URJC Escuela de Ingeniería de Fuenlabrada

GSyC

Modelado y Simulación de Robots

PyBullet

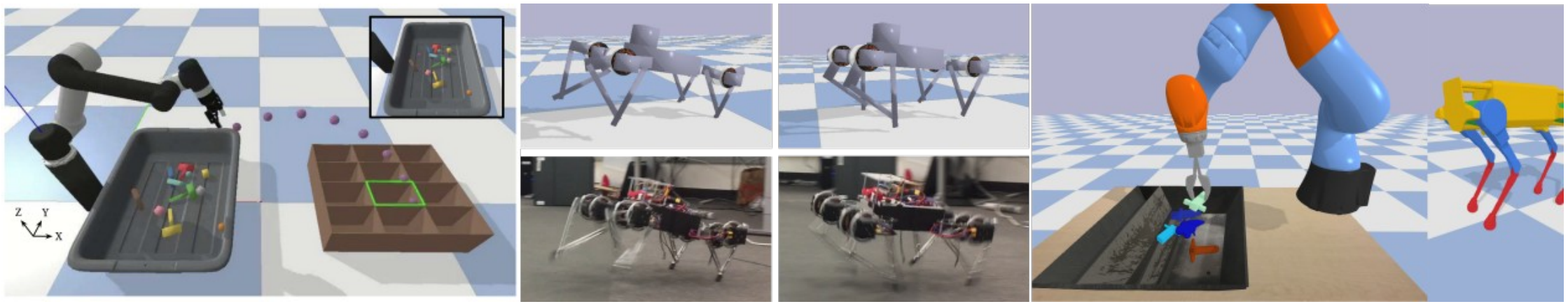
Grado en Ingeniería de Robótica Software

Teoría de la Señal y las Comunicaciones y Sistemas Telemáticos y Computación

Roberto Calvo Palomino
roberto.calvo@urjc.es

Pybullet

- Motor de simulación de físicas
  - Detección colisiones tiempo real
  - VR & games
  - Robotics, machine learning
- Última release, Abril 2022
- https://github.com/bulletphysics/bullet3/



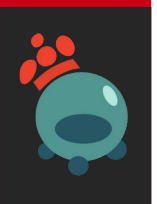
Instalación

- Accede a los ordenadores del laboratorio (físicamente o mediante VNC) https://labs.etsit.urjc.es/vnc/
- Receta válida para laboratorio y vuestros ordenadores

```
$ pip3 install pybullet gym==0.22.0
$ pip3 install numpy==1.26.4
```

- Instalará pybullet en vuestro $HOME, sin necesidad de ser root ni tener permisos de administrador.
- Comprueba que se ha instalado correctamente, visualizando el directorio de ejemplos

```
$ ls  $HOME/.local/lib/python3.12/site-packages/pybullet_envs/examples/
```

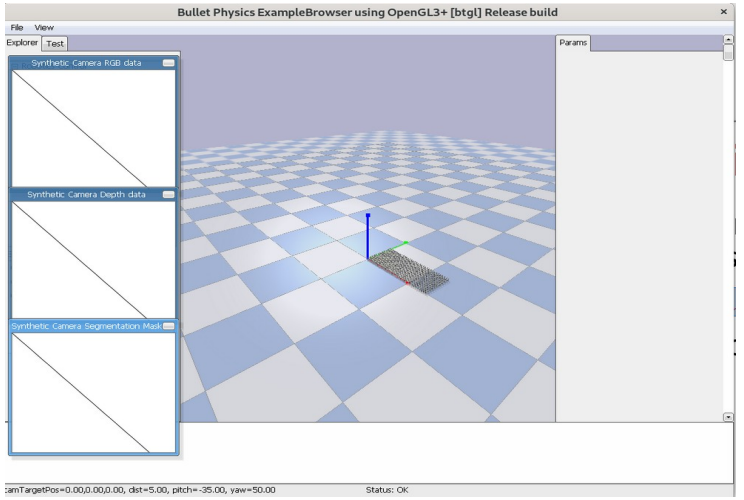



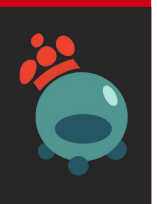
Probando la instalación

- Exporta PYTHON_PATH
  - `$ export PYTHON_PATH=$PYTHON_PATH:$HOME/.local/lib/python3.12/site-packages/`
- Ejecuta el siguiente comando para iniciar el ejemplo de fichas de domino en el motor de físicas

```
$ python3 -m pybullet_envs.examples.dominoes
```

