



Modelación de sistemas multiagente con gráficas computacionales

Equipo 5

David Rodríguez Fragoso A01748760

Erick Hernández Silva A01750170

Israel Sánchez Miranda A01378705

Renata Montserrat de Luna Flores A01750484

Roberto Valdez Jasso A01746863

Manual del sistema

Asesores

Profesor Sergio Ruiz Loza

Doctor Jorge Adolfo Ramírez Uresti

Tabla de contenidos.

| | |
|---|----------|
| Descarga del sistema. | 2 |
| Requisitos. | 2 |
| Descarga. | 2 |
| Extracción. | 2 |
| Escena de Unity e importación de assets. | 3 |
| Escena principal de Unity. | 3 |
| Extracción del Unity package. | 4 |
| Importación de assets. | 5 |
| Servidor en Python. | 6 |
| Archivo del servidor. | 6 |
| Correr el servidor localmente. | 7 |
| Probar la simulación. | 7 |
| Configuraciones y consideraciones del sistema. | 8 |
| Archivo de configuración. | 8 |
| Correr el sistema en la IBM Cloud. | 8 |

1. Descarga del sistema.

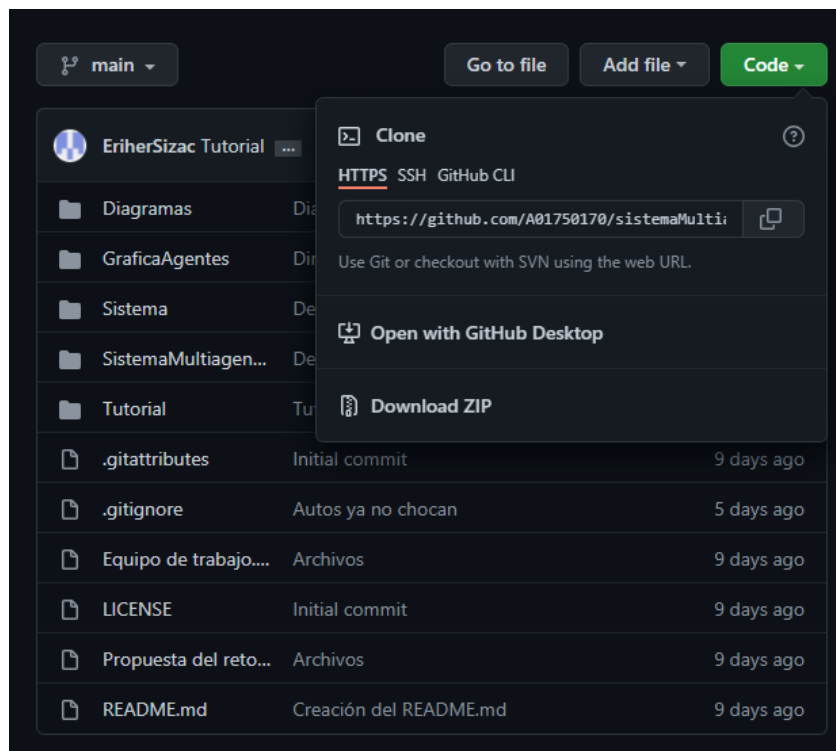
1.1. Requisitos.

Los requisitos para poder descargar y ejecutar el sistema en condiciones óptimas son los siguientes:

- Tener Unity con la versión 2020.3.22f1.
- Tener Python 3.8 o superior.
- Contar con algún editor de código, IDE o manera de ejecutar códigos de Python.
- Espacio libre en disco duro de mínimo 1 GB.
- Memoria RAM mínima de 8GB.
- Sistema operativo Windows 10/ MacOS más reciente (se recomienda correr el sistema en Windows).

