Tecnológico de Monterrey

Modelación de Sistemas Multiagentes con Gráficas Computacionales TC2008B

Actividad: Reto Movilidad Urbana

Alumno:
Daniel Soult Gómez - A01782985
Ángel Adrián Afonso Castellanos - A01782545

Profesor:
Octavio Navarro
Gilberto Echeverría

Campus Santa Fe

30 de noviembre del 2023

Problemática

El problema que se busca resolver a lo largo de la situación problema es el de la movilidad urbana. Este es considerado como la habilidad de moverse de un sitio a otro, siendo este sumamente importante para el desarrollo de una población. La movilidad se ve muy relacionada con el uso de un algún tipo de transporte como lo es el carro, siendo esto un signo distintivo del progreso. Pero en los tiempos más actuales esto ya no es lo mismo gracias al crecimiento exponencial que hay de automóviles en México, fomentando políticas erróneamente asociadas con la movilidad sostenible teniendo efectos negativos en el país tanto económicos, sociales y ambientales.

El incremento del uso del automóvil ha sido extremadamente alto, pasando de 106 millones VKT en 1990, a 339 millones en el año 2010. Teniendo graves afectaciones en distintas áreas como lo son lo social y lo ambiental, como, el smog, accidentes, enfermedades y congestiones vehiculares. Y para que nuestro país se pueda colocar dentro de las mejores economías globales es necesario que la movilidad de sus ciudades mejore, especialmente en sus ciudades más pobladas las cuales son donde se acumulan un número mayor de vehículos.

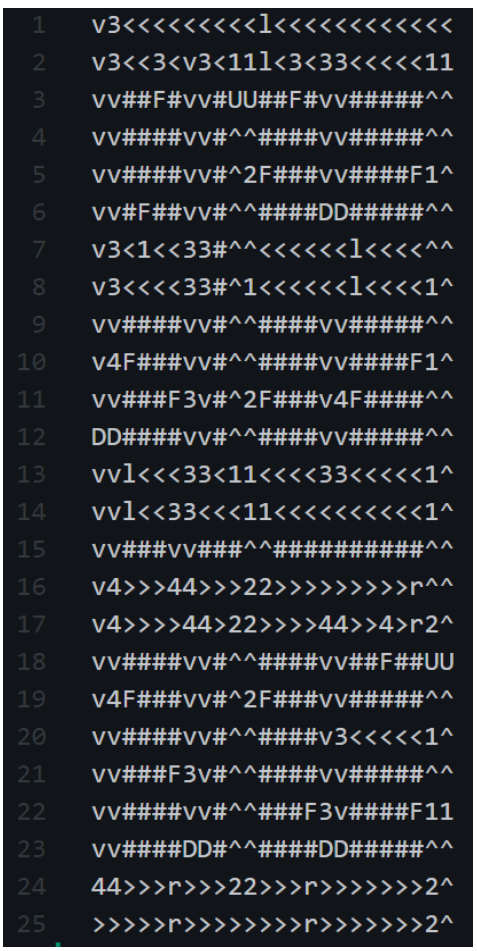
Es aquí donde se presenta la problemática que se plantea resolver, donde se propone una solución para mejorar la movilidad urbana y que se reduzca la congestión vehicular al simular de manera gráfica el tráfico con distintos puntos de salida en un sistema multi agentes.

Propuesta de solución:

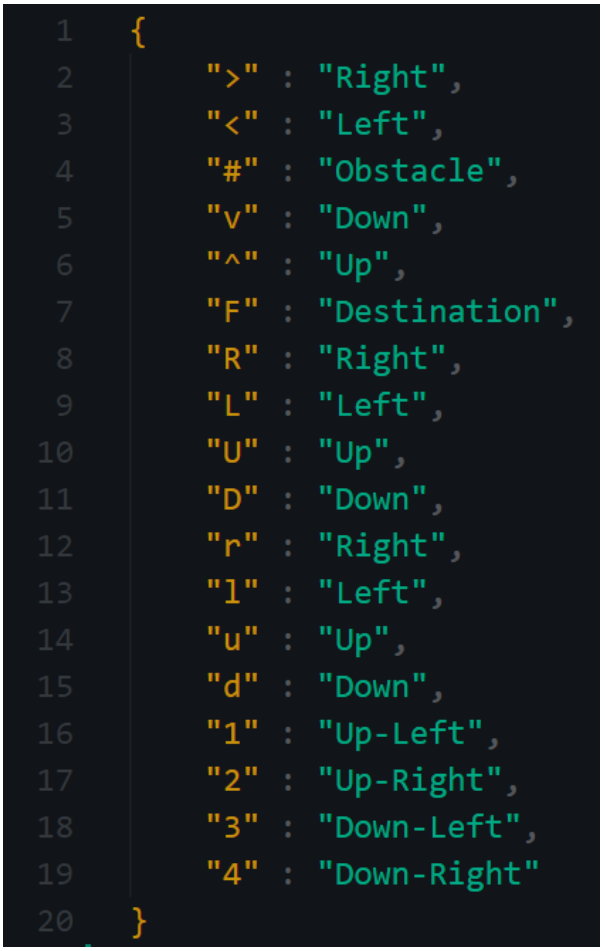
Para atender esta problemática es necesaria la creación de un ambiente donde se pueda simular esta situación, por lo que se creó un modelo en python con la utilización de la librería Mesa, la cual permite la simulación de modelos basados en agentes, que son entidades autónomas interactuando entre ellos y con el entorno. Los agentes para esta problemática fueron de 5 tipos: Carro, Semáforo, Destino, Obstáculo y Calle. Cada uno de estos tipos de agentes cuentan con comportamientos diferentes, por ejemplo, el agente Carro será el único agente que responderá y tomará decisiones basadas en la información que reciba por parte del entorno, el cual estará conformado por el resto de los agentes cuyos valores y estados serán la información a transmitir al agente principal.

