UNIVERSIDAD SANTO TOMÁS

Facultad de Ingeniería Electrónica

Dockerización de Simulaciones con PyBullet

Autores:

Yossed Riaño

Jeferson Hernández

Miguel Montaña

Asignatura: Digital 3

Docente: [Nombre del profesor]

Bogotá D.C., Octubre de 2025

Índice

1. Introducción

El presente informe documenta el proceso de desarrollo y dockerización de tres simulaciones físicas utilizando la librería **PyBullet**. El objetivo fue ejecutar simulaciones interactivas dentro de contenedores Docker con salida gráfica, garantizando portabilidad y estabilidad en diferentes sistemas operativos.

Las simulaciones realizadas fueron:

- **Brazo Robótico:** un modelo KUKA IIWA articulado.
- **Carrito Deslizante:** cubo móvil sobre superficie plana.
- **Volador:** dron simplificado mediante fuerza de empuje vertical.

2. Objetivos

Objetivo General

Implementar y dockerizar tres simulaciones interactivas en PyBullet, comprobando su correcto funcionamiento físico y gráfico dentro de contenedores aislados.

Objetivos Específicos

- Crear imágenes Docker independientes para cada simulación.
- Ejecutar cada entorno en Windows y Linux con salida gráfica.
- Documentar evidencias de ejecución y análisis de desempeño.

3. Configuración del entorno

3.1. Requisitos en Windows

1. Instalar **Docker Desktop**.
2. Instalar **VcXsrv** y ejecutar con la opción *Disable access control*.
3. Abrir PowerShell o VSCode en la carpeta del proyecto.

3.2. Requisitos en Linux

```
sudo apt install docker.io
xhost +local:docker
```

3.3. Dependencias del contenedor

```
FROM python:3.10-slim
RUN apt-get update && apt-get install -y xvfb x11-apps libgl1 \
    && pip install pybullet \
    && apt-get clean && rm -rf /var/lib/apt/lists/*
```

4. Simulación 1: Brazo Robótico

Descripción

Simulación del brazo KUKA IIWA incluido en PyBullet. Se implementa control de articulaciones y renderizado 3D en tiempo real.

Archivo y Dockerfile

Archivo: `brazo.py`

```
FROM python:3.10-slim
RUN apt-get update && apt-get install -y xvfb x11-apps libgl1 \
    && pip install pybullet \
    && apt-get clean && rm -rf /var/lib/apt/lists/*
WORKDIR /app
COPY brazo.py .
CMD ["python", "brazo.py"]
```