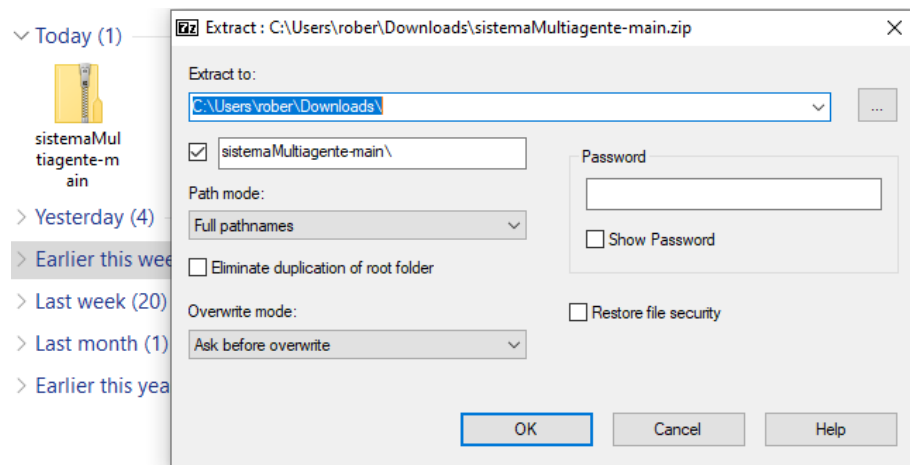
1.2. Descarga.

Para descargar el archivo se debe de ingresar a la siguiente liga: [A01750170/sistemaMultiagente \(github.com\)](https://github.com/A01750170/sistemaMultiagente). Y hacer click en el botón verde que dice código y descargar el Zip del repositorio.



1.3. Extracción.

Al descargar el archivo, este se encontrará comprimido en un Zip, se le debe hacer click derecho y luego click en la opción de “Extraer en” y escoger el destino en el que se desea extraer el archivo.



Dando como resultado una carpeta que contendrá los siguientes archivos:

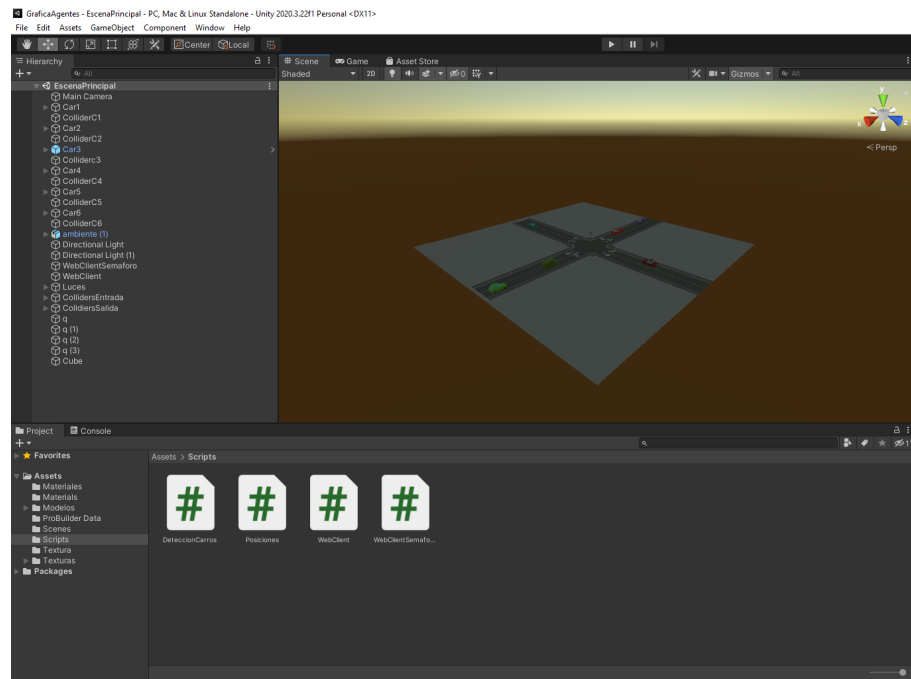
| Name | Date modified | Type | Size |
|---------------------|-------------------|--------------------|--------|
| Diagramas | 12/1/2021 1:16 PM | File folder | |
| GraficaAgentes | 12/1/2021 1:16 PM | File folder | |
| Sistema | 12/1/2021 1:16 PM | File folder | |
| SistemaMultiagentes | 12/1/2021 1:16 PM | File folder | |
| Tutorial | 12/1/2021 1:16 PM | File folder | |
| .gitattributes | 12/1/2021 1:16 PM | GITATTRIBUTES File | 1 KB |
| .gitignore | 12/1/2021 1:16 PM | GITIGNORE File | 1 KB |
| Equipo de trabajo | 12/1/2021 1:16 PM | Text Document | 4 KB |
| LICENSE | 12/1/2021 1:16 PM | File | 2 KB |
| Propuesta del reto | 12/1/2021 1:16 PM | Documento Adob... | 551 KB |
| README | 12/1/2021 1:16 PM | MD File | 1 KB |