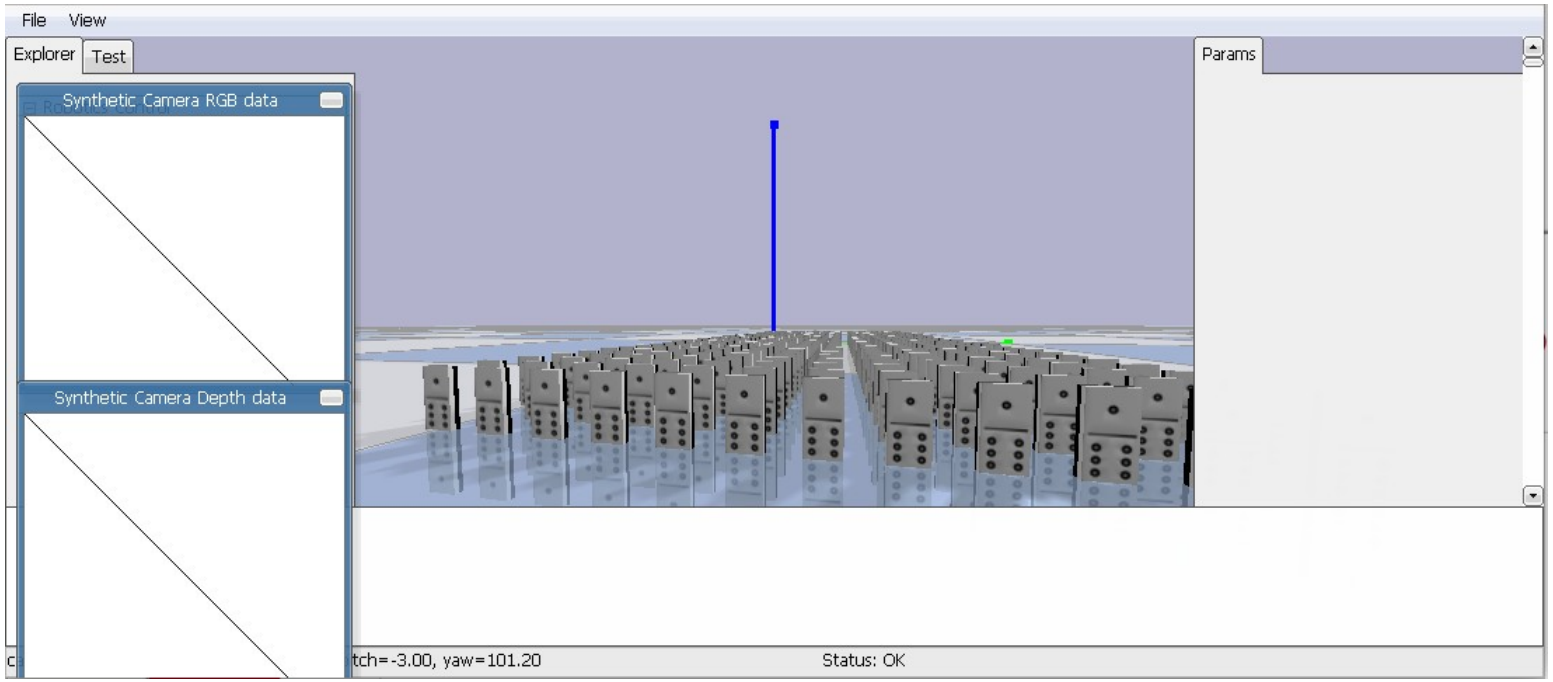
- Deberías ver la interfaz





Probando la instalación

- Alt + Boton-Izqdo: Cambiar Perspectiva
- Rueda ratón: Cambiar Zoom.
- Prueba a generar movimientos en las fichas de domino.

