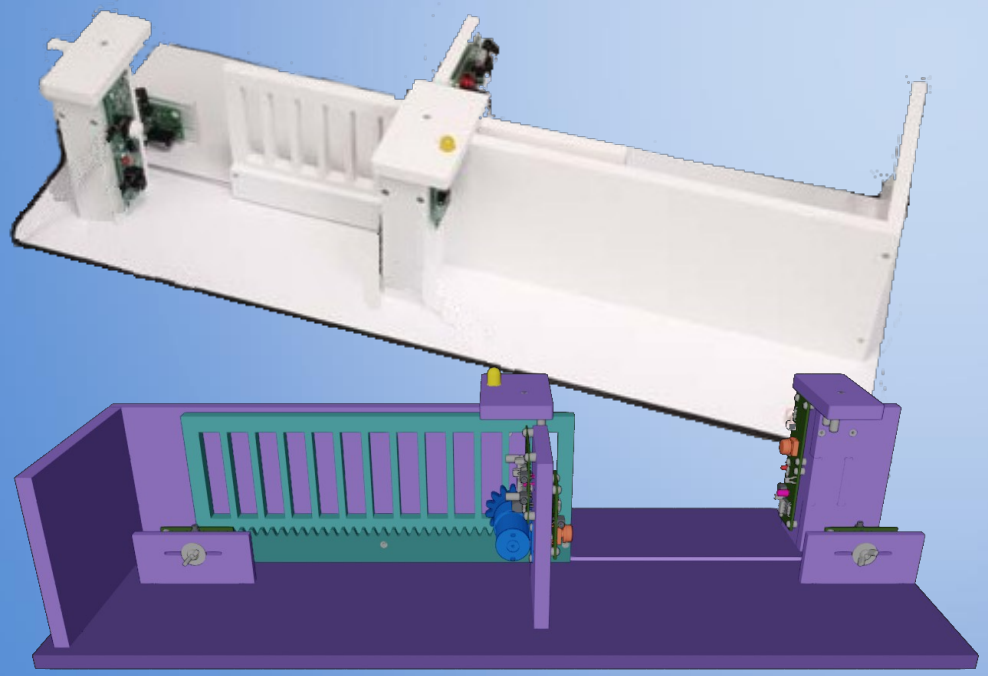
Séquence 3

Algorithme et programmation

SI

LYCÉE LÉONARD DE VINCI

Document Technique : Jumeau numérique d'un portail coulissant

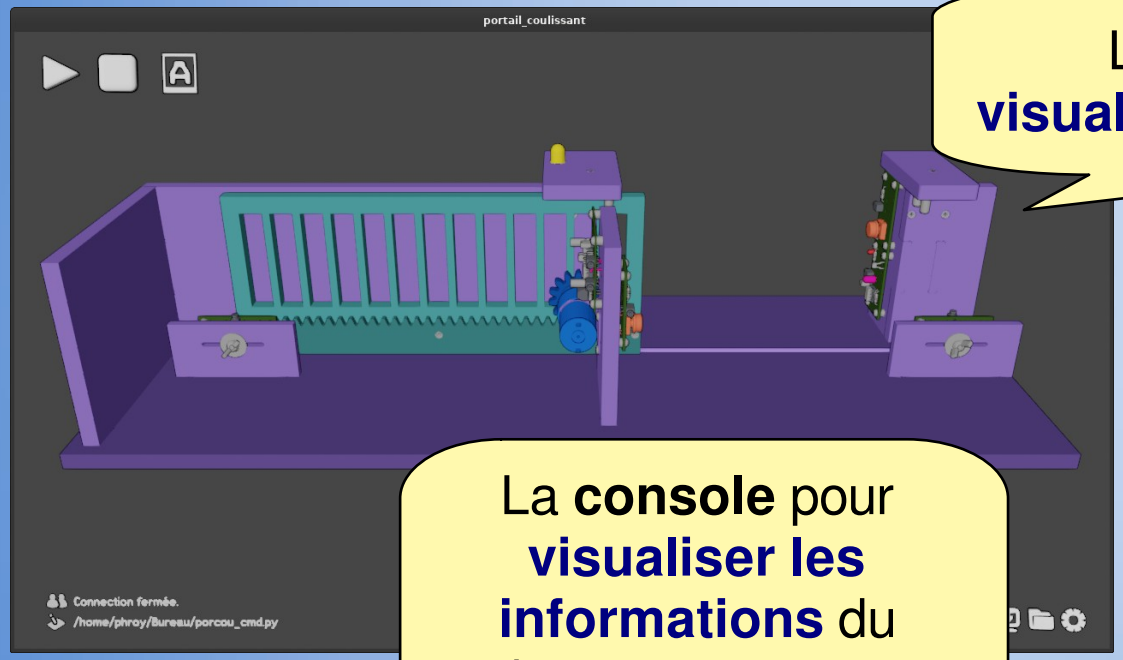


LA FORGE
des communs
numériques
éducatifs

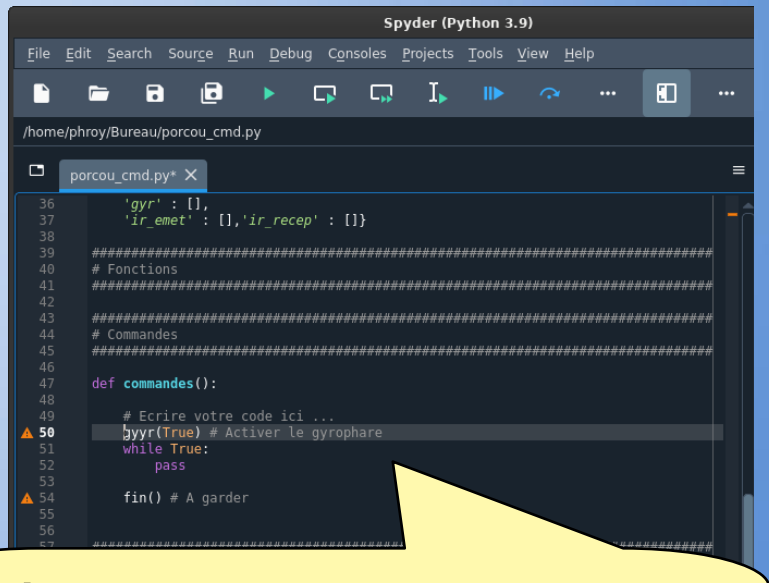
Philippe Roy <philippe.roy@ac-grenoble.fr>