2. *Escena de Unity e importación de assets.*

2.1. *Escena principal de Unity.*

Una vez extraído el archivo se deberá ingresar a la carpeta que genera, en ella se encontrará la carpeta GraficaAgentes. Se debe de ingresar a la siguiente ruta: GraficaAgentes > Assets > Scenes > EscenaPrincipal y abrirla con el editor de Unity.

Nota: Al abrir la escena por primera vez tardará unos minutos en abrir, debido a la carga de todos los archivos y assets en el mismo.

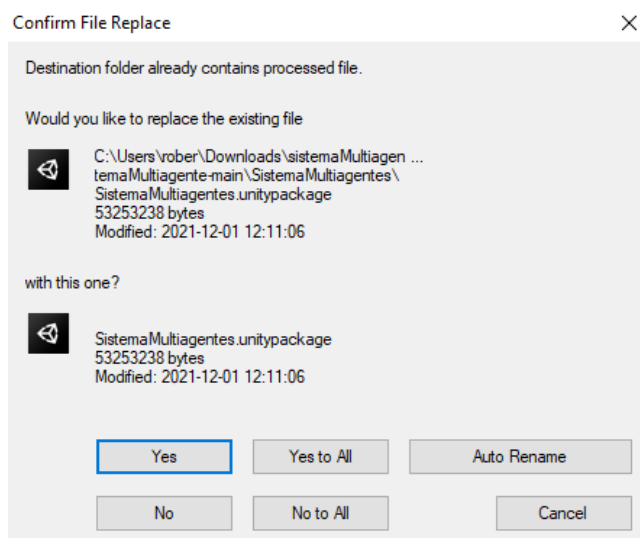


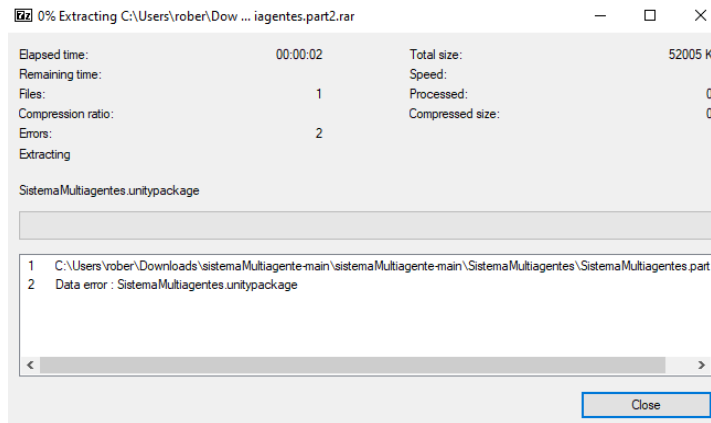
2.2. Extracción del Unity package.