Ahora se nos presenta la siguiente interrogante, ¿Cómo vamos a tener una ciudad sin tráfico?, y de acuerdo a lo aprendido en clase, básicamente tendríamos que evitar la aglomeración de coches en semáforos, los choques, o cualquier otro tipo de circunstancia, calculando la ruta óptima para que todos puedan llegar a su destino, considerando como ruta óptima la ruta más corta, o la ruta con el menor tiempo de espera posible. Esto es posible mediante la implementación de algoritmos avanzados de búsqueda como A Estrella o Dijkstra, pero eso sería en el caso de que todos los conductores fueran reales, personas apresuradas. Teniendo un caso utópico como lo es el mundo sin tráfico, lleno de puros conductores ejemplares que respetan los semáforos, que no chocan y que no tienen la horrorosa experiencia de estar 3 horas sentadas en su coche esperando a que la fila vehicular avance, en este proyecto decidimos tomar también una situación utópica donde los pasajeros pueden tener un cambio de opinión, ¿qué tal que en el camino vio una cafetería y decidió detener su camino a casa y llegar a otro destino? Por eso no fue necesario el cálculo de una ruta ideal para el coche, se decidió darles autonomía absoluta y solo respetar las reglas de tránsito.

El modelo está basado en un mapa de caracteres que está asociado a un diccionario que define las direcciones o identidad de cada elemento, a continuación se presentan ambos archivos:



```
v3<<<<<<<<<l<<<<<<<<<<
v3<<3<v3<11l<3<33<<<<<11
vv##F#vv#UU##F#vv#####^^
vv###vv#^^###vv#####^^
vv###vv#^2F###vv####F1^
vv#F##vv#^^####DD#####^^
v3<1<<33#^^<<<<<<l<<<<^^
v3<<<<33#^1<<<<<<l<<<<1^
vv###vv#^^###vv#####^^
v4F###vv#^^###vv####F1^
vv###F3v#^2F###v4F####^^
DD###vv#^^###vv#####^^
vvl<<<33<11<<<<33<<<<<1^
vvl<<33<<<11<<<<<<<<<<1^
vv###vv###^^##########^^
v4>>>44>>>22>>>>>>>>>r^^
v4>>>>44>22>>>>44>>4>r2^
vv###vv#^^###vv##F##UU
v4F###vv#^2F###vv#####^^
vv###vv#^^###v3<<<<<1^
vv###F3v#^^###vv#####^^
vv###vv#^^###F3v####F11
vv####DD#^^####DD#####^^
44>>>r>>>22>>>r>>>>>>>2^
>>>>>r>>>>>>>>r>>>>>>>2^
```



```
{
    ">" : "Right",
    "<" : "Left",
    "#" : "Obstacle",
    "v" : "Down",
    "^" : "Up",
    "F" : "Destination",
    "R" : "Right",
    "L" : "Left",
    "U" : "Up",
    "D" : "Down",
    "r" : "Right",
    "l" : "Left",
    "u" : "Up",
    "d" : "Down",
    "1" : "Up-Left",
    "2" : "Up-Right",
    "3" : "Down-Left",
    "4" : "Down-Right"
}
```

Imágen 1: Archivo .txt con los caracteres que generan el mapa.
Imagen 2: Archivo .json que otorga un significado a los caracteres del mapa.

