Solución Reto Movilidad Urbana

Manual de Instalación

Modelación de Sistemas Multiagentes con Gráficas Computacionales

Melissa Garduño Ruiz **A01748945**
Omar Rodrigo Sorchini Puente **A01749389**
Emilio Ríos Ochoa **A01378965**

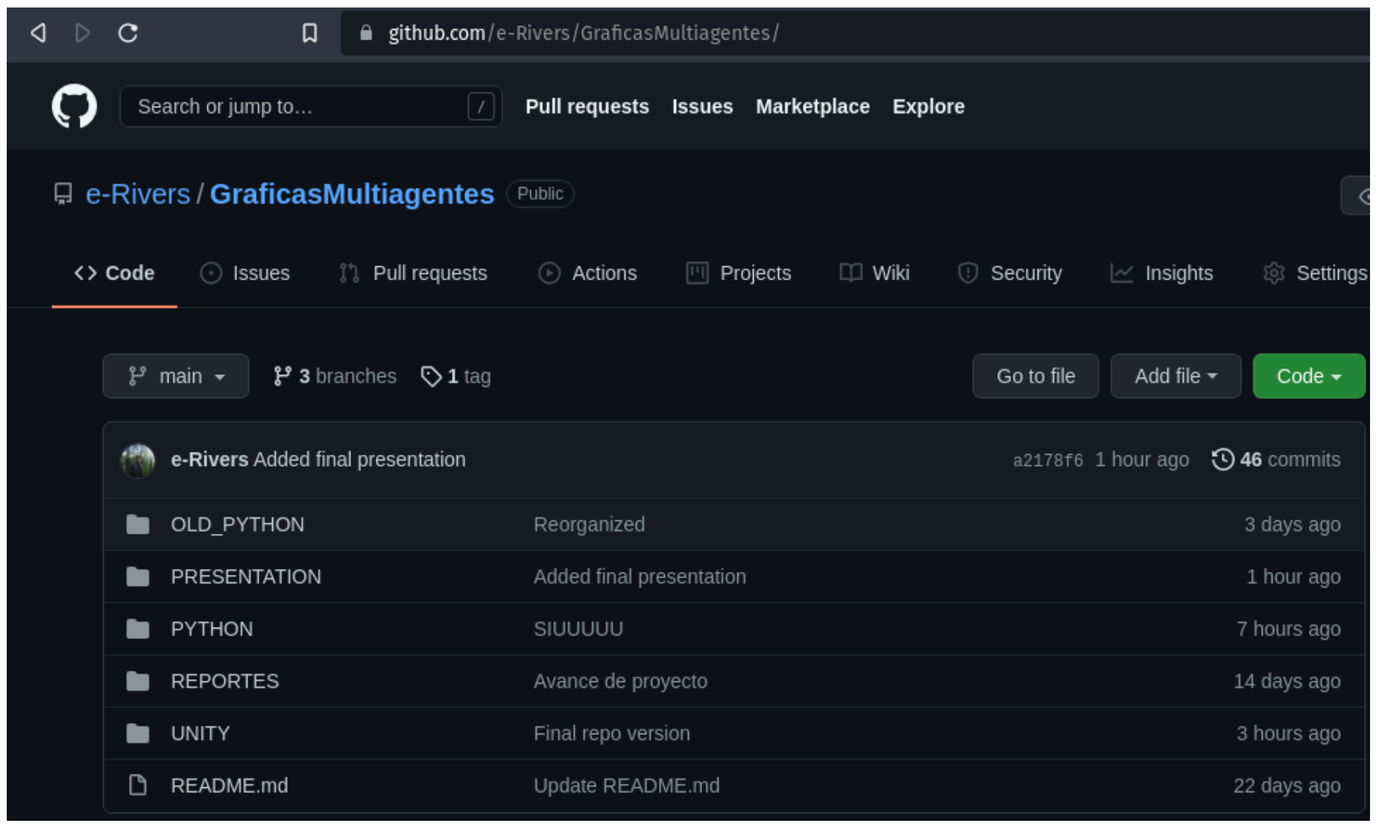
Requerimientos Previos

Es necesario tener instalado:

- Unity versión 2020.3.21f1
- Python 3.x con las librerías Flask y Mesa

1

Clonación del repositorio



https://github.com/e-Rivers/GraficasMultiagentes/

Lo primero a realizar es clonar el repositorio desde Github hacia la máquina de manera para acceder a su contenido, la liga para esto es la mostrada debajo dela imagen contenida en esta página.

Configuración del Cliente (Unity)

2

```
21
22 public class AgentController : MonoBehaviour {
23     string serverUrl = "https://reto-robot-python-flask-a01748945-wacky-gazelle-yg.mybluemix.net";
24     //string serverUrl = "localhost:8000";
25     string getCarsEndpoint = "/getCars";
26     string sendConfigEndpoint = "/init";
27     string updateEndpoint = "/update";
28
```

Si quisiera utilizarse el servidor de manera local, se debe ingresar a la carpeta del repositorio clonado siguiendo la siguiente jerarquía GraficasMultiagentes > UNITY > Assets > Scripts, una vez en Scripts, abrir el archivo AgentController.cs y descomentar la línea 24 y comentar la línea 23.

3

Por otro lado, si se quisiera utilizar el servidor de IBM (más lento) no es necesario realizar ninguna configuración adicional y puede saltarse hasta el paso 4.

Ejecución del Servidor (Python)

(SÓLO SI SE DESEA EJECUTAR USANDO EL SERVIDOR DE MANERA LOCAL)

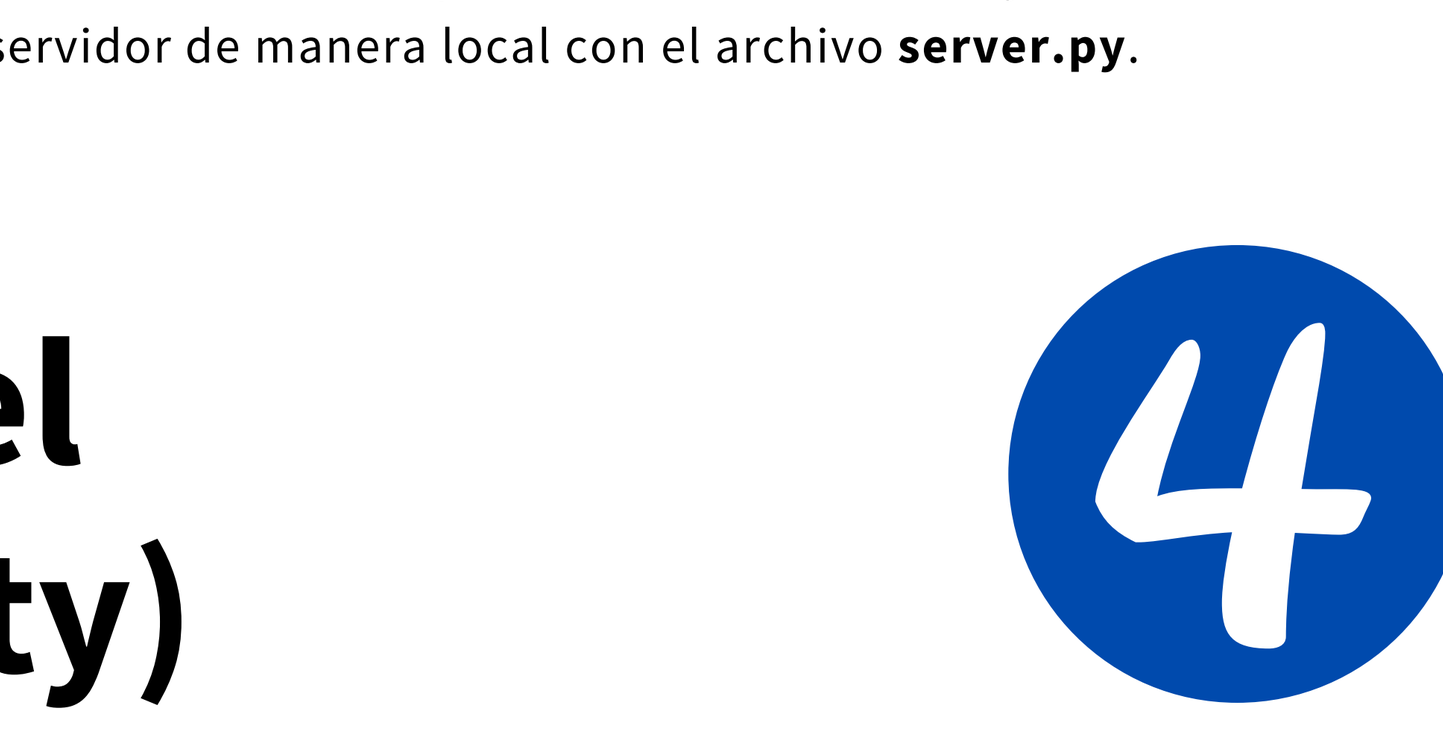
```
emiliorivers@garivers in repo: GraficasMultiagentes on ma
λ cd PYTHON

emiliorivers@garivers in repo: GraficasMultiagentes/PYTHON
λ python3 server.py
* Serving Flask app 'Traffic Simulation' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a p
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://localhost:8000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 145-304-246
```