Cuando se haya abierto la escena de Unity se tendrá que volver a la carpeta principal del sistema y acceder a SistemaMultiagentes. Esta carpeta contiene dos archivos RAR: SistemaMultiagentes.part1 y SistemaMultiagentes.part2, se deben de extraer ambos archivos en la misma carpeta con 7zip, WinRar o cualquier aplicación similar.

| Name | Date modified | Type | Size |
|-------------------------------|--------------------|--------------------|-----------|
| SistemaMultiagentes.part1.rar | 12/1/2021 1:45 PM | RAR File | 25,600 KB |
| SistemaMultiagentes.part2.rar | 12/1/2021 1:45 PM | RAR File | 10,205 KB |
| SistemaMultiagentes | 12/1/2021 12:11 PM | Unity package file | 0 KB |
| SistemaMultiagentes_1 | 12/1/2021 12:11 PM | Unity package file | 0 KB |

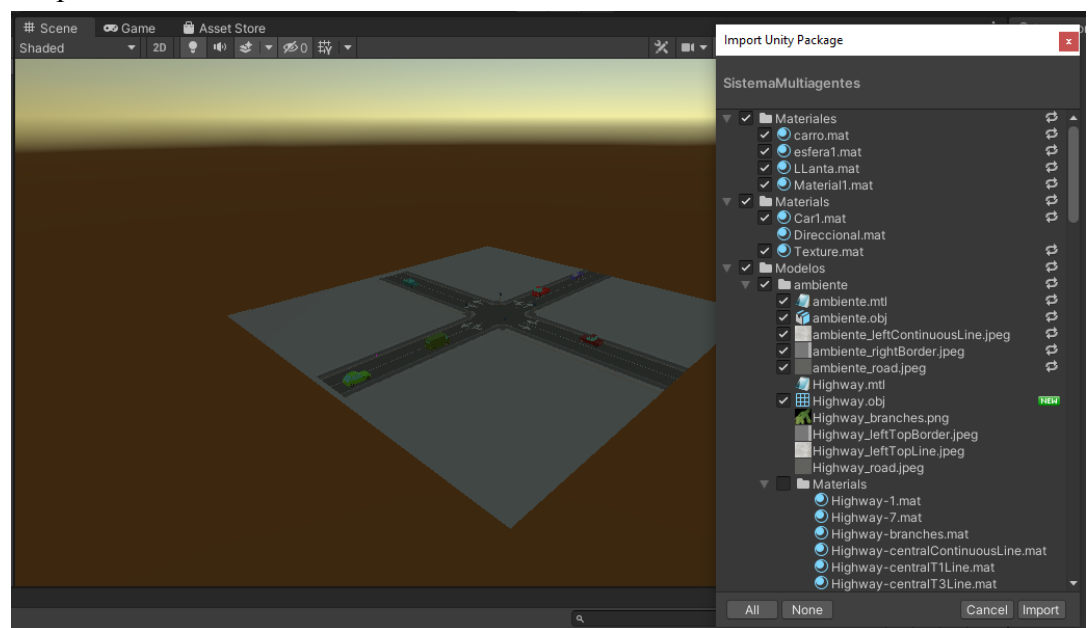
Cuando se extraiga el segundo archivo puede aparecer un error indicando que el archivo ya existe, para resolver esto, se tiene que dar click en el botón “Renombrar archivo automáticamente”.