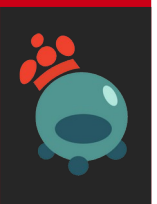


Documentación

- PyBullet Quickstart Guide
  - https://docs.google.com/document/d/10sXEhzFRSnvFcl3XxNGhnD4N2SedqwdAvK3dsihxVUA/edit#

- Carpeta de ejemplos de pybullet

```
$HOME/.local/lib/python3.12/site-packages/pybullet_envs/examples/
```

Hola Mundo en Pybullet

PyBullet

```python
import pybullet as p
import pybullet_data
import time

physicsClient = p.connect(p.GUI)
p.setAdditionalSearchPath(pybullet_data.getDataPath())
p.setGravity(0,0,-9.8)

planeId = p.loadURDF("plane.urdf")

euler_angles = [0,0,0]
startOrientation = p.getQuaternionFromEuler(euler_angles)
startPosition = [0,0,1]

robotId = p.loadURDF("r2d2.urdf",startPosition, startOrientation)

for i in range (10000):
    p.stepSimulation()
    time.sleep(1./240.)

p.disconnect()
```

Imports de las librerías

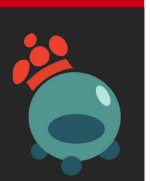
Conectamos motor con GUI

Establecemos gravedad (X,Y,Z)

Cargamos un modelo (plano)

Cargamos un nuevo objeto, con una posición (x,y,z) y una orientación dada en cuaternion (C,X,Y,Z)

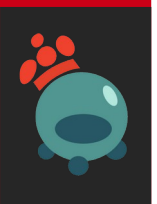
Bucle principal que ejecuta los pasos de la simulación.
Por defecto utilizaremos siempre time step de 1/240 segundos.



Simulación

- **stepSimulation**
  - La simulación avanza solo un “step” (paso) de acuerdo con los pasos establecidos en setTimeStep
  - Util cuando se quiere controlar la simulación donde se necesita procesar cada paso antes de avanzar al siguiente

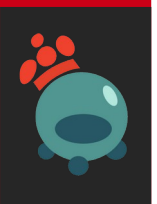
```
1   import pybullet as p
2   import time
3
4   p.connect(p.GUI)
5   p.setGravity(0, 0, -9.8)
6   p.setTimeStep(1./240.)
7
8   for _ in range(1000):
9       p.stepSimulation()
10      time.sleep(1./240.)
```

Simulación

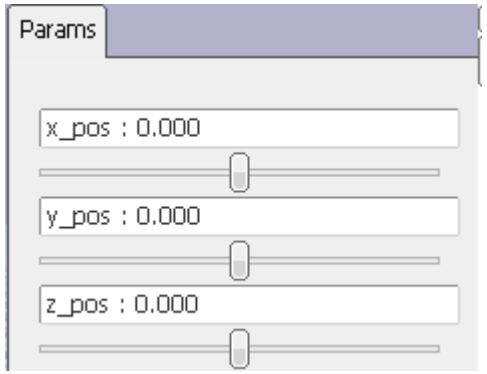
- **SetRealTimeSimulation**
  - El motor de fisicas no realiza pausas en la simulación
  - Ejecuta en tiempo real acorde al RTC del sistema.
  - Depende del rendimiento del sistema.

```
1   import pybullet as p
2   import time
3
4   p.connect(p.GUI)
5   p.setGravity(0, 0, -9.8)
6   p.setTimeStep(1./240.)
7   p.setRealTimeSimulation(True)
8
9   for _ in range(1000):
10      # your code here
```

Parámetros de depuración

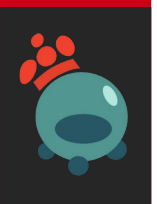
- Añadir parámetros en la pestaña “params”. Útiles para poder cambiar dinámicas en tiempo real.




```
x_pos_id = p.addUserDebugParameter("x_pos", MIN_VALUE, MAX_VALUE, INIT_VALUE)
```

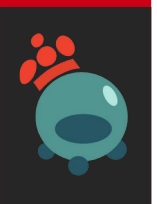
- Desde el bucle de control, puedes obtener los cambios del parámetro de depuración de la siguiente manera

```
x_euler = p.readUserDebugParameter(x_euler_id)
```

