



Modelación de sistemas multiagentes y gráficas computacionales

Revisión de avance 2

Integrantes

Diego Enrique Jiménez Urgell	A01652617
Agustín Abreu Callejas	A01653126
Yusdivia Molina Román	A01653120

Profesores

Dr. Sergio Ruiz Loza
Dr. David Christopher Balderas Silva

Índice

1. Equipo de trabajo	3
1.1. Conformación del equipo	3
1.1.1 Diego Jiménez	3
1.1.1.1. Fortalezas	3
1.1.1.2. Áreas de oportunidad	3
1.1.1.3. Expectativas	3
1.1.2 Agustín Abreu	3
1.1.2.1. Fortalezas	3
1.1.2.2. Áreas de oportunidad	4
1.1.2.3. Expectativas	4
1.1.3 Yusdivia Molina	4
1.1.3.1. Fortalezas	4
1.1.3.2. Áreas de oportunidad	4
1.1.3.3. Expectativas	4
1.2. Expectativas como equipo y compromisos	4
2. Creación de herramientas de trabajo colaborativo	5
2.1. Repositorio de GitHub	5
3. Propuesta formal del reto	5
3.1. Descripción del reto a desarrollar	5
3.2. Identificación de los agentes involucrados	7
3.2.1. Diagrama de clase	8
3.2.2. Diagrama de protocolos de interacción	8
3.3. Plan de trabajo y aprendizaje adquirido	9
3.3.1. Plan de trabajo	9
3.3.2. Aprendizajes adquiridos	10
3.3.2.1 Gráficas computacionales	11
3.3.2.2 Sistemas multiagentes	11
3.3.3. Las actividades pendientes y el tiempo en el que se realizarán.	12
3.3.4. Actividades planeadas para la primera revisión	13
3.4. Evidencia de avance	13
3.4.1. Python	13
3.4.2. Unity	18
3.4.2.1 Mapa y diseño del ambiente	18
3.4.2.2. Implementación de movimiento del automóvil	20
3.4.2.3. Comportamiento del color de los semáforos	22
Anexos	24
Referencias	25

1. Equipo de trabajo

1.1. Conformación del equipo

Indicar los integrantes del equipo de trabajo. Además deben identificar las fortalezas y áreas de oportunidad de cada uno de ustedes. Así como las expectativas que tienen del bloque. Posteriormente, elaborar un breve listado de lo que esperan lograr y obtener como equipo de trabajo en el presente bloque, así como sus compromisos para lograrlo.

1.1.1 Diego Jiménez

1.1.1.1. Fortalezas

Mi principal fortaleza es la resiliencia, soy capaz de seguir adelante incluso cuando la situación no parece favorable. Esto está ligado con la dedicación, puedo llegar a ser muy obsesivo con las cosas que me interesan, y planeo focalizar parte de esa energía en el bloque. Considero que soy asertivo al comunicarme, y puedo expresarme de manera adecuada y amigable cuando no estoy de acuerdo con algo. Además, me tomo muy en serio mis compromisos cuando impactan a otras personas, como en el caso de un equipo de trabajo.

1.1.1.2. Áreas de oportunidad

Una de mis debilidades actuales es la distribución y organización de mi tiempo. Debido a otras actividades que tengo, en algunas ocasiones no me da suficiente tiempo para realizar mis tareas como me gusta. En muchos de estos casos sacrifico mis horas de sueño, lo que se va acumulando y me genera un agotamiento crónico para el final del bloque. Uno de mis objetivos en este bloque es disminuir esto. Otra área de oportunidad es que me distraigo cuando un tema no me gusta, y eso me atrasa después. Además puedo llegar a ser impaciente.

1.1.1.3. Expectativas

En este bloque sobre todo espero aprender mucho de sistemas multiagentes. Esto no solo es útil para videojuegos, sino para muchas aplicaciones del mundo real que requieren inteligencia artificial. Este es un campo en el que me he interesado desde antes, y me parece una buena oportunidad para explorarlo. En cuanto a gráficas computacionales, me parece interesante conocer más de ellas puesto que son un área crucial de las ciencias computacionales, aunque no me veo en este campo en un futuro.

1.1.2 Agustín Abreu

1.1.2.1. Fortalezas

Personalmente me considero una persona creativa y crítica que siempre busca realizar las actividades de la mejor forma posible. Por otra parte, también soy una persona bastante apasionada por los temas de su interés, y es esta idea la que me motiva cada día para seguir

haciendo un mejor trabajo. Finalmente, también soy alguien a quien le gusta tomar riesgos e intentar nuevas formas de hacer las cosas.

1.1.2.2. Áreas de oportunidad

En cuanto a las áreas de oportunidad que tengo como integrante del equipo es que a veces no organizo mi tiempo para entregar las actividades, y por tanto algunas ocasiones el trabajo no lo hago como se esperaba. Otro punto que debo de mejorar es que la mayoría de las veces no soy conciso explicando mis ideas, por lo que puede haber malentendidos en las juntas de equipo.

1.1.2.3. Expectativas

Durante este bloque espero aprender y aplicar nuevos temas de computación. Por otro lado, también espero que la resolución del reto sea una actividad que en algún momento pueda ocupar en mi vida profesional o incluso en la personal.

1.1.3 Yusdivia Molina

1.1.3.1. Fortalezas

Me gusta mucho seguir aprendiendo y siempre perfeccionar mis habilidades. Soy muy dedicada y me gusta formar opiniones e ideas con base en la evidencia y el análisis. No me gusta hacer trabajos a medias o con el mínimo esfuerzo. Soy ambiciosa y me gusta estar orientada a objetivos concretos. Siempre estoy abierta a nuevas ideas.

1.1.3.2. Áreas de oportunidad

Llego a ser impaciente, puedo llegar a ser muy crítica y no me gusta seguir a una persona, idea o regla ciegamente me gusta entender el porqué.

1.1.3.3. Expectativas

Determinar y definir si es que el área de inteligencia artificial y/o de gráficas es de mi preferencia para especializarme en algún futuro. Aprender y enriquecer mis habilidades como ingeniera en esta área.

1.2. Expectativas como equipo y compromisos

Esperamos tener un buen desempeño en todo el bloque. Esperamos desarrollar el reto final cumpliendo con las expectativas del curso y conforme lo aprendido en el curso. Para ello deberemos tener una muy buena comunicación y apoyo mutuo en todo momento. Cada integrante del equipo se compromete a trabajar de manera honesta, siempre dando lo mejor de sí. Comunicando si existiese algún percance, o una área de oportunidad en el conocimiento o producto a entregar. Asimismo, nos comprometemos a siempre entregar las actividades respetando los tiempos límites de entrega.