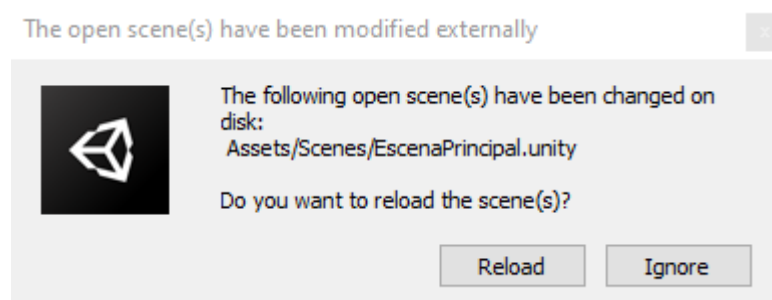


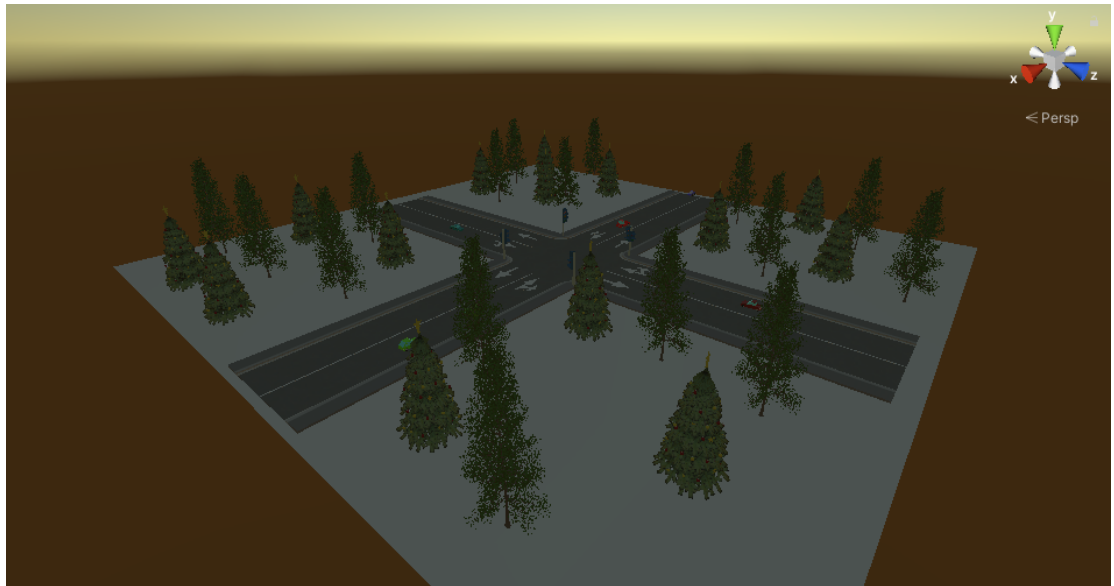
2.3. *Importación de assets.*

Posteriormente a haber extraído el archivo Unity Package, se tendrá que dar doble click para abrirlo y Unity lo abrirá automáticamente en la escena que ya se abrió previamente. Se abrirá una ventana en donde se mostrarán todos los assets y archivos que este contiene, y finalmente se debe dar click en “Import”.



Una vez importados los assets aparecerá una ventana que indica que se tiene que recargar el proyecto para aplicar los cambios, se le debe de dar click a “recargar” y aparecerá la escena actualizada con todos los cambios hasta la fecha.








3. Servidor en Python.

3.1. Archivo del servidor.

Una vez hechas las configuraciones de la escena en Unity se debe de volver a la carpeta principal y acceder a la carpeta Sistema, en esta se tendrá que abrir el archivo Server.py con el editor de código o IDE de preferencia.

| | | | | |
|---|---------------|-------------------|-------------|-------|
|  | config | 12/1/2021 1:45 PM | Python File | 1 KB |
|  | Server | 12/1/2021 1:45 PM | Python File | 12 KB |
|  | Traffic_Model | 12/1/2021 1:45 PM | Python File | 4 KB |