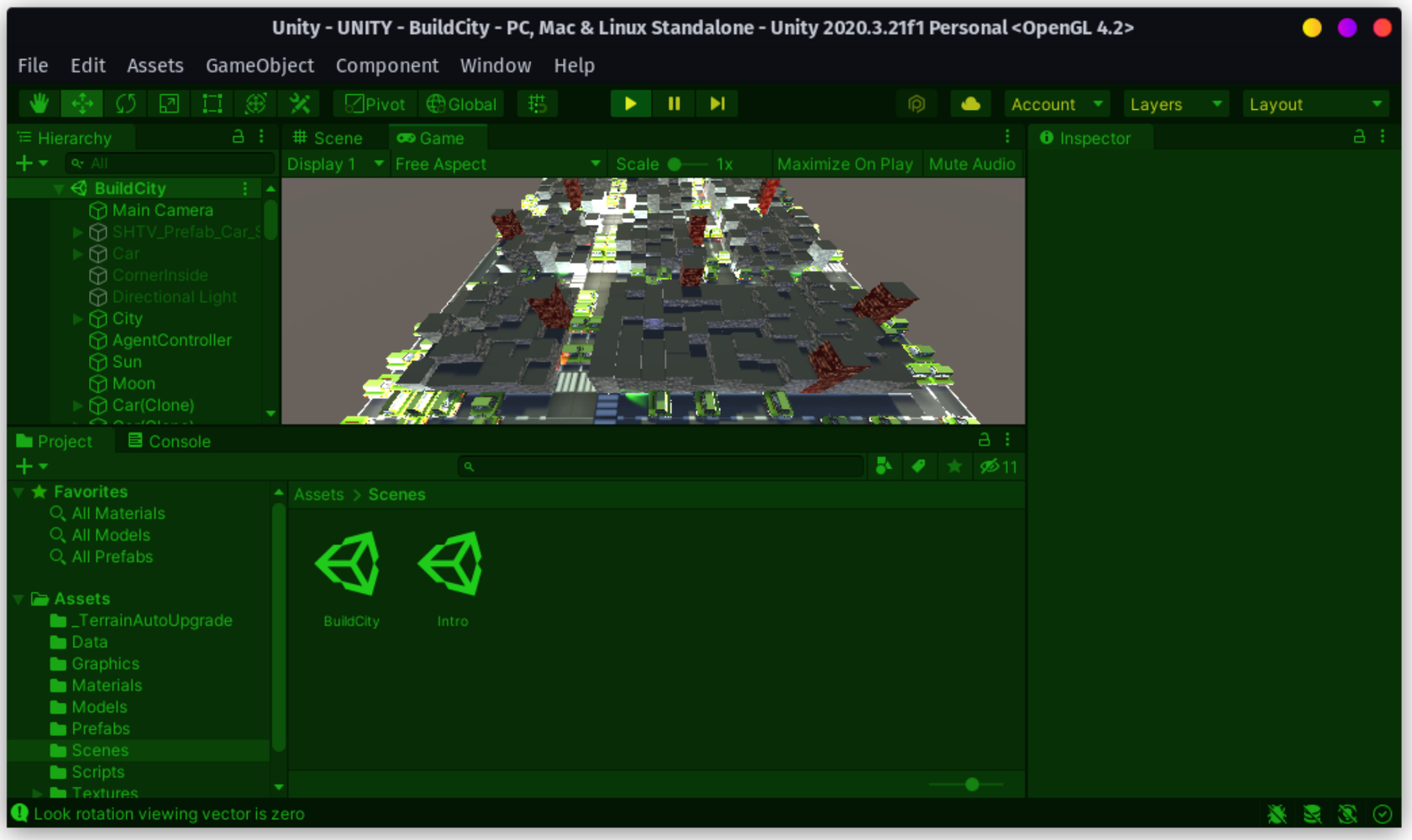
Una vez descargado el repositorio, se debe acceder a la carpeta nombrada PYTHON y realizar el comando de ejecución para levantar el servidor de manera local con el archivo **server.py**.
Comando: *python3 server.py*