Dado que el desarrollo de una solución para el reto requiere tanto conocimientos técnicos como pensamiento creativo, en el equipo esperamos seguir un flujo de trabajo que favorezca ambos enfoques. Cuando se presente un problema, cada miembro deberá de pensar en posibles soluciones de manera individual, realizando la investigación conveniente. Posteriormente, tendremos una sesión de ideación en la que se compartirán las propuestas y se analizarán sus ventajas y desventajas. En este punto debemos tomar en cuenta la factibilidad técnica, así como definir un plan de implementación. Finalmente, se repartirán las labores de desarrollo para agilizar el proceso. Claramente, cada integrante se compromete a cumplir con sus asignaciones a su propio ritmo, pero con cierta fecha límite definida por nosotros mismos.

Por otro lado, nos comprometemos a estar constantemente revisando los canales de comunicación con el equipo y con los profesores. Además de que nos comprometemos a apoyarnos entre nosotros al revisar los entregables y escuchar de manera atenta y con respeto las ideas de los demás. Adicionalmente, esperamos tener un ambiente de trabajo ameno, con el objetivo de tener una mejor productividad y mayor compromiso con el proyecto y con el equipo. Todo lo anterior con la finalidad de tener una sinergia que nos permita seguir creciendo y aprendiendo tanto conocimientos técnicos como las habilidades relacionadas al trabajo en equipo. Anteriormente habíamos trabajado en equipo, por lo que este aspecto ya tiene cierto grado de avance.

2. Creación de herramientas de trabajo colaborativo

2.1. Repositorio de GitHub

<https://github.com/E1-CarpoolProject/CarpoolProject>

3. Propuesta formal del reto

3.1. Descripción del reto a desarrollar

La movilidad urbana habla acerca del desplazamiento que tienen las personas y la mercancía dentro de una ciudad. Este objetivo de llegar de un lado al otro se hace utilizando diferentes medios de transporte como el público y privado, de forma peatonal, entre muchas otras. Por otro lado, desde la segunda revolución industrial se consideraba al automóvil como un símbolo de progreso. Además que desde su invención supuso una revolución en el mundo del transporte, al ser más seguro y cómodo (Muñoz, 2015).

Actualmente, el uso indiscriminado de esta gran herramienta ha generado efectos negativos en diferentes sectores. Por ejemplo, nos ha traído problemas de contaminación al medio ambiente, horas perdidas en el tráfico e inclusive accidentes que se podrían fácilmente evitar. Estos problemas enlistados y demás comprometen el bienestar y la calidad de vida de las personas. Añadiendo que este panorama se complica año con año, ya que según el INEGI en

Méjico el número de automóviles en circulación aumentó en un 25.9% del 2015 al 2020 (2021).

Ante esta necesidad se deben plantear nuevas soluciones las cuales cuentan con planes de movilidad que permita responder ante las necesidades de la ciudadanía. Es decir, construir ciudades incluyentes y sustentables, donde se tenga como punto central el desarrollo del ser humano. Una parte importante de una ciudad incluyente es tener un plan de enfoque que reduzca la congestión vehicular. Ante ello, desarrollaremos una propuesta de solución, utilizando herramientas de gráficas computacionales y representando la salida de un sistema multiagentes.

En particular, se presentará un modelo de solución a un problema muy específico al que estamos expuestos de manera frecuente: el *carpooling*. Consiste en compartir el vehículo con otras personas de tal manera que la ruta del conductor no sea drásticamente modificada, y se pueda dejar a cada uno de los pasajeros en sus destinos durante el viaje. Esta práctica tiene un gran número de beneficios sociales y ambientales. Debido a que se utilizan menos vehículos, se reduce el consumo de combustibles y la generación de gases del efecto invernadero, lo que a su vez disminuye la contaminación del aire. Para el conductor y los pasajeros también hay beneficios que van desde una mejora de productividad y estado de ánimo, hasta una mejora notable en las finanzas personales puesto que el gasto de los trayectos se reparte equitativamente (Shaheen et al, 2018).

A pesar de que el *carpooling* es altamente atractivo, también existen muchas problemáticas que dificultan su implementación. La mayoría de las soluciones de *carpooling* están basadas en la tecnología, puesto que permite vincular fácilmente a los conductores y los pasajeros por medio de un dispositivo móvil. El sistema computacional debe de procesar una gran cantidad de información para determinar la repartición óptima de los pasajeros entre el conjunto de conductores. Sin embargo, existen varias maneras de evaluar la optimización, puesto que se podrían minimizar variables como el tiempo total estimado de recorrido, la distancia global recorrida, la cercanía de los vehículos con los pasajeros, etc. En este sentido, modelar el sistema utilizando multiagentes es favorable puesto que cada uno de ellos tiene un objetivo específico y un conjunto de reglas a seguir.

Adicionalmente, existe el problema de la infraestructura. Al ser tan grande la cantidad de datos a evaluar, una computadora normal sería subóptima para el procesamiento. Además, en una implementación de la vida real, debe de existir algún mecanismo centralizado de control con la capacidad de dar servicio concurrente y en tiempo real a todos los usuarios. Para esto se pueden utilizar plataformas en la nube, que virtualiza la infraestructura y permite una escalabilidad sencilla. Es por esto que la participación de IBM como socio formador es de gran valor para nuestra solución, pues nos permitirá implementar un sistema de manera similar a como se haría en el mundo real, utilizando tecnología de alto nivel.

3.2. Identificación de los agentes involucrados

Para la solución del carpooling se deben tomar en cuenta cuatro agentes principales:

- Automóvil (junto con conductor): Cada vehículo debe considerarse como un agente que parte de un punto inicial y desea llegar a un destino final. Su propósito es recoger a ciertos pasajeros en el camino y llevarlos a su propio destino. Sin embargo, le interesa realizarlo maximizando el número de pasajeros y minimizando el aumento en la distancia recorrida en comparación con la ruta original.
- Pasajeros (usuarios): Los pasajeros también tienen un punto inicial y un destino final. Desean que un automóvil pase a recogerlos, pero prefieren ser asignados al que esté más cercano para reducir su tiempo de espera. A ellos no les afecta la ruta final de automóvil, pero preferirían que su recorrido específico también sea lo más directo posible.
- Semáforos: Estos agentes se encargan de controlar el flujo de vehículos en los cruces. Para este caso particular no se espera optimizar su comportamiento. Sin embargo, deben de seguir ciertas reglas básicas de sincronización, por lo que es útil plantearlos como agentes que pueden comunicarse entre sí. Cada semáforo pertenece a un cruce.
- Ruteador: Este agente se encarga de dar información acerca de la topología a los agentes automóviles. Su labor es buscar la distancia más pequeña que existe entre dos puntos tomando en consideración la distribución de las calles, no la distancia absoluta. Cumple una función similar a lo que haría Google Maps o Waze. Para realizar los cálculos, se realizará una abstracción de la ciudad a una estructura de grafos, en la que habrá nodos de tipo cruce y de tipo pasajero. Los edges entre cada nodo tendrán una dirección y una longitud.