Présentation du jumeau numérique et de son environnement de programmation

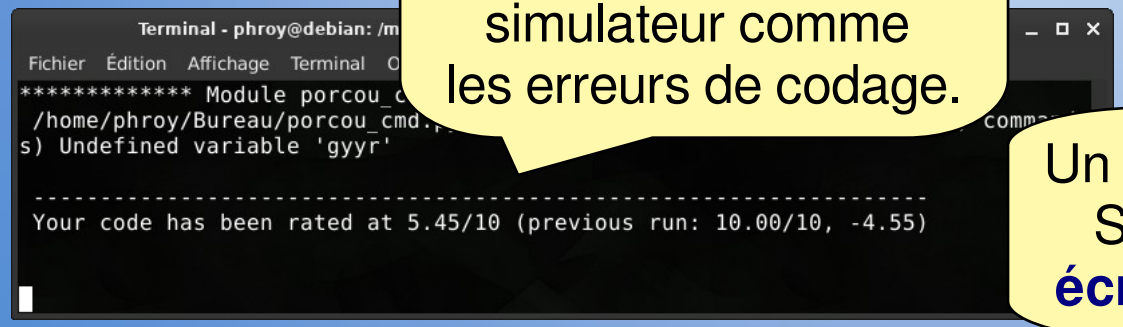
Le jumeau numérique est une maquette numérique qui se commande grâce au langage **Python**. L'interface de programmation se décompose en **3 fenêtres** : un éditeur de texte, le simulateur et la console.