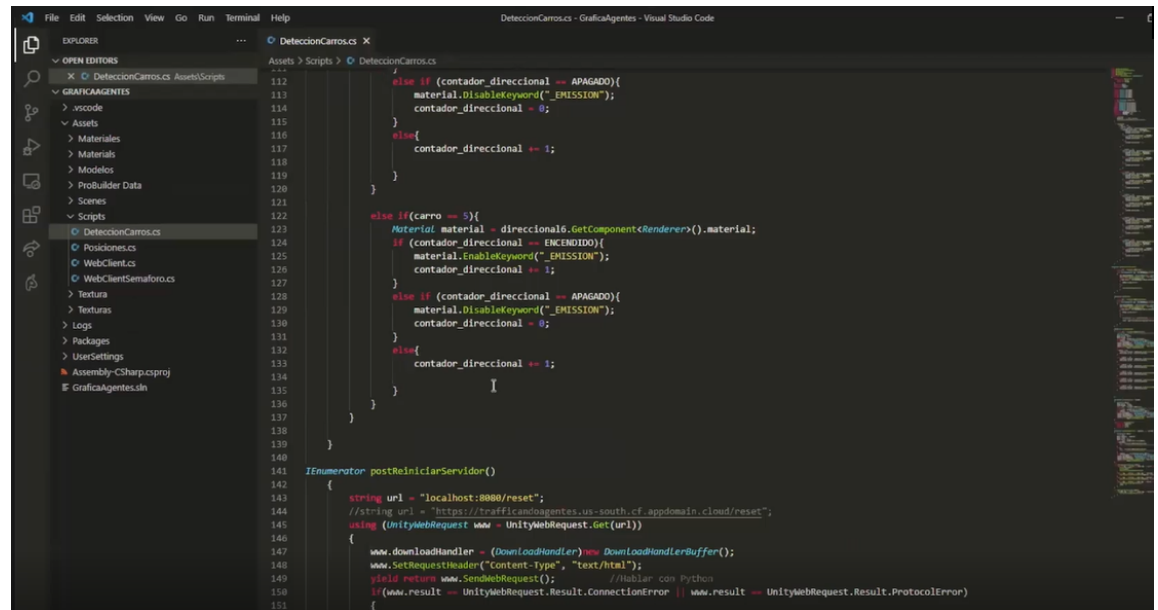
```

Sistema Traffic_Model.py
Project
  Sistema C:\Users\robert\Downloads\sistema\MultiAgentes
    config.py
    Server.py
    Traffic_Model.py
  External Libraries
  Scratches and Consoles
Traffic_Model.py
1 from mesa import Agent, Model
2 from mesa.time import RandomActivation
3 import random
4 from random import randint
5 import config
6
7 class CarAgent(Agent):
8     def __init__(self, unique_id, model, direccion):
9         super().__init__(unique_id, model)
10        random.seed()
11        self.estado = 2 #Detenido 0, Disminuyendo 1, Avanzando 2
12        self.cruzando = 0 #Fuera del cruce 0, En el cruce 1, Cruzado 3
13        self.pos = [0, 0, 0]
14        self.direccion = direccion #x 0, -x 1, z 2, -z 3
15        self.forzapalto = 0
16        vuelta = randint(1, 101)
17        if vuelta <= config.PROBABILIDAD_DE_DAR_VUELTA:
18            self.vuelta = 1
19        else:
20            self.vuelta = 0
21
22        # Establece el multiplicador de velocidad
23        if config.VELOCIDAD_VARIABLE:
24            self.velocidad = random.uniform(config.VELOCIDAD_MIN, config.VELOCIDAD_MAX)
25        else:
26            self.velocidad = 1
27
28        def darVuelta(self):
29            self.pos = [0, 0, 0]
30            if self.direccion == 0:
31                self.direccion = 3
32
33            elif self.direccion == 1:
34                self.direccion = 2
35
36            elif self.direccion == 2:
37                self.direccion = 0
38

```

3.2. *Correr el servidor localmente.*

Para correr el servidor de manera local, se deberá ejecutar el archivo Server.py, sin realizar modificaciones.