3.2.1. Diagrama de clase

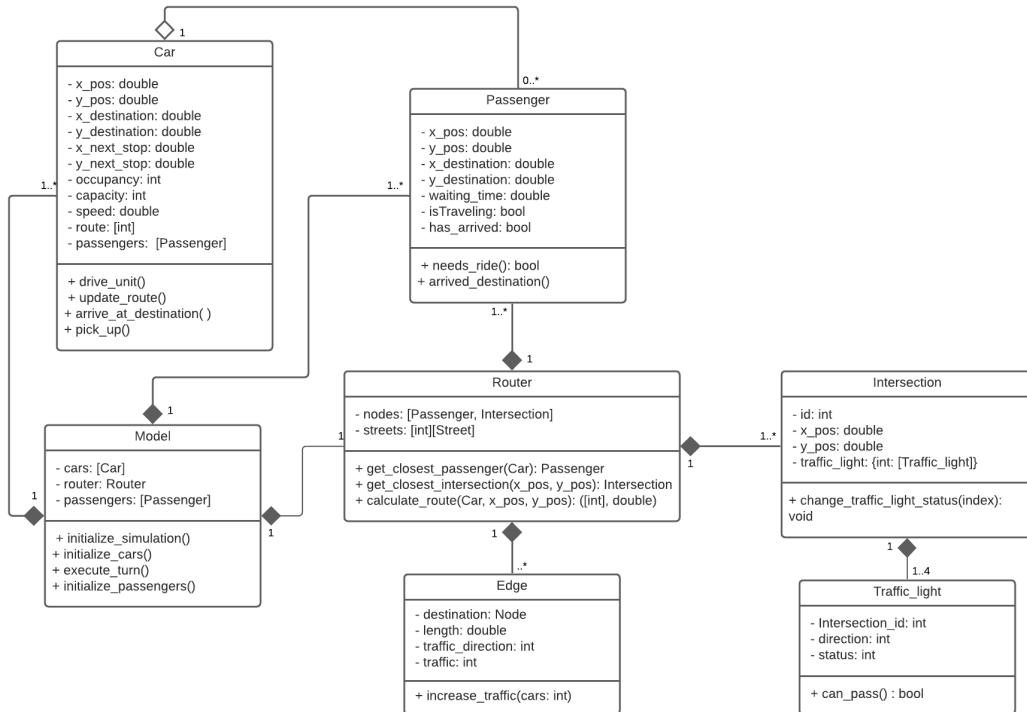


Imagen 1 Diagrama de clases

3.2.2. Diagrama de protocolos de interacción

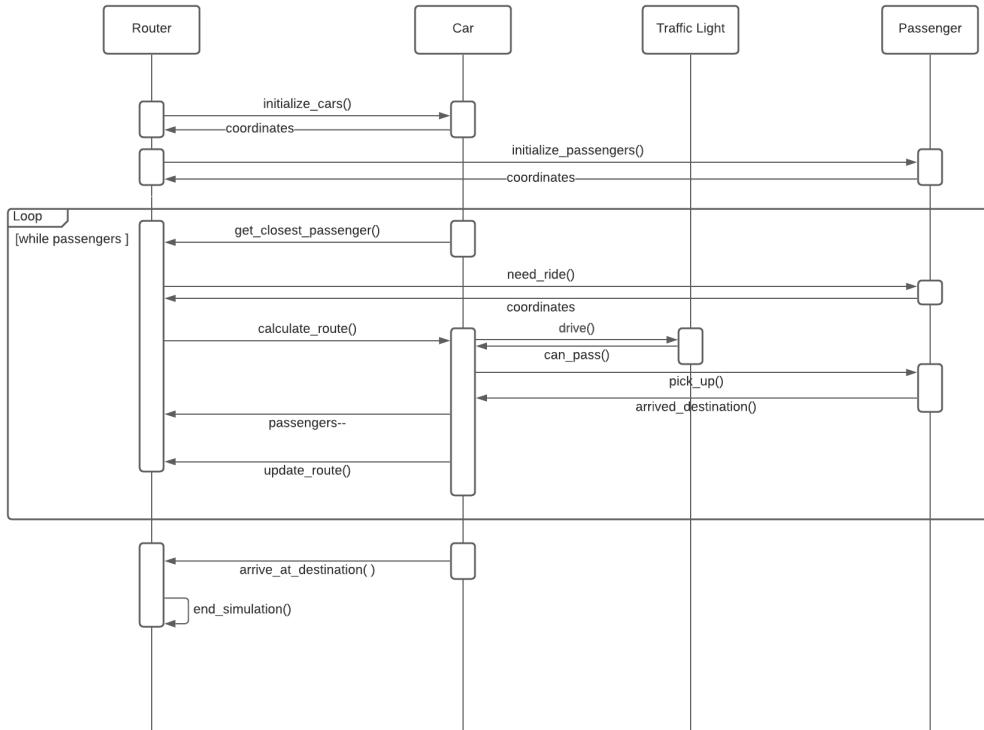


Imagen 2 Diagrama de secuencia

3.3. Plan de trabajo y aprendizaje adquirido

3.3.1. Plan de trabajo

Como una herramienta para tener una mejor visión acerca de lo que se solicita en cada entregable y en el reto en general, se realizó un plan de trabajo, el cual hasta el día viernes 12 de noviembre, queda de la siguiente forma:

PLAN DE TRABAJO EQUIPO 1

TAREA/ACTIVIDAD	FECHA INICIO	FECHA FIN	INICIA EN	DURACIÓN EN DÍAS	ESFUERZO ESTIMADO	RESPONSABLE	PORCENTAJE COMPLETO
Semana 2							
Etapa 1.1	11/8	11/12	0	4	5	Todos	95%
Conformación del equipo	11/9	11/10	1	1	1	Todos	100%
Descripción del reto a desarrollar	11/8	11/10	0	2	1	Yus	100%
Diagrama de clases	11/10	11/12	2	2	2	Agustín	100%
Diagrama de protocolos de interacción	11/10	11/12	2	2	3	Diego	100%
Plan de trabajo	11/8	11/9	0	1	1	Agustín y Yus	100%
Aprendizaje adquirido	11/11	11/12	3	1	1	Diego y Agustín	100%
Referencias y subir el documento	11/12	11/12	4	0	1	Yus y Diego	90%
Conseguir modelos 3D	11/8	11/12	0	4	2	Agustín	85%
Registro a IBM Cloud	11/10	11/11	2	1	1	Todos	100%
Configuración de repositorio GitHub	11/9	11/10	1	1	2	Agustín y Yus	100%
Presentación de avances	11/11	11/12	3	1	1	Todos	100%
Semana 3							
Etapa 1.2	11/15	11/19	7	4	21	Todos	85%
Diseño del sistema 3d (calles)	11/15	11/17	7	2	13	Agustín y Yus	90%
Simulación en Unity	11/16	11/18	8	4	13	Diego y Agustín	20%
Simulación de coches en línea recta	11/17	11/19	9	2	8	Todos	100%
Simulación coches dando la vuelta	11/17	11/19	9	2	8	Agustín	100%
Simulación de semáforos en funcionamiento	11/15	11/18	7	3	8	Yus y Diego	50%
Diseño de un coche por nosotros	11/17	11/19	9	2	13	Diego	100%
Entregable .py	11/17	11/19	9	2	8	Todos	40%
Definición clases con atributos y métodos	11/18	11/19	10	1	8	Yus	40%
Semana 4							
Etapa 2.1	11/22	11/26	14	4	34	Todos	0%
Negociación entre agentes	11/22	11/23	14	1	34	Todos	0%
Simulación de negociación	11/23	11/26	15	3	34	Todos	0%
Entregable en Unity	11/22	11/26	14	4	21	Todos	0%
Entregable .py	11/22	11/26	14	4	21	Todos	0%
Semana 5							
Etapa 2.2.	11/29	12/3	21	4	21	Todos	0%
Entregable en Unity	11/29	12/3	21	4	34	Todos	0%
Entregable .py	11/29	12/3	21	4	34	Todos	0%
Documentación final	11/30	12/3	22	3	13	Todos	0%
Presentación socio formador	12/2	12/3	24	1	3	Todos	0%

*Se usa la serie de Fibonacci
1,1,2,3,5,8,13,21,34

Imagen 3 Plan de trabajo

En este podemos observar que las actividades para la segunda semana del bloque están casi cubiertas en su totalidad. Dejando como pendiente subir este documento y encontrar más

modelos 3D que nos puedan ser de utilidad para la resolución del reto. Asimismo, las actividades de las semanas subsecuentes aún no se encuentran totalmente detalladas, esto lo estaremos realizando conforme desarrollemos el proyecto. Por otro lado, decidimos estimar el esfuerzo de cada actividad con la serie de Fibonacci; donde podemos ver que actividades como crear una cuenta está dentro de las más fáciles y las relacionadas al código se encuentran con una mayor complejidad ya que creemos será lo que más nos tomará tiempo de realizar.

Asimismo, realizamos un diagrama de Gantt con el objetivo de visualizar de mejor manera la ruta crítica del proyecto, y qué tareas son las que podemos paralelizar, para aprovechar de mejor manera los recursos con los que contamos (en este caso el tiempo). Este quedó de la siguiente manera:

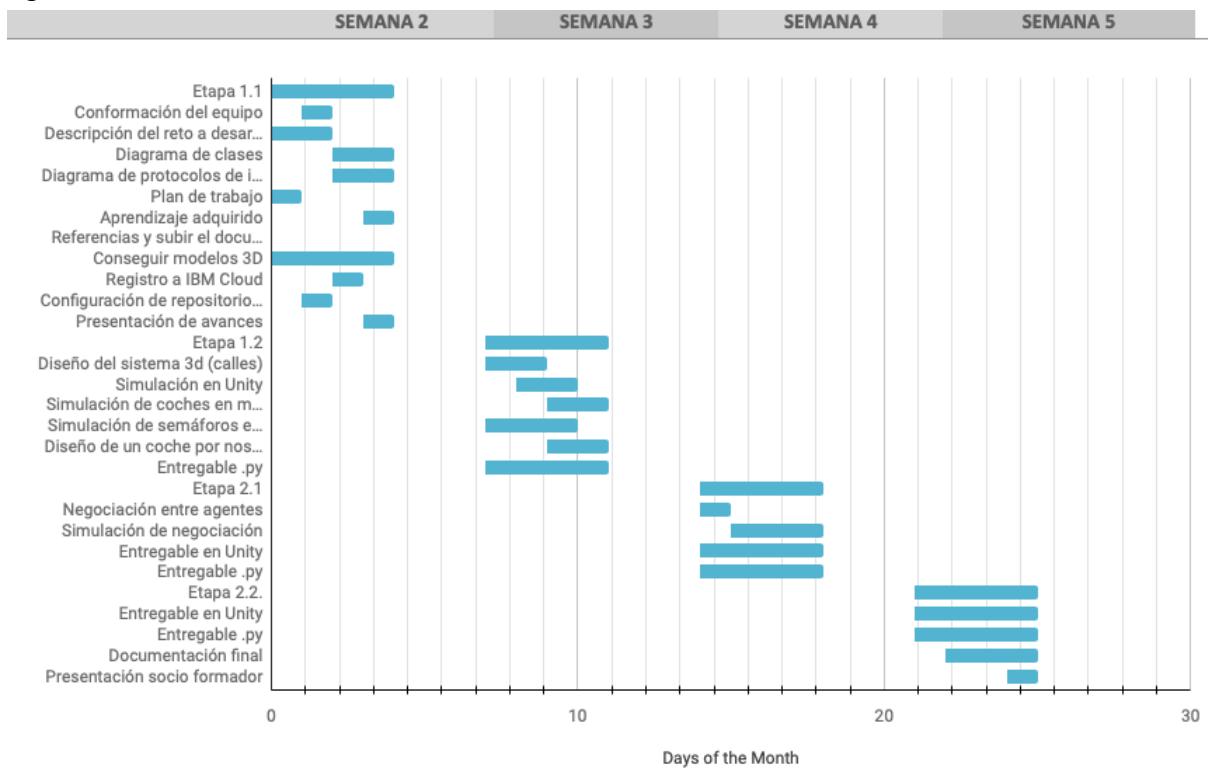


Imagen 4 Diagrama de Gantt

Para una mejor visualización del plan de trabajo se puede consultar la liga siguiente:
<https://docs.google.com/spreadsheets/d/1VUu2x0GdWkeFjR5Wqz8ogRXiVcVxM9z49T1Q9Nsx73w/edit?usp=sharing>

3.3.2. Aprendizajes adquiridos

Desde el inicio de este bloque, hemos tenido múltiples conocimientos en torno a dos ejes centrales, las gráficas computacionales y sistemas multiagentes; mismos que serán desarrollados a continuación.

