## Введение

Требования к обучающей среде агента могут существенно отличаться. Для ряда задач планирования достаточно сеточного мира, состояние которого легко представляется числовой матрицей, но чтобы обучить робота, способного взаимодействовать с реальность, требования к среде будут значительно выше. Как минимум, понадобится движок, имеющий модель физики и средства визуализации.

В рамках проекта предлагаем воспользоваться открытым движком PyBullet (<https://pybullet.org/wordpress/>), в котором реализована хорошая модель взаимодействия упругих тел. Он широко применяется для моделирования робототехнических систем, RL и расчет игровой физики. Кроме того он довольно хорошо документирован и содержит много примеров примеров кода.

## Установка

Пакет PyBullet устанавливается с помощью команды

pip install pybullet

Здесь я бы рекомендовал сразу создать для работы над проектом отдельную виртуальную среду:

python3 -m venv /path/rlenv

source /path/rlenv/bin/activate

pip3 install pybullet

## Hello world

Рассмотрим стартовый код из документации и разберем его основные идеи:

import pybullet as p

import time

import pybullet\_data

physicsClient = p.connect(p.GUI)#or p.DIRECT for non-graphical

p.setAdditionalSearchPath(pybullet\_data.getDataPath()) #optionally

p.setGravity(0,0,-10)

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

startPos = [0,0,1]

startOrientation = p.getQuaternionFromEuler([0,0,0])

boxId = p.loadURDF("r2d2.urdf",startPos, startOrientation)

#set the center of mass frame (loadURDF sets base link frame) startPos/Ornp.resetBasePositionAndOrientation(boxId, startPos, startOrientation)

for i in range (10000):

p.stepSimulation()

time.sleep(1./240.)

cubePos, cubeOrn = p.getBasePositionAndOrientation(boxId)

print(cubePos,cubeOrn)

p.disconnect()

Движок PyBullet работает как сервер, к которому подключаются клиенты и отправляют запросы на обработку шагов симуляции мира, поэтому сперва нужно подключиться

physicsClient = p.connect(p.GUI)

Объект, загружаемы в PyBullet должен иметь один из допустимых форматов. Т.к. мы здесь занимаемся роботами, то и формат будет urdf (Unified Robot Description Format). Он применяется в промышленной робототехнике для описания структуры и физических свойств различных роботов. Скоро мы разберем его более детально, а сейчас просто загрузим существующие описания

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

boxId = p.loadURDF("r2d2.urdf",startPos, startOrientation)

Цикл симуляции может выполняться по шагам (как сейчас) или в реальном времени.

for i in range (10000):

p.stepSimulation()

time.sleep(1./240.)

В основном на будет более удобен именно такой режим исполнения. Каждый шаг происходит расчет сил, скоростей и положение объектов, загруженных ранее. Без движка PyBullet, пришлось бы проводить все эти вычисления самостоятельно!

## Ещё один пример

Ещё один пример работы PyBullet лежит в текущем каталоге в файле

pybullet\_simple\_kiker.py. Он включает в себя urdf-модель механизма, толкающего шарик и небольшую симуляцию.

## Документация

Стартовый гайд

<https://docs.google.com/document/d/10sXEhzFRSnvFcl3XxNGhnD4N2SedqwdAvK3dsihxVUA/edit#heading=h.2ye70wns7io3>

Интересное: PyBullet можно запустить в колабе

<https://pybullet.org/wordpress/index.php/2021/04/15/pybullet-in-a-colab/>

Примеры кода

Очень рекомендую склонировать (git clone <https://github.com/bulletphysics/bullet3>) себе репозиторий PyBullet, потому что там есть очень много полезных примеров кода, которые понадобятся в процессе работы, лежат в каталоге bullet3/examples/pybullet/.

Интересные объекты для изучения: biped2d\_pybullet.py, minitaur\_test.py, racecar.py, cartpole.py