The screenshot shows the Visual Studio Code editor with the file 'DetectionCarros.cs' open. The left sidebar shows the project structure with folders like 'Assets', 'Scripts', and 'Assets/Scripts'. The main editor area displays the following C# code:

```
112 else if (contador_direccional == APAGADO){
113     material.DisableKeyword("_EMISSION");
114     contador_direccional = 0;
115 }
116 else{
117     contador_direccional += 1;
118 }
119 }
120 }
121 }
122 }
123 }
124 }
125 }
126 }
127 }
128 }
129 }
130 }
131 }
132 }
133 }
134 }
135 }
136 }
137 }
138 }
139 }
140 }
141 IEnumerator postReiniciarServidor()
142 {
143     string url = "localhost:8080/reset";
144     //string url = "https://trafficandagentes.us-south.cf.apptomain.cloud/reset";
145     using (UnityWebRequest www = UnityWebRequest.Get(url))
146     {
147         www.downloadHandler = (DownloadHandler)new DownloadHandlerBuffer();
148         www.SetRequestHeader("Content-Type", "text/html");
149         yield return www.SendWebRequest(); //Hablar con Python
150         if(www.result == UnityWebRequest.Result.ConnectionError || www.result == UnityWebRequest.Result.ProtocolError)
151         {
```

3.3. *Probar la simulación.*

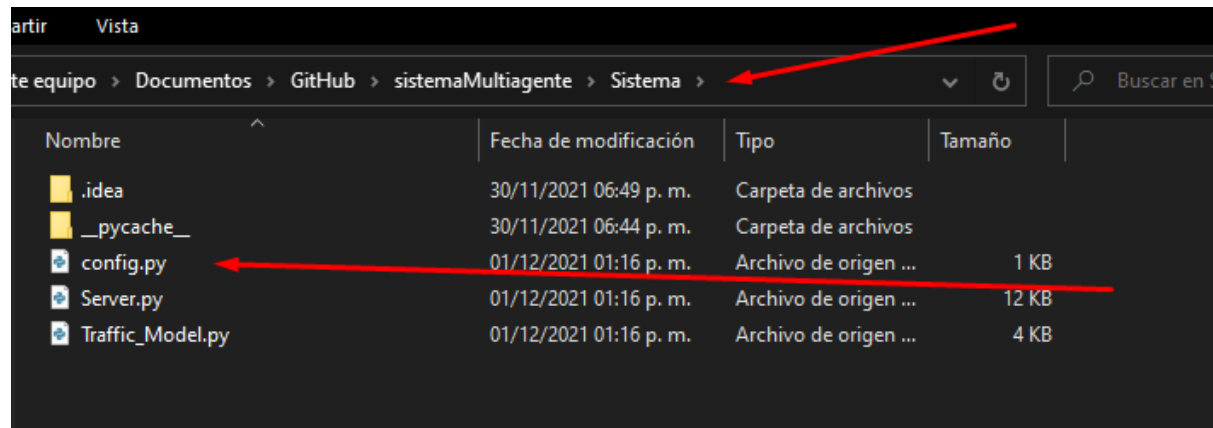
Una vez que el servidor esté corriendo se deberá regresar a la escena de Unity y seleccionar el botón de “Play” para observar la simulación. Esta se puede pausar y reiniciar cuantas veces se desee. Cada que se le hace click al botón de “Play” se crea una simulación nueva y diferente con parámetros aleatorios basados en las configuraciones del sistema.