3.3.2.1 Gráficas computacionales

Durante las semanas 1 y 2 se adquirieron conocimientos acerca de gráficas computacionales, desde cómo se crea una figura en tres dimensiones, hasta las nuevas tecnologías que ayudan a que un objeto se vea más real. Las principales herramientas que se han utilizado para aprender sobre estos temas son las matemáticas, que nos dan la teoría necesaria para entender el tema, y Unity, como motor gráfico para aplicar los conocimientos obtenidos durante las clases. Algunos de los temas vistos en clase son:

- Modelación en 3D
 - *Rendering Pipeline*
 - Geometría
 - Topología
 - GPU
 - Mallas poligonales
 - Archivos OBJ
- Transformación de figuras geométricas
 - Traslación
 - Escalabilidad
 - Rotación
- Iluminación
 - Iluminación difusa
 - Iluminación especular
 - Iluminación ambiental
- Texturas
 - UV *mapping*
 - Normal map

Durante la semana 3 se vieron los siguiente temas:

- Radar VFC -> Optimización
- Diagrama de arquitectura de la situación problema
 - Ejemplo con Unity y un servidor HTTP hecho por Python

3.3.2.2 Sistemas multiagentes

Ahora bien, como se mencionó anteriormente, la siguiente área de la computación que se ha estudiado, son los agentes computacionales. En el desarrollo del reto, este tema será sumamente importante para que la solución que se busca tenga la suficiente capacidad para que la interacción de los diversos agentes sea la esperada. Las principales herramientas que se han utilizado para este tema es Python, como ambiente de aplicación de los conocimientos obtenidos. Algunos de los temas vistos en clase son:

- Agentes computacionales
- Agentes inteligentes
- Arquitectura de agentes
 - De razonamiento deductivo

- De razonamiento práctico
- Modelación e implementación de agentes inteligentes
 - Modelado de agentes
 - Librerías de desarrollo
 - Mesa
- Comunicación entre agentes
 - Mecanismos de comunicación
 - Cooperación entre agentes
 -

Durante la semana 3 se vieron los siguiente temas:

- Optimización
 - Búsqueda exhaustiva (TSP)
 - Búsqueda aleatoria
- Hill climbing
- Gradiente descendiente y el método de Newton
- Algoritmos genéticos
 - Ciclo de reproducción
 - Mutación
 - Crossover
- Mesa
 - Grid
 - Estadística
- Optimización por enjambre de partículas

3.3.3. Las actividades pendientes y el tiempo en el que se realizarán.

Según el plan de trabajo, para esta segunda semana en curso tenemos como únicos dos pendientes, buscar y subir a Unity más modelos 3D que nos sean de utilidad para la resolución del reto (coches, semáforos, calles, etc.), actualmente ya tenemos unos pero consideramos es conveniente añadir. Esta actividad será realizada durante el día sábado por Agustín, quien es encargado de esta actividad. Por otro lado, se está pendiente de finalizar los últimos detalles del documento, para poder subirlo. Esto se hará en las siguientes horas.

Para esta tercera semana tenemos como pendiente seguir con la simulación en Unity, específicamente en la parte de los semáforos y los coches dando vuelta en las intersecciones. Los encargados son Diego y Agustín. Por la parte de python, aún debemos seguir codificando la lógica. Hay una parte faltante en la definición de las clases, la encargada es Yus. Del resto, todos somos responsables. Ambas actividades se realizarán el día sábado y en la tarde del viernes.

3.3.4. Actividades planeadas para la primera revisión

Se puede ver en nuestro plan de trabajo las actividades realizadas, los responsables, las fechas establecidas por nosotros mismos según las fechas de entrega del reto y un número de la serie de Fibonacci para estimar el esfuerzo que implica cada actividad.

3.4. Evidencia de avance

3.4.1. Python

Durante la semana 3 se pasó el diagrama de clases a código y se empezó a implementar métodos de la parte del ambiente, como el cambio de luces en el semáforo. A continuación se muestran unas capturas de pantalla de un poco lo que se realizó:

```
8  class CarAgent(Agent):
9      """ An agent with fixed initial wealth."""
10     def __init__(self, unique_id, model):
11         super().__init__(unique_id, model)
12         a = np.random.randint(len(positions))
13         while True:
14             b = np.random.randint(len(positions))
15             if a != b:
16                 break
17         self.x_pos = positions[a][0]
18         self.y_pos = positions[a][1]
19         self.y_destination = positions[b][1]
20         self.x_destination = positions[b][0]
21         self.x_next_stop = 0
22         self.y_next_stop = 0
23         self.occupancy = 1
24         self.capacity = 5
25         # REVISAR
26         self.passengers = []
27
28     def step(self):
29         # The agent's step will go here.
30         print ("Car " + str(self.unique_id) +".")
31
32     def drive_unit(self):
33         return True
34
35     def update_route(self):
36         return True
37
38     def arrive_at_destination(self):
39         return self.x_pos == self.x_destination and self.y_pos == self.y_destination
40
41     def pick_up(self):
42         if self.occupancy < self.capacity:
43             self.occupancy += 1
44             return True
45         else:
46             print("El coche está lleno :(")
47             return False
48
```

Imagen 5 Clase coche

```

2 class Edge:
3
4     def __init__(self, length, direction):
5         self.length = length
6         self.direction = direction
7         self.traffic = 0
8
9     def add_car(self):
10        self.traffic += 1

```

Imagen 6 Clase arista

```

1 ENVIRONMENT = {
2     "intersections_data": [
3         {
4             "x": 0,
5             "y": 0,
6             "directions": ["UP", "DOWN", "LEFT", "RIGHT"]
7         },
8         {
9             "x": 0,
10            "y": -1,
11            "directions": ["UP"]
12        },
13        {
14            "x": 0,
15            "y": 1,
16            "directions": ["DOWN"]
17        },
18        {
19            "x": 1,
20            "y": 0,
21            "directions": ["LEFT"]
22        },
23        {
24            "x": -1,
25            "y": 0,
26            "directions": ["RIGHT"]
27        }
28    ],
29    "roads_data": [
30        {
31            "source": 0,
32            "destination": 1,
33            "direction": "DOWN"
34        },
35        {
36            "source": 0,
37            "destination": 2,
38            "direction": "UP"
39        },
40        {
41            "source": 0,
42            "destination": 3,
43            "direction": "RIGHT"

```

Imagen 7 Simulación del entorno

```

36     def prepare_for_light_change(self):
37         self.next_light = self.active_light + 1
38         self.next_light %= len(self.traffic_lights)
39         self.traffic_lights[self.active_light].warn_change()
40
41     def change_traffic_light_status(self):
42         self.traffic_lights[self.active_light].toggle()
43         self.traffic_lights[self.next_light].toggle()
44         self.active_light = self.next_light
45
46     def points_turn(self, direction):
47         if direction == 1:
48             self.x_prev = self.x_pos + 1
49             self.y_prev = self.y_pos - 1
50         elif direction == 2:
51             self.x_prev = self.x_pos - 1
52             self.y_prev = self.y_pos + 1
53         elif direction == 3:
54             self.x_prev = self.x_pos - 1
55             self.y_prev = self.y_pos - 1
56         elif direction == 4:
57             self.x_prev = self.x_pos + 1
58             self.y_prev = self.y_pos + 1
59
60     def step(self):
61         self.ticks_to_light_change -= 1
62
63         if self.ticks_to_light_change == 1:
64             self.prepare_for_light_change()
65
66         elif self.ticks_to_light_change == 0:
67             self.change_traffic_light_status()
68             self.ticks_to_light_change = TICKS_TO_CHANGE
69
70         status = f"""
71             Intersection
72             x: {self.x_mid}, y: {self.y_mid}
73             Traffic lights status:""""
74         status += "\n"
75         for light in self.traffic_lights:
76             status += "\t\t" + str(light)
77         status += "\n"
78         print(status)
79