Ejecución del Cliente (Unity)

4

Con el servidor en línea, se procede a abrir Unity con la carpeta de UNITY y una vez dentro, se debe cargar la escena de Build City.
En caso de querer modificar algún parámetro, se puede proceder a clickear el objeto Agent Controller y modificar lo que se necesite en el menú lateral de la derecha.





Para ejecutarlo, se le da click al play en la parte central superior de la interfaz de Unity para visualizar el modelo en acción.

OJO: si no funciona la primera vez por problemas de indexación, basta con detener la simulación y volverla a reanudar para que funcione correctamente de nuevo.

5 *BONUS. Ejecución desde el servidor default de MESA

```
└λ python3 mesaServer.py
Interface starting at http://127.0.0.1:8521

(brave:579698): Gtk-WARNING **: 23:00:10.319: Theme parsing error: gtk.css:5822:26: '-shadow' is not a valid color name

(brave:579698): Gtk-WARNING **: 23:00:10.319: Theme parsing error: gtk.css:5825:14: not a number

(brave:579698): Gtk-WARNING **: 23:00:10.319: Theme parsing error: gtk.css:5826:13: not a number

(brave:579698): Gtk-WARNING **: 23:00:10.319: Theme parsing error: gtk.css:5827:11: Expected a length
Opening in existing browser session.
libva error: /usr/lib/dri/iHD_drv_video.so init failed
[579736:579736:0100/000000.007928:ERROR:sandbox_linux.cc(376)] InitializeSandbox() called with multiple threads in process gpu-process.
Socket opened!
{"type":"reset"}
WARNING:tornado.access:404 GET /favicon.ico (127.0.0.1) 37.40ms
```

Para su visualización en MESA, basta con correr el archivo **mesaServer.py** contenido dentro de la carpeta PYTHON para que se abra una pantalla en el navegador predeterminado de su preferencia y con ello podrá verlo en formato 2D en un tablero.

Los vehículos serían los círculos, los semáforos los cuadros rojos o verdes y los obstáculos serían todos aquellos cuadros azules, dejando los grises como camino libre para los agentes.