Comandos

Construcción:

```
docker build -t brazo-pybullet -f Dockerfile.brazo .
```

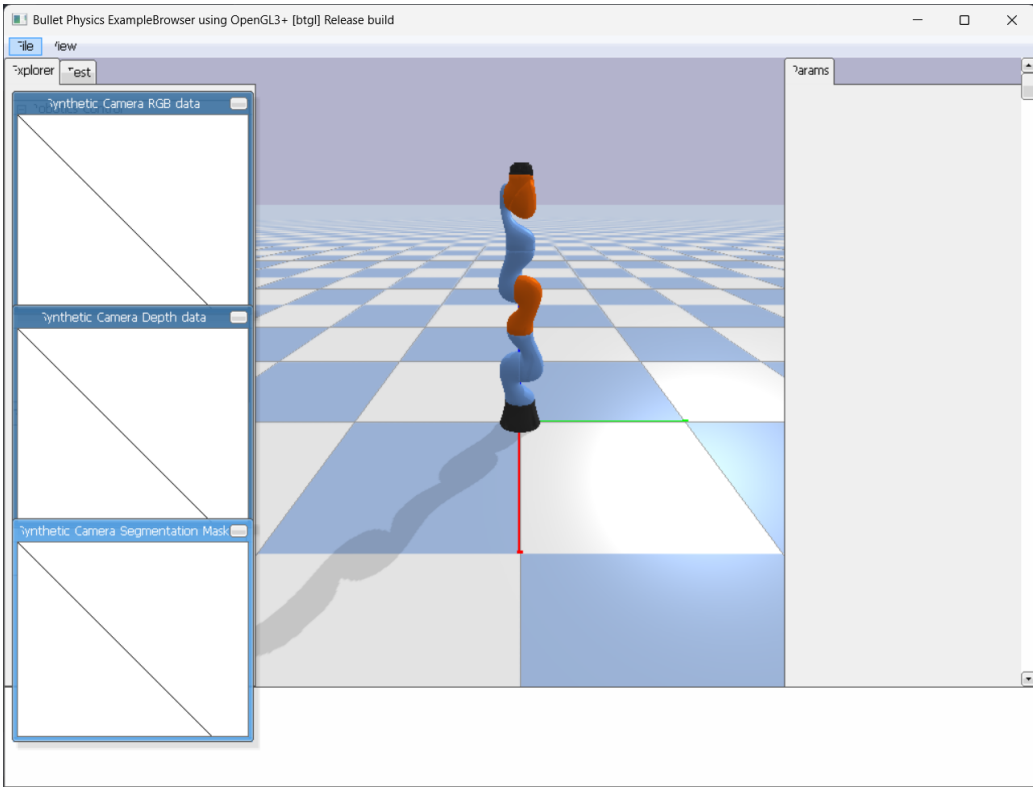
Ejecución (Windows):

```
docker run -it --rm -e DISPLAY=host.docker.internal:0.0 brazo-pybullet
```

Ejecución (Linux):

```
docker run -it --rm -e DISPLAY=$DISPLAY \
    -v /tmp/.X11-unix:/tmp/.X11-unix brazo-pybullet
```

Evidencias




```
PS C:\Users\JEFERSON\Documents\DIGI3\code>
PS C:\Users\JEFERSON\Documents\DIGI3\code> docker build -t brazo-pybullet -f Dockerfile.brazo .
[+] Building 51.0s (6/9)                                                    docker:desktop-linux
 => [internal] load metadata for docker.io/library/python:3.10-slim                        1.4s
 => [auth] library/python:pull token for registry-1.docker.io                              0.0s
 => [internal] load .dockerignore                                                          0.0s
 => => transferring context: 2B                                                            0.0s
 => CACHED [1/4] FROM docker.io/library/python:3.10-slim@sha256:621488956f7e14ca249420e37  0.1s
 => => resolve docker.io/library/python:3.10-slim@sha256:621488956f7e14ca249420e37dbecd59f  0.1s
 => [internal] load build context                                                          0.0s
```

5. Simulación 2: Carrito Deslizante

Descripción

Simulación básica de movimiento sobre un plano horizontal, verificando física de traslación, colisiones y fricción.

Archivo y Dockerfile

Archivo: `carrito.py`

```
FROM python:3.10-slim
RUN apt-get update && apt-get install -y xvfb x11-apps libgl1 \
    && pip install pybullet \
    && apt-get clean && rm -rf /var/lib/apt/lists/*
WORKDIR /app
COPY carrito.py .
CMD ["python", "carrito.py"]
```