```

Imagen 8 Clase intersección

```

1  from car import CarModel
2  from passenger import PassengerModel
3  from traffic_light import TrafficLight
4  from router import RouterModel
5  from environment import ENVIRONMENT
6
7
8  if __name__ == "__main__":
9
10         # Inicializar el ambiente
11         router = RouterModel(ENVIRONMENT)
12         for i in range(10):
13             print(f"NEW TICK {i}")
14             router.step()
15
16
17         # inicializa 5 pasajeros
18         passenger_model = PassengerModel(5)
19         passenger_model.step()
20
21         # inicializa 5 coches
22         car_model = CarModel(5)
23         car_model.step()
24

```

Imagen 9 Main

```

1  from mesa import Agent, Model
2  from mesa.time import RandomActivation
3  import numpy as np
4
5  positions = [[1, 1], [1, 2], [1, 3], [2, 1], [2, 2], [2, 3]]
6
7
8  class PassengerAgent(Agent):
9      def __init__(self, unique_id, model):
10          super().__init__(unique_id, model)
11          #posiciones de origen y destino random
12          a = np.random.randint(len(positions))
13          while True:
14              b = np.random.randint(len(positions))
15              if a != b:
16                  break
17          self.x_pos = positions[a][0]
18          self.y_pos = positions[a][1]
19          self.y_destination = positions[b][1]
20          self.x_destination = positions[b][0]
21          self.is_traveling = False
22          self.has_arrived = False
23
24      def imprimirPos(self):
25          if self.is_traveling:
26              print("estoy viajando")
27          elif self.has_arrived:
28              print("ya llegue")
29          else:
30              print("estoy esperando ride")
31      def needs_ride(self):
32          return not self.is_traveling or not self.has_arrived
33
34      def arrived_destination(self):
35          return self.has_arrived
36
37      def step(self):
38          # The agent's step will go here.
39          print ("pasajero número " + str(self.unique_id))
40          print( "Mi posicion de origen es: "+str(self.x_pos)+" , "+ str(self.y_pos)+ " . ")
41          print( "Mi posicion de destino es: "+str(self.x_destination)+" , "+ str(self.y_destination)+ " . ")
42          self.imprimirPos()

```

Imagen 10 Clase Pasajero

```
1  from edge import Edge
2  from intersection import IntersectionModel
3
4  from mesa import Agent, Model
5  from mesa.time import RandomActivation, BaseScheduler
6
7
8  class RouterAgent(Agent):
9
10     def __init__(self, unique_id: int, model: Model, environment_data):
11         super().__init__(unique_id, model)
12
13         intersections_data = environment_data["intersections_data"]
14         model = IntersectionModel(intersections_data)
15         self.nodes = model.schedule.agents
16         self.intersection_model = model
17
18         self.edges = []
19         for _ in range(len(self.nodes)):
20             self.edges.append([0] * len(self.nodes))
21
22         roads_data = environment_data["roads_data"]
23         for road in roads_data:
24             edge = Edge(length=1, direction=road["direction"])
25             source, destination = road["source"], road["destination"]
26             self.edges[source][destination] = edge
27
28         print(self.edges)
29
30     def step(self):
31         self.intersection_model.step()
32
33
34  class RouterModel(Model):
35      """A model with some number of agents."""
36
37      def __init__(self, environmental_data):
38          super().__init__()
39          self.schedule = BaseScheduler(self)
40          a = RouterAgent(unique_id=0, model=self, environment_data=environmental_data)
41          self.schedule.add(a)
```

Imagen 11 Clase central de control

```

1  from enums import LightStatus
2
3
4  class TrafficLight:
5
6      def __init__(self, tid, direction):
7          self.tid = tid
8          self.direction = direction
9          self.status = LightStatus.RED.value
10
11     def can_pass(self):
12         return self.status == LightStatus.GREEN.value
13
14     def warn_change(self):
15         self.status = LightStatus.YELLOW.value
16
17     def toggle(self):
18         if self.status == LightStatus.YELLOW.value:
19             self.status = LightStatus.RED.value
20
21         elif self.status == LightStatus.RED.value:
22             self.status = LightStatus.GREEN.value
23
24     def __str__(self):
25         return f"Direction: {self.direction}, Status: {self.status}\n"