Ejercicio 1

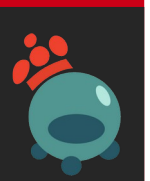
- Partiendo del Hola Mundo modifica el código para los siguientes casos:
  - Diferente gravedad (Marte o La Luna)
    - Mueve el robot con el ratón para comprobar el correcto funcionamiento.
  - Diferentes posiciones (x,y,z) del robot.
  - Diferentes orientaciones utilizando ángulos de Euler
    - Trata de orientar el robot 90º a la izquierda sobre su eje Z
    - Trata de orientar el robot 45º a la derecha sobre su eje Y



Ejercicio 2

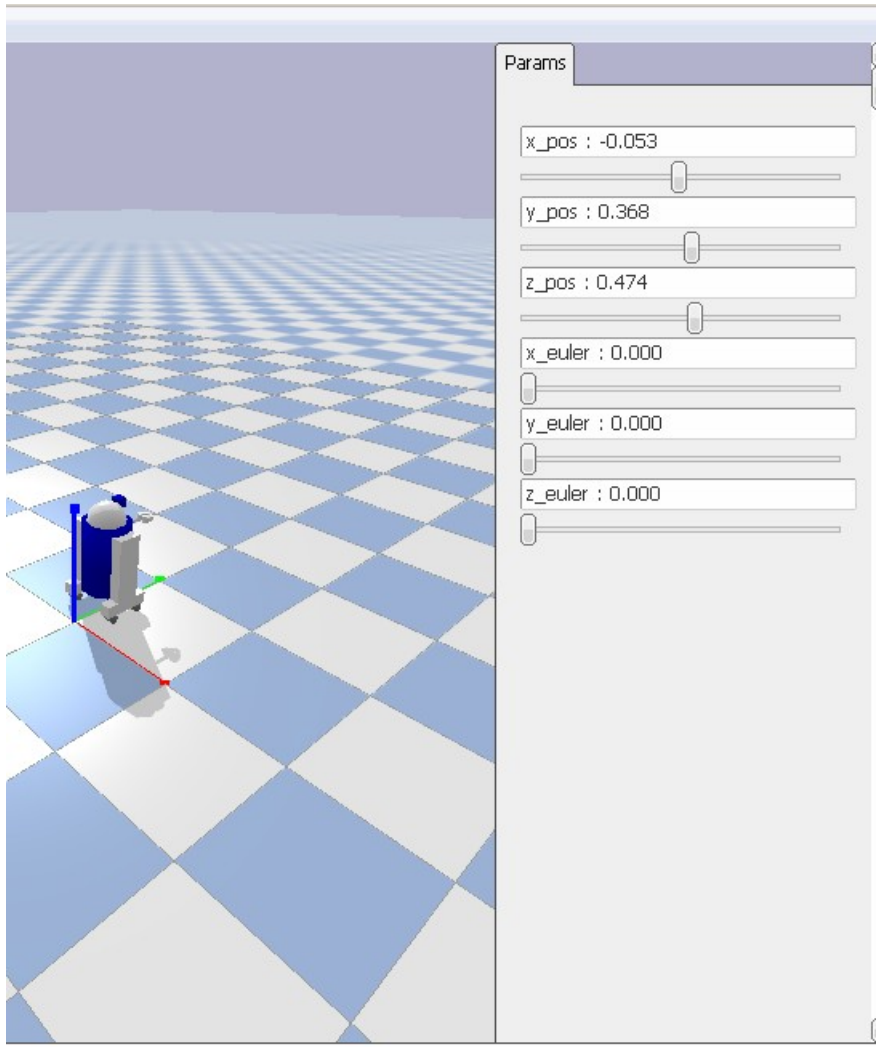
- Genera 6 parámetros de depuración para traslación y rotación de un objeto de la escena.
- Modifica esos sliders y el robot debe cambiar en tiempo real su posición y orientación.
- Los ángulos euler deben estar en radianes.
- Intenta simular el bloqueo de cardán.
- Busca en el manual la función que te permite asignar posición y orientación.

PyBullet



PyBullet

Ejercicio 2



URJC Escuela de Ingeniería de Fuenlabrada