Le **simulateur** permet de **visualiser l'évolution du système.**

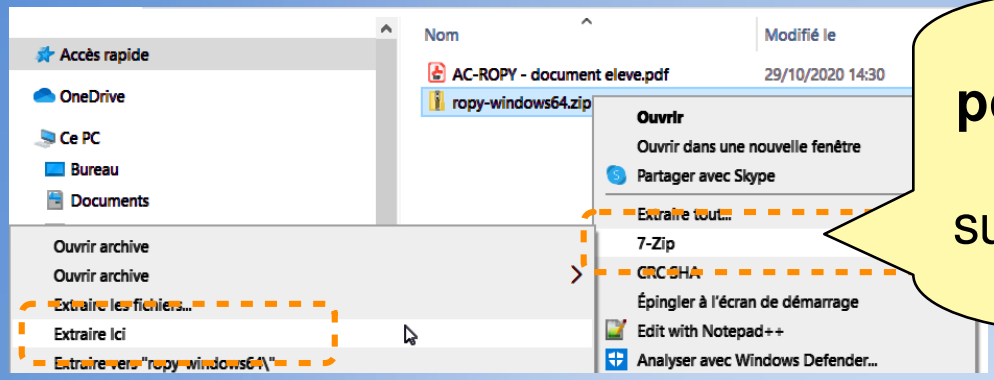


La **console** pour **visualiser les informations** du simulateur comme les erreurs de codage.



Un **éditeur de texte** (Notepad++, Spyder, Atom, Emacs, ...) pour **écrire le programme** en **Python**.

Mettre en place l'environnement de développement



1 : Récupérer l'archive **portail_coulissant-windows64.zip** et la décompresser avec **7-Zip** sur le **bureau**. L'extraction va créer le répertoire portail_coulissant.

Exécuter le programme

Afficher l'**aide**

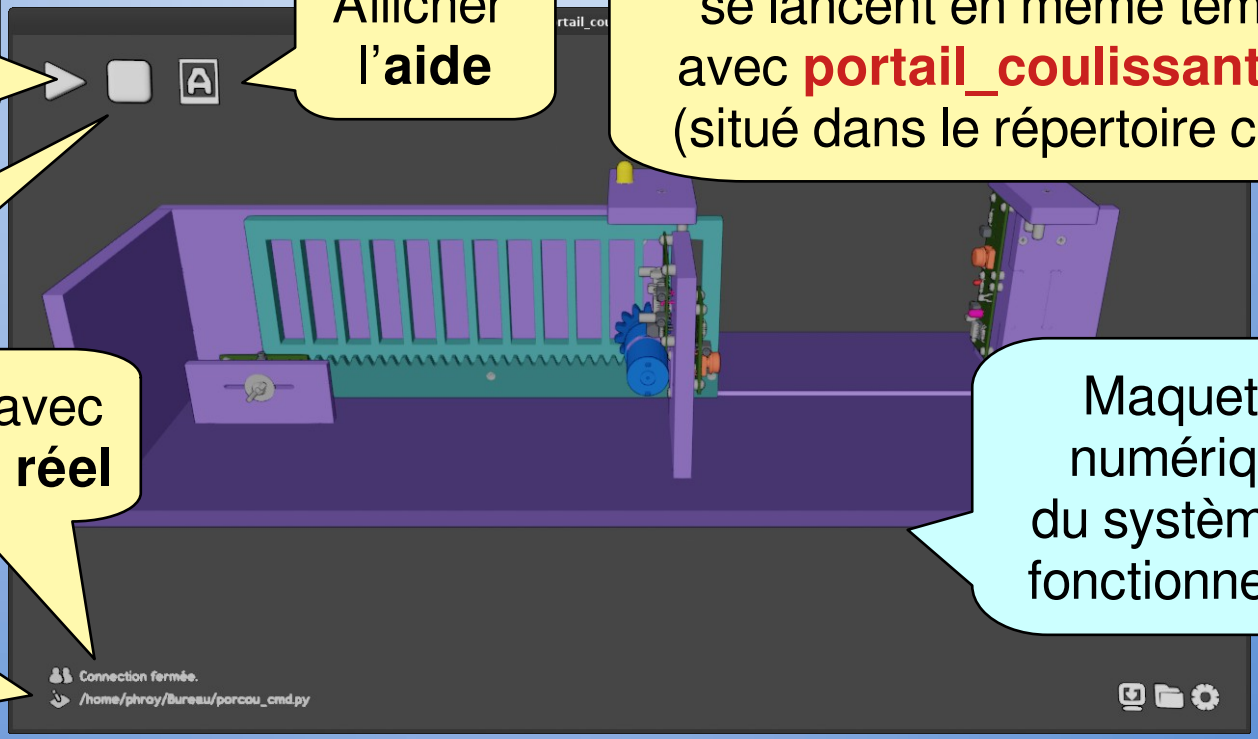
Le **simulateur** et la **console** se lancent en même temps avec **portail_coulissant.bat** (situé dans le répertoire créé).