```

Imagen 12 Clase semáforo

3.4.2. Unity

3.4.2.1 Mapa y diseño del ambiente

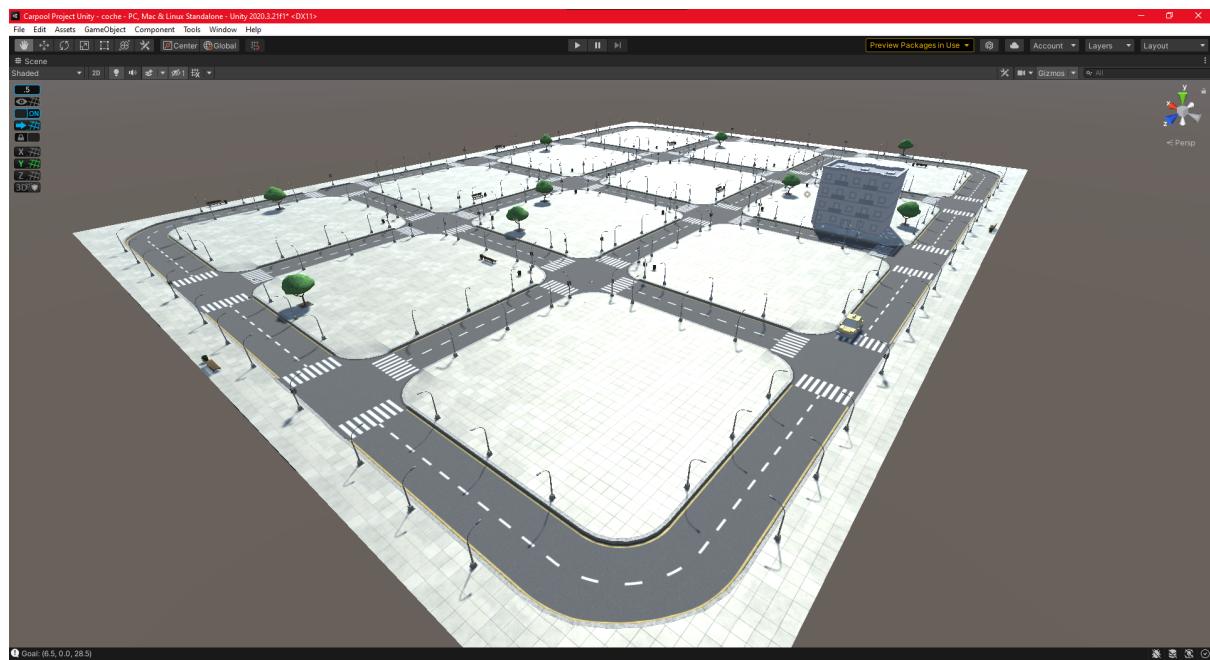


Imagen 13 Calle vista global

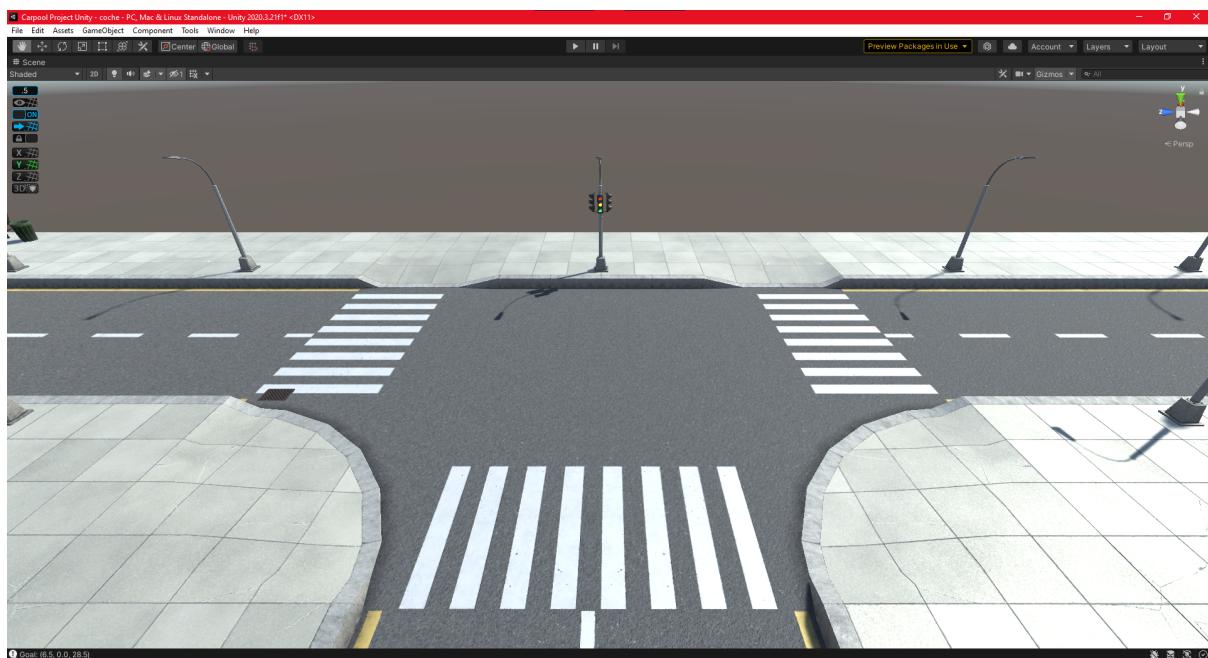


Imagen 14 Cruce con semáforo

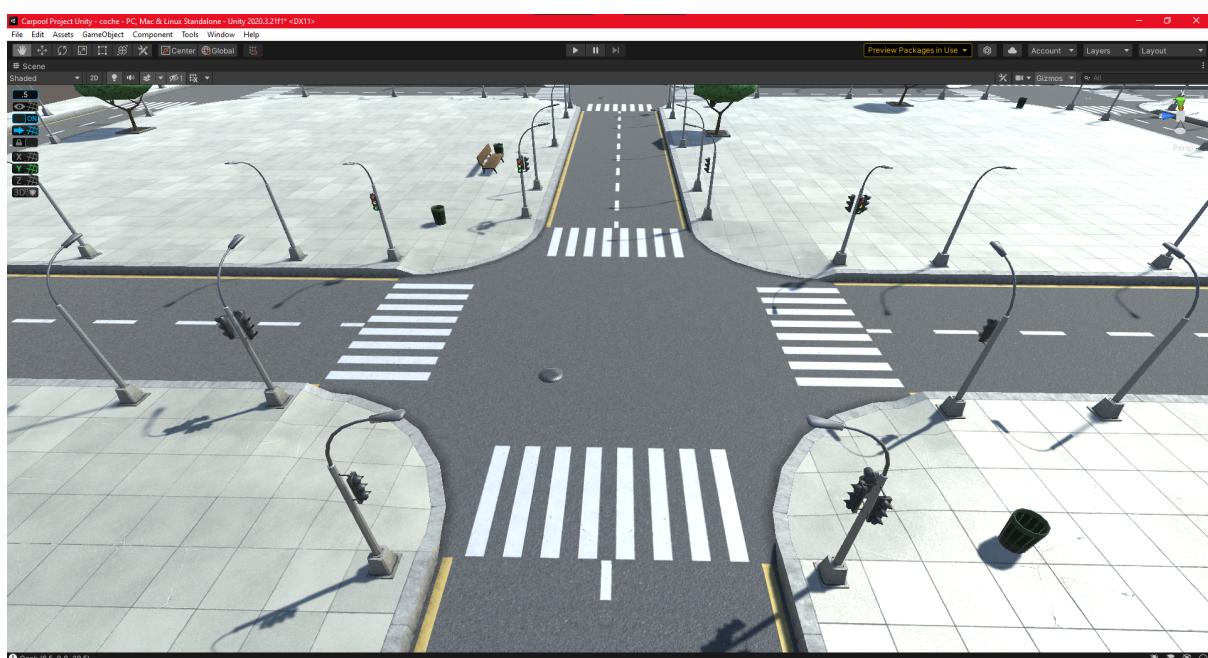


Imagen 15 Cruce de 4 calles

3.4.2.2. Implementación de movimiento del automóvil



Imagen 16 Punto de inicio de la simulación



Imagen 17 Taxi dando la vuelta



Imagen 18 Taxi terminando la vuelta

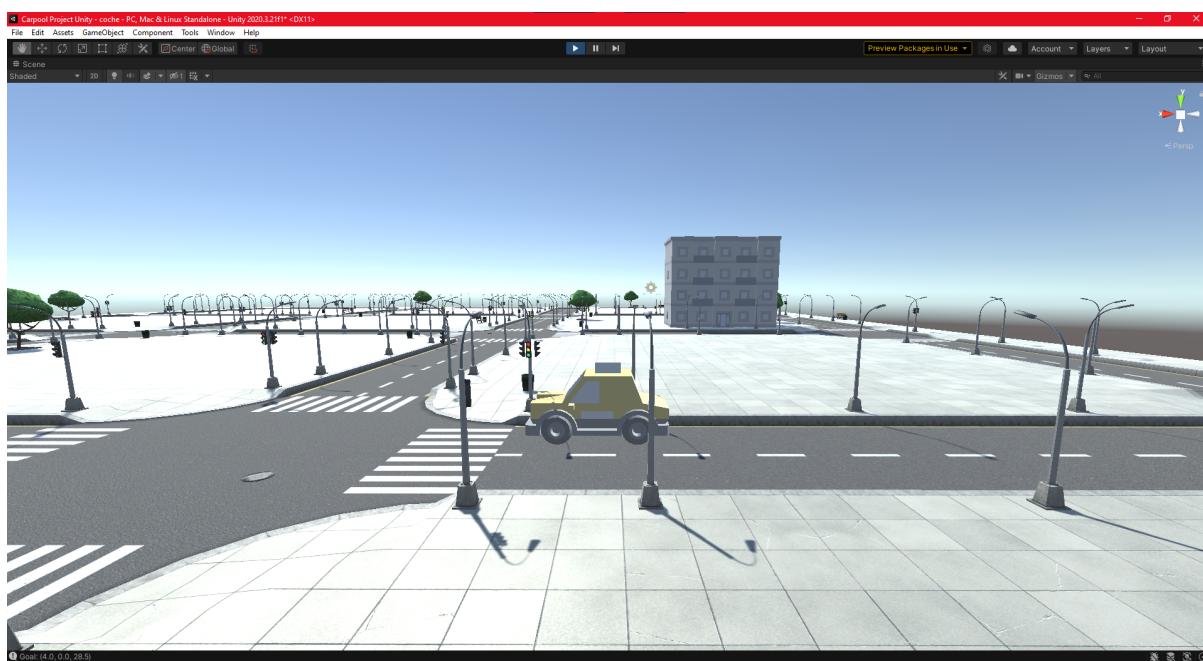


Imagen 19 Punto final de la simulación

3.4.2.3. Comportamiento del color de los semáforos

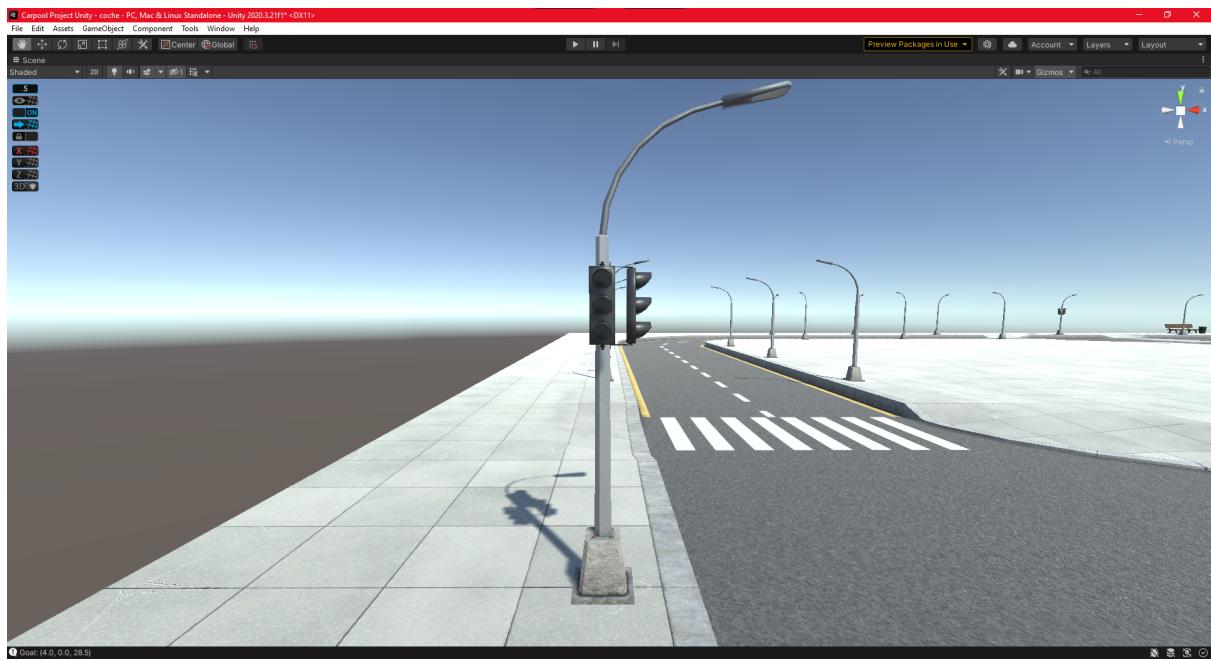


Imagen 20 Semáforo apagado



Imagen 21 Semáforo en rojo



Imagen 22 Semáforo en amarillo



Imagen 23 Semáforo en verde

Anexos

Repositorio:

<https://github.com/E1-CarpoolProject/CarpoolProject>

Presentación del avance semanal:

[https://docs.google.com/presentation/d/1xj0UgzZQuRGirIorjEJaYFqb8rY24EFNaiyFS2EVu
MY/edit?usp=sharing](https://docs.google.com/presentation/d/1xj0UgzZQuRGirIorjEJaYFqb8rY24EFNaiyFS2EVuMY/edit?usp=sharing)

