

# Projet Drone : Simulation Drone

**Objectifs S2 :** Loi de commande, introduction à l'automatique

- Expérimenter différente loi de commande
- Découvrir ce qu'est un PID
- Découvrir l'influence des différents facteur du PID
- Mettre en place un petit pilote automatique simulé

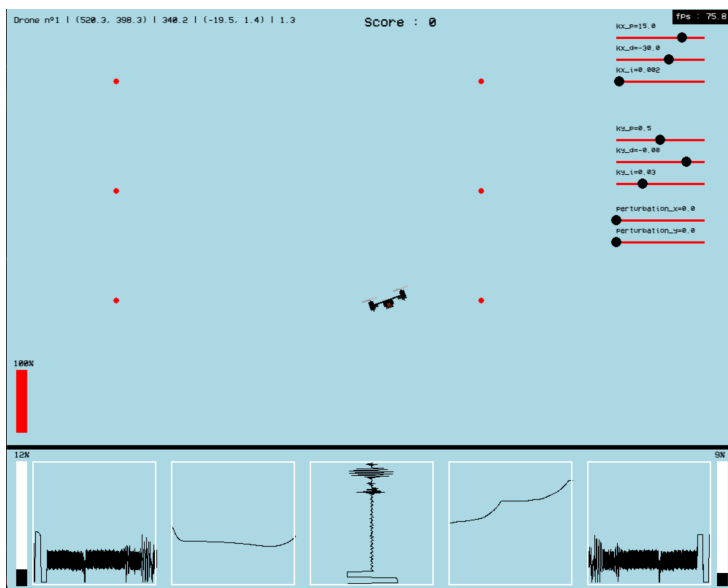
**Supports :**

- Ce document
- Fichiers python sur [Github](#)

**Requirements :** python 3+, éditeur python (vscode, pycharm, ...)

**Library :** pygame

## 1. Démonstration / Explication du simulateur (15 min)



Le drone simulé est en 2 dimensions pour simplifier les choses. Il est propulsé par 2 moteurs (Gauche et Droite).

Pour programmer le drone, on fait appelle à un « *controller* » qui est comme une *télécommande*.

```
import CodeTheMachine_ao.controller
```

Cette télécommande peut être utilisé soit par des consigne utilisateur venant d'une vrai manette ou d'un clavier, mais peut aussi être directement utilisé par un programme automatique.

*On observe sur la simulation ci-dessus : le drone, des points rouges (checkpoints), une jauge rouge (énergie restante), des « slider » (pour controller des paramètres), 2 jauges noirs (puissance des moteurs gauche et droite), 5 graphiques (respectivement de gauche à droite : puissance gauche, position x, différence de puissance gauche-droite, position y, puissance droite)*

Comment utilisé le simulateur ? Un exemple de programme de base est le plus parlant :

```
-----
import CodeTheMachine_ao.controller as control
    ↖ Import le « controller » (~la 'télécommande')

machine = control.LoadMachine(machine_type="drone")
    ↖ Chargement de la machine de type « drone »

Défini une fonction « update » du drone ↴
def power_left_motor(m: control.machine.Drone):
    m.setLeftPower(50)
    ↖ Mets le moteur gauche à 50% de puissance

machine.set_update_function(power_left_motor)
    ↖ Informe la machine la fonction à appelé pour se mettre à jour

control.StartSimulation()
    ↖ Lance la simulation
-----
```

## 2. Résolution des problèmes : programmation (? min)

Pour apprendre à utilisé le simulateur et essayer des lois de commande de base, 5 « niveaux » progressif permettent d'avoir les bases de ce qu'on peut faire.

Dans '/drones\_levels/..' ouvre le niveau 0 et lance le programme, si tout fonctionne alors le simulateur fonctionne correctement et tu peux faire les niveau 1 à 5.

En cas de crash du niveau 0, vérifier l'installation de pygame ainsi que le bon téléchargement et organisation de tout les fichiers.

## 3. Exploration (? min)

Une fois passé tout les niveaux, tu peux essayer de faire ton propre niveau et de programmer ou piloter le drone pour le réussir.

Duplique 'base\_template.py' et renomme ton fichier en 'mon\_programme.py'.

Ouvre ce nouveau fichier python et lis le attentivement.

(Attention, je rappelle l'axe y (ordonnée) et dirigée vers le bas)

A présent tu peux essayer de faire ton propre niveau et de controller le drone pour le réussir !