Arrêter et **réinitialiser**

Message avec le **jumeau réel**

Maquette numérique du système en fonctionnement

Fichier de commandes

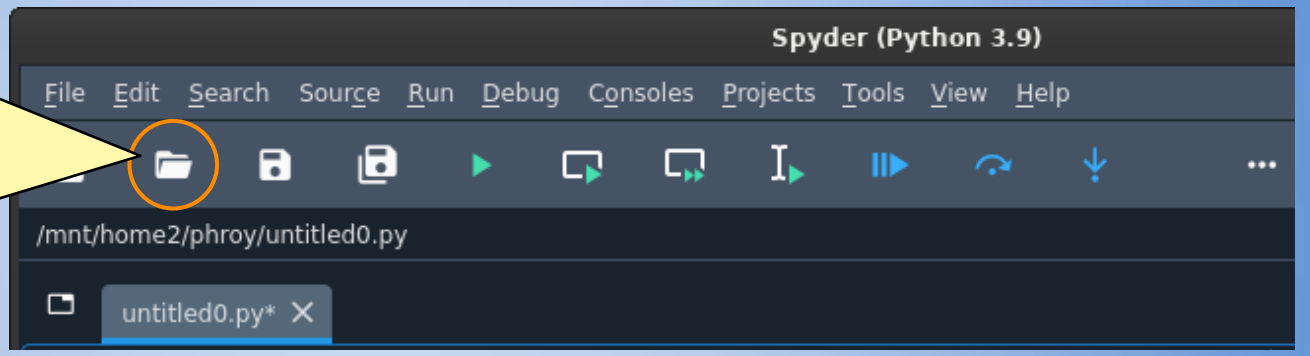


Mettre en place l'environnement de développement

2 : Copier **dans votre répertoire** le fichier de commandes : **porcou_cmd.py** (portail coulissant commandes).