Plan de trabajo:

[https://docs.google.com/spreadsheets/d/1VUUu2x0GdWkeFjR5Wqz8ogRXiVcVxM9z49T1Q9
Nsx73w/edit?usp=sharing](https://docs.google.com/spreadsheets/d/1VUUu2x0GdWkeFjR5Wqz8ogRXiVcVxM9z49T1Q9Nsx73w/edit?usp=sharing)

Referencias

EsmartCity. (2021). Movilidad Urbana. ESMARTCITY. Recuperado el 9 de noviembre de 2021, de <https://www.esmartcity.es/movilidad-urbana>

Instituto Nacional de Estadística y Geografía (INEGI). (2021). Parque vehicular.

Inegi.org.mx. Recuperado el 9 de noviembre de 2021, de
<https://www.inegi.org.mx/temas/vehiculos/>

Muñoz, G. (2015). El automóvil y el progreso moral. Información. Recuperado el 9 de noviembre de 2021, de

<https://www.informacion.es/opinion/2015/06/15/automovil-progreso-moral-6393485.html>

Shaheen, S., Cohen, A., & Bayen, A. (2018). The Benefits of Carpooling. *UC Berkeley: Transportation Sustainability Research Center*. <http://dx.doi.org/10.7922/G2DZ06GF>

Recuperado de <https://escholarship.org/uc/item/7jx6z631>