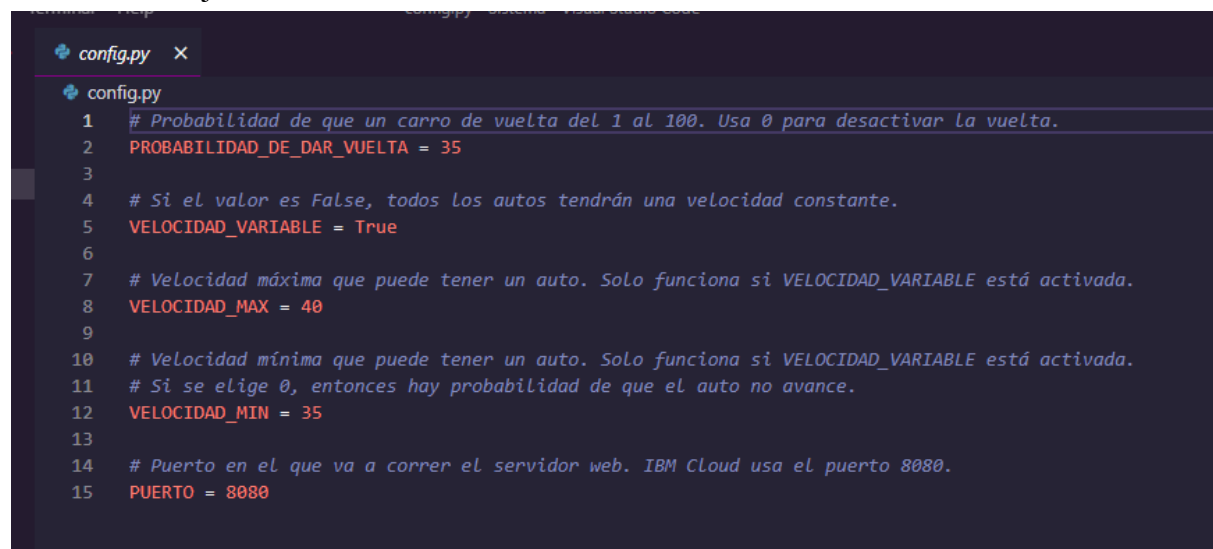
4. Configuraciones y consideraciones del sistema.

4.1. Archivo de configuración.

Es posible modificar ciertos parámetros de la simulación, para hacerlo se debe acceder a la carpeta de Sistema, misma en la que se encuentra el archivo Server.py y se deberá abrir el programa config.py con el editor de código o IDE de preferencia.



En este archivo se encontrarán distintos parámetros de la simulación que se pueden configurar, cada uno viene con sus posibles valores y su descripción, el modificar estos parámetros sugiere un cambio en la manera en la que la simulación se ejecuta.



Cada que se modifique el archivo config.py se deberá reiniciar el servidor a través de la terminal desde donde se ejecutó, presionando CTRL + C o Command + C en Mac se detendrá la ejecución del servidor y se deberá ejecutar nuevamente el archivo Server.py para registrar los cambios realizados.

4.2. Correr el sistema en la IBM Cloud.

El sistema se puede ejecutar de manera remota a través de los servicios de la nube de IBM, para hacerlo es necesario acceder a los scripts del proyecto de Unity a través del editor o bien accediendo desde la carpeta principal a la ruta:

GraficasAgentes > Assets > Scripts y se deberán abrir los cuatro scripts dentro de esta carpeta.

Cada que se encuentren líneas de código similar a las siguientes:

```
string url = "localhost:8080/carro";  
//string url = "https://trafficanandoagentes.us-south.cf.apptdomain.cloud/carro";
```

Se deberá comentar con // la primera línea y descomentar la segunda quitando los // al inicio. Esto se deberá de hacer para cada línea de código similar para los cuatro scripts que se tengan.

Es importante destacar que:

- No se podrán modificar los parámetros de la simulación desde el archivo config.py debido a que el servidor se maneja de manera remota, por lo que la simulación observada se mantendrá bajo los parámetros definidos por el equipo desarrollador en el servidor remoto.
- Para evitar saturar al servidor con peticiones, por cada frame se deberán modificar las funciones Update de los scripts del proyecto de Unity para que se hagan peticiones cada cierto tiempo, es decir, se debe de modificar la condición del if dentro del void Update() a un valor recomendado mayor o igual a 20 para que el sistema funcione correctamente. En caso de no realizar estas modificaciones, es muy probable que existan errores en la simulación y esta no se ejecute de la manera esperada.
- Debido a las modificaciones realizadas en el punto anterior, cuando la simulación se ejecuta en la nube de IBM, esta irá considerablemente más lento.