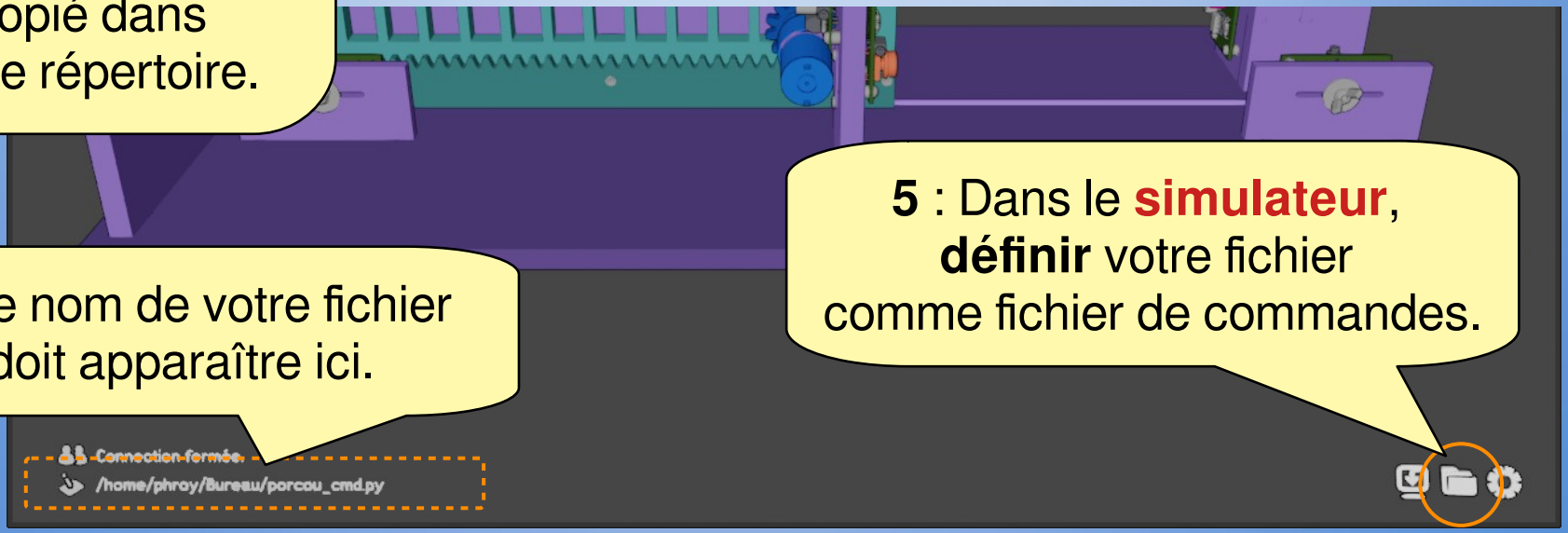
3 : Lancer **Spyder**.

4 : Dans **Spyder** **ouvrir** le fichier de commandes qui a été précédemment copié dans votre répertoire.



5 : Dans le **simulateur**, **définir** votre fichier comme fichier de commandes.

6 : Le nom de votre fichier doit apparaître ici.



Mettre en place l'environnement de développement

8 : **Sauvegarder** le fichier

Attention !

Toujours sauvegarder le fichier avant son exécution.

9 : **Exécuter** le programme.

7 : **Écrire** le code Python.

10 : Si votre code a **une erreur**, la console indique où elle se trouve. Ici la ligne 50 contient une commande indéfinie : `'gyyr'`.

File Edit Search Source Run Deb

/home/phroy/Bureau/porcou_cmd.py

porcou_cmd.py

```
29  ##############################################
30
31  # Brochage du portail coulissant
32  brochage={
33      'bp_ext' : [],'bp_int' : [],
34      'fdc_o' : [],'fdc_f' : [],
35      'mot_o' : [],'mot_f' : [],
36      'gyr' : [],
37      'ir_emet' : [],'ir_recep' : []}
38
39  ##############################################
40  # Fonctions
41  ##############################################
42
43  ##############################################
44  # Commandes
45  ##############################################
46
47  def commandes():
48
49      # Ecrire votre code ici ...
50      gyyr(True) # Activer le gyrophare
51      while True:
52          pass
53
54      fin() # A garder
55
56
57  ##############################################
58  # En: External call << DONT CHANGE THIS SECTION
59  # Fr: Appel externe << NE PAS MODIFIER CETTE SE
60  ##############################################
61
```

conda: base (Python 3.9.13) Completions: conda(base)

commandes

portail_coulissant-17

Terminal - phroy@debian: /mnt/home2/phroy/Bureau/portail_cou

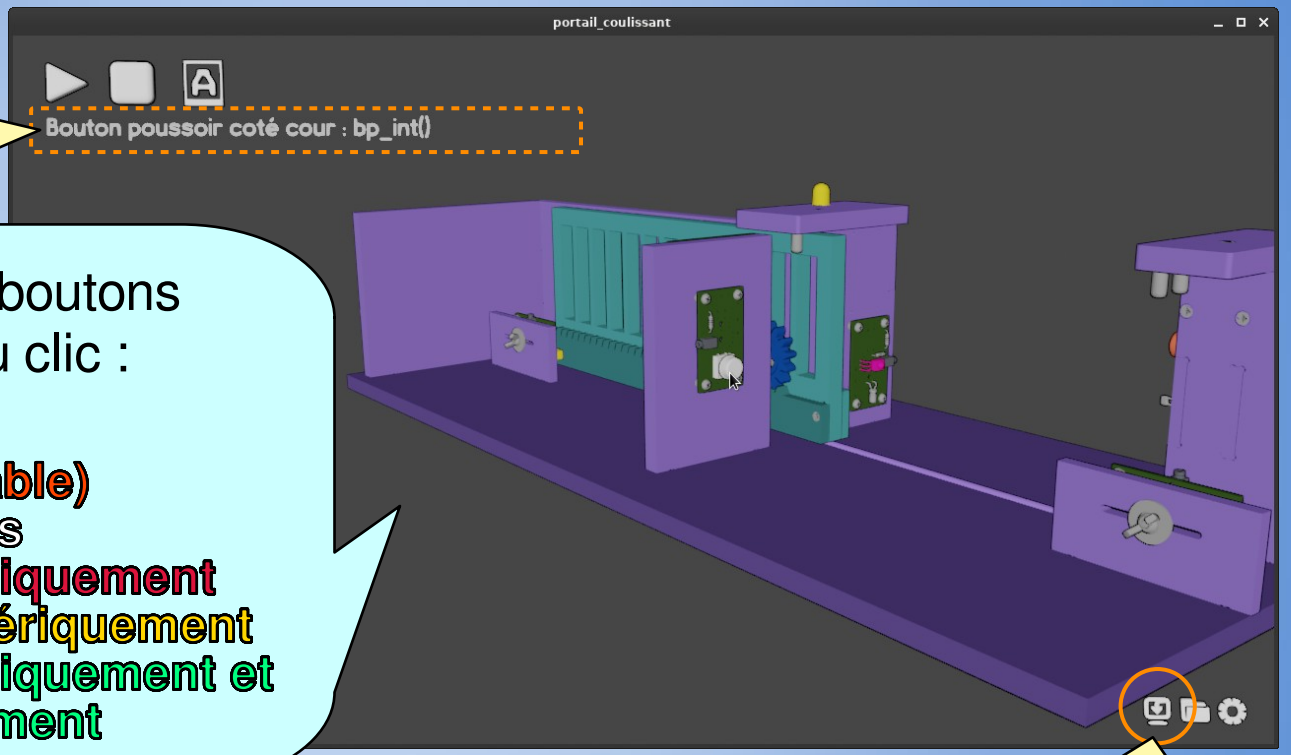
Fichier Édition Affichage Terminal Onglets Aide

```
************* Module porcou_cmd
 /home/phroy/Bureau/porcou_cmd.py:50: error (E0602, undefined-variable, commande
s) Undefined variable 'gyyr'

 -------------------------------------------------------------------
 Your code has been rated at 5.45/10 (previous run: 10.00/10, -4.55)
```

Manipulation de la maquette numérique

Description du composant qui a le focus de la souris



Les capteurs et les boutons sont sensibles au clic :

- **Magenta :** passif
- **Orange :** actif (activable)
- **Blanc :** focus souris
- **Rouge :** activé physiquement
- **Jaune :** activé numériquement
- **Vert :** activé physiquement et numériquement

Réinitialisation de la vue

Le **bouton du centre** sert à **manipuler** le modèle 3D :

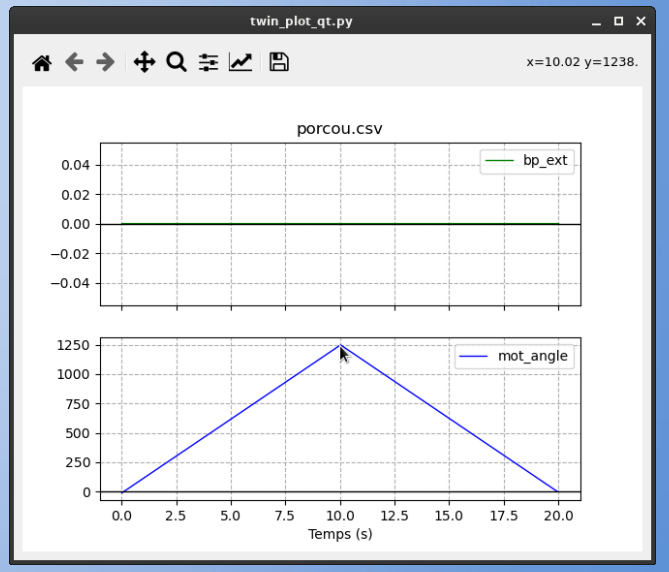
- **Clic centre** : Rotation du mécanisme (Orbit)
- **Clic centre + Maj** : Déplacement du mécanisme (Pan)
- **Clic centre + Ctrl** : Zoom
- **Molette** : Zoom