Con esta información, fue generado un modelo que lee el mapa, crea agentes en la posición indicada y les asigna las características necesarias para poder ser interactuables por parte del objeto, visualizandose de esta manera:

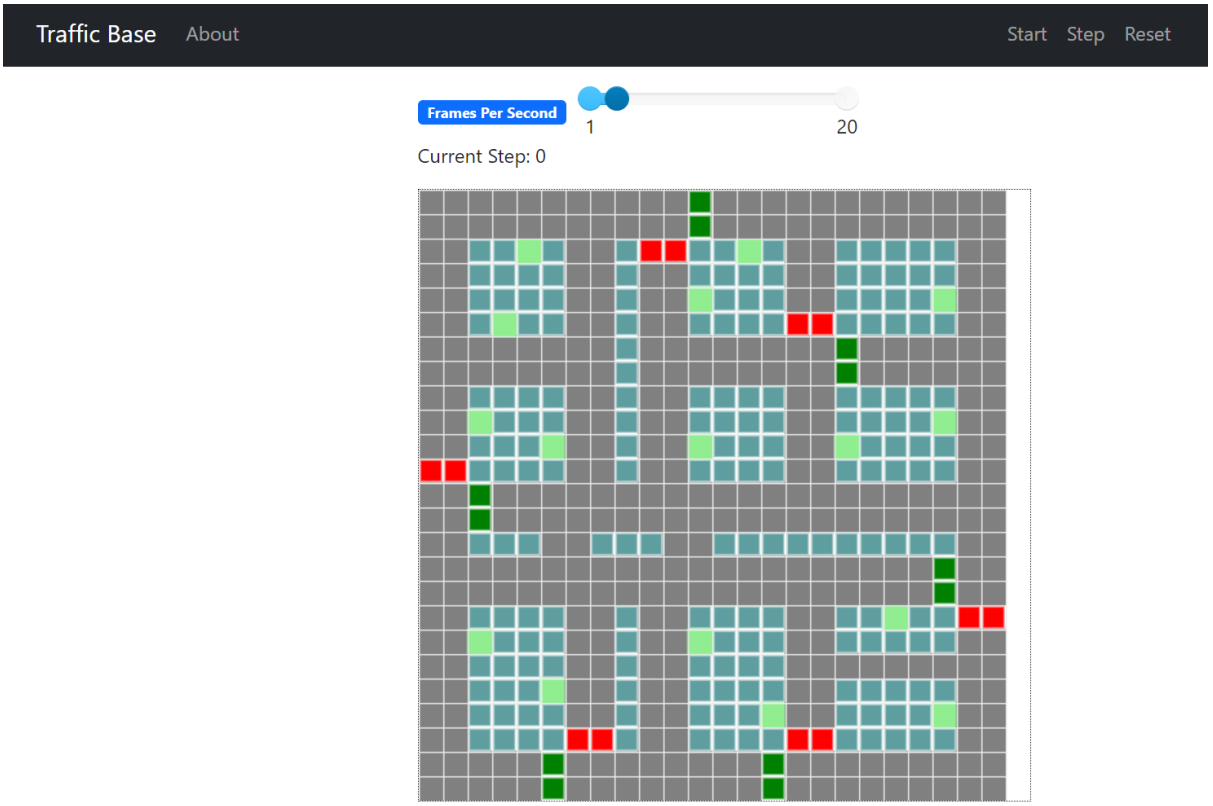


Imagen 3: Vista del modelo sin coches generado en mesa.

Nos quedaría únicamente hacer posible estas características, así que se programó a cada agente en una capa inferior a la del agente, consiguiendo así que este pueda moverse por el mapa con el movimiento indicado en el mapa, y evitando a toda costa chocar con otro agente de su mismo tipo y a la vez respetando las reglas de tránsito (semáforos únicamente). Finalmente, con todo esto explicado, y con la lógica de que el agente puede llegar o no a su destino inicial, podemos pasar a hablar del diseño y la lógica de los agentes de nuestro ambiente.

El diseño de los agentes (objetivo, capacidad efectora, percepción, proactividad, métricas de desempeño, etc.).

En la propuesta se plantea la existencia de 5 agentes cuyo objetivo sea simular las acciones del ambiente o interactuar con el mismo, por lo que esta tabla ayudará a entender un poco de todo lo que abarca cada agente:

	Objetivo	Capacidad efectora	Percepción	Proactividad	Métricas de desempeño
Carro	Llegar a su destino o a un destino	Este agente no cuenta con	Sólo conoce que hay en su misma	No es proactivo ya que solo	No cuenta con métricas de

	que encuentre en el camino.	capacidad efectora ya que es influenciado por otros agentes, pero el no influencia a ninguno.	celda y si están ocupadas las celdas de sus costados y enfrente una sola posición.	reacciona a lo que le indique el contenido de los agentes de su misma casilla.	desempeño.
Semáforo	Cambiar de color entre rojo y verde en un lapso de tiempo.	Frena o