Comandos

Construcción:

```
docker build -t carrito-pybullet -f Dockerfile.carrito .
```

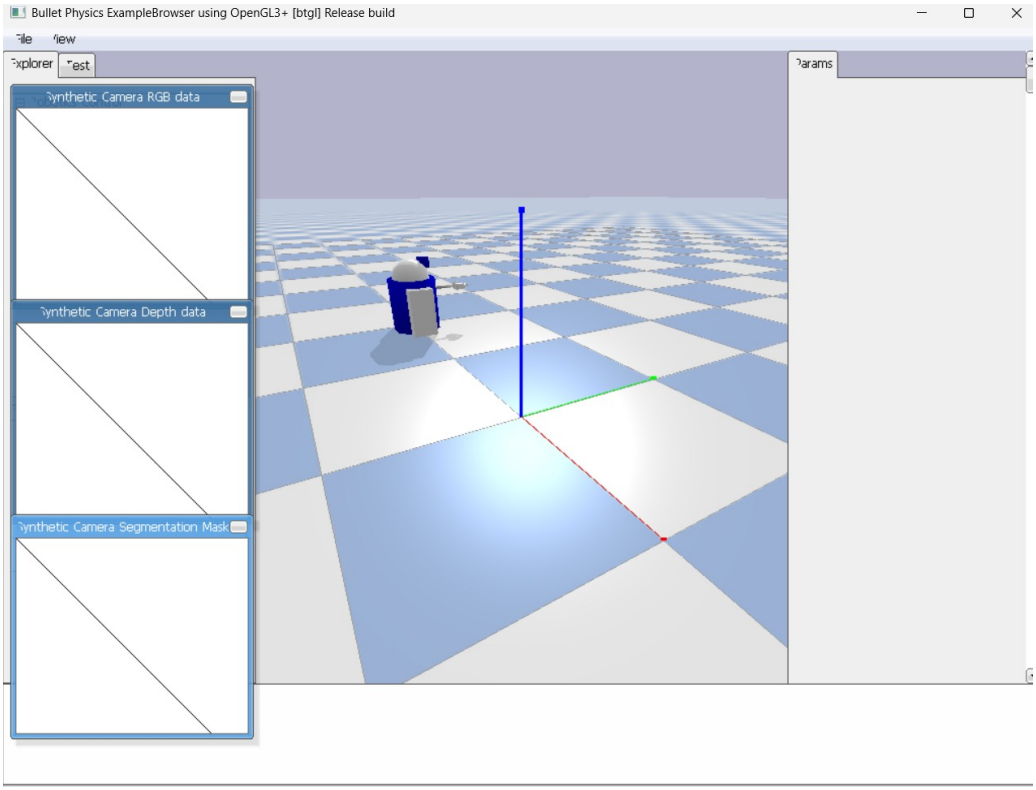
Ejecución (Windows):

```
docker run -it --rm -e DISPLAY=host.docker.internal:0.0 carrito-pybullet
```

Ejecución (Linux):

```
docker run -it --rm -e DISPLAY=$DISPLAY \
    -v /tmp/.X11-unix:/tmp/.X11-unix carrito-pybullet
```

Evidencias




```
PS C:\Users\JEFERSON\Documents\DIGI3\code> docker build -t carrito-pybullet -f Dockerfile.carrito .
>> docker build -t carrito-pybullet -f Dockerfile.carrito .
>>
[+] Building 9.7s (6/9)                                                    docker:desktop-linux
 => [internal] load metadata for docker.io/library/python:3.10-slim                        1.1s
 => [auth] library/python:pull token for registry-1.docker.io                              0.0s
 => [internal] load .dockerignore                                                          0.0s
 => => transferring context: 2B                                                            0.0s
```

6. Simulación 3: Volador (Dron Simple)

Descripción

Simulación de un cubo flotante controlado por una fuerza de empuje vertical, representando el comportamiento de un dron simple.

Archivo y Dockerfile

Archivo: `volador.py`

```
FROM python:3.10-slim
RUN apt-get update && apt-get install -y xvfb x11-apps libgl1 \
    && pip install pybullet \
    && apt-get clean && rm -rf /var/lib/apt/lists/*
WORKDIR /app
COPY volador.py .
CMD ["python", "volador.py"]
```