Acquisition de données

Il possible de suivre les valeurs des entrées/sorties ainsi que des grandeurs physiques du systèmes (position, vitesse).

Dans le script Python, l'**enregistrement des données est activé** par la commande `daq([variables])`. `'[variables]'` est la liste des variables à suivre. Un fichier de données au format CSV sera généré à la fin du cycle. Par exemple : `daq(['bp_ext', 'gyr', 'mot_angle']}`.

L'affichage **du graphique** est déclenchée par `plot([variables])`. `'[variables]'` est la liste des variables à visualiser (variables enregistrées avec la commande `daq`). Par exemple : `plot(['bp_ext','mot_angle']}`.



Données : `'bp_ext'`, `'bp_ext_r'`, `'bp_int'`, `'bp_int_r'`, `'fdc_o'`, `'fdc_o_r'`, `'fdc_f'`, `'fdc_f_r'`, `'mot_o'`, `'mot_f'`, `'gyr'`, `'mot_angle'`, `'mot_vitesse'`, `'portail_x'`, `'portail_vitesse'`, `'ir_emet'`, `'ir_recep'`, `'ir_recep_r'`

`*_r` correspond à la valeur de la variable du jumeau réel.

Jumelage et brochage

Le jumelage est basé sur le **protocole Firmata**. Il faut téléverser le programme **StandardFirmata** (IDE Arduino) vers la carte Arduino afin

- qu'elle transmette les ordres de l'ordinateur vers les actionneurs,
- qu'elle remonte les compte-rendus des capteurs vers l'ordinateur.

Dans le script Python le **jumelage est activé** par la commande `jumeau(brochage)` . `'brochage'` est un dictionnaire faisant le lien entre les composants numériques (objet 3D) et les composants réels :

```
brochage={'composant_num':['type',broche,'mode']}.
```

- **a** pour analogique
- **d** pour binaire (digital)

numéro de la broche (0 à 13)

- **i** pour input (entrée)
- **o** pour output (sortie)
- **p** pour pwm (sortie variable)

Par exemple :

```
brochage={'bp_ext':['d',2,'i'],'gyr':['d',3,'o']}.
```

Le composant 3D `bp_ext` est associé à la broche **2** en mode **entrée**

Le composant 3D `gyr` est associé à la broche **3** en mode **sortie**

Carte de référence du portail coulissant

Boutons :
- Bouton poussoir coté rue : `bp_ext()`
- Bouton poussoir coté cour : `bp_int()`

Capteurs de fin de course :
- Capteur portail ouvert : `fdc_o()`
- Capteur portail fermé : `fdc_f()`

Moteur :
- Ouvrir le portail : `mot_o(ordre)`
- Fermer le portail : `mot_f(ordre)`

Gyrophare :
- Allumé/éteindre : `gyr(ordre)`

Capteur barrage :
- Activation de l'émetteur : `ir_emet(ordre)`
- État du récepteur : `ir_recep()`

Valeur retournée par les capteurs et les boutons
- `True` : actif
- `False` : inactif

Ordre pour les actionneurs
- `True` : activer
- `False` : désactiver

Brochage (composants numériques) **:**
`'bp_ext'`, `'bp_int'`, `'fdc_o'`, `'fdc_f'`, `'mot_o'`, `'mot_f'`, `'gyr'`, `'ir_emet'` et `'ir_recep'`.