Comandos

Construcción:

```
docker build -t volador-pybullet -f Dockerfile.volador .
```

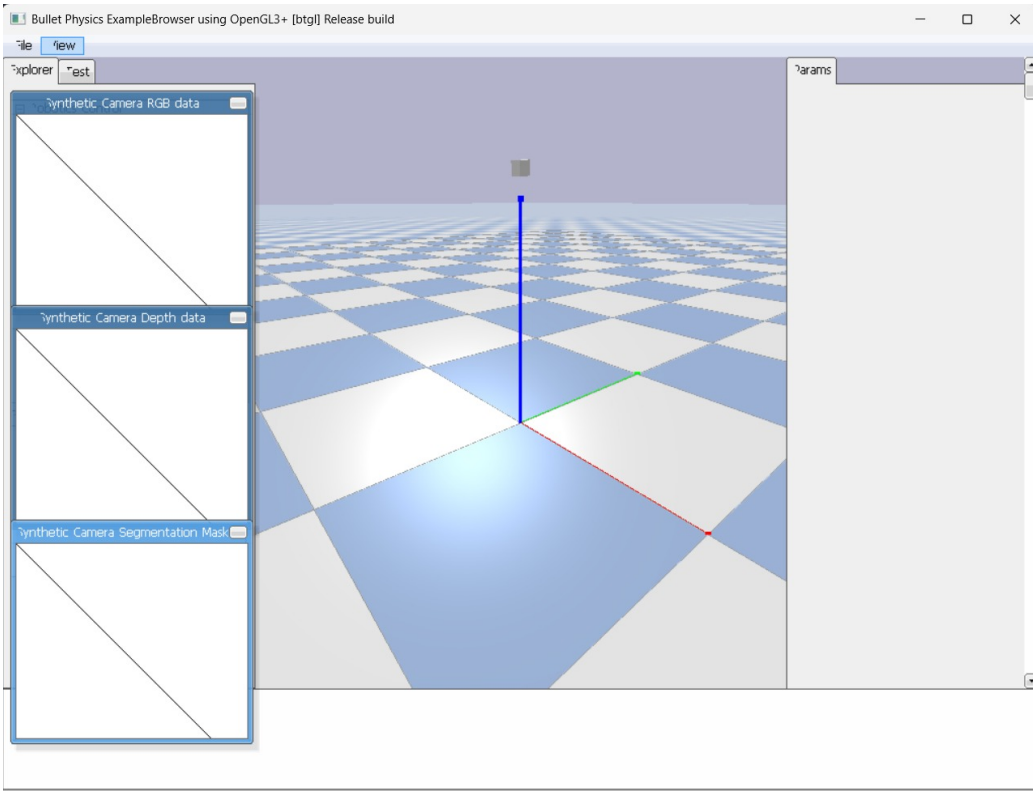
Ejecución (Windows):

```
docker run -it --rm -e DISPLAY=host.docker.internal:0.0 volador-pybullet
```

Ejecución (Linux):

```
docker run -it --rm -e DISPLAY=$DISPLAY \
    -v /tmp/.X11-unix:/tmp/.X11-unix volador-pybullet
```

Evidencias




```
>>
PS C:\Users\JEFERSON\Documents\DIGI3\code> docker build -t volador-pybullet -f Dockerfile.volador .
>>
[+] Building 2.3s (5/8)                                                                    docker:desktop-linux
 => => transferring dockerfile: 318B                                                                   0.0s
 => [internal] load metadata for docker.io/library/python:3.10-slim                                    0.6s
 => [internal] load .dockerignore                                                                      0.0s
 => => transferring context: 2B                                                                        0.0s
 => CACHED [1/4] FROM docker.io/library/python:3.10-slim@sha256:621488956f7e14ca249420e37dbecd59f669a8776bef04  0.1s
 => => resolve docker.io/library/python:3.10-slim@sha256:621488956f7e14ca249420e37dbecd59f669a8776bef0429aa89a  0.0s
 => [internal] load build context                                                                      0.0s
```

7. Análisis y resultados

- Docker permitió aislar dependencias gráficas y físicas sin conflictos.
- Las simulaciones fueron ejecutadas correctamente en Windows y Linux.
- Se logró interacción gráfica estable con PyBullet a través de VcXsrv.

8. Conclusiones

- Se logró dockerizar tres simulaciones PyBullet con soporte visual.
- Docker facilita la portabilidad y replicabilidad de los entornos.
- PyBullet demostró ser ideal para simulaciones ligeras en 3D.

9. Referencias

- PyBullet Documentation: https://pybullet.org/wordpress/
- Docker Documentation: https://docs.docker.com/
- VcXsrv Display Server: https://sourceforge.net/projects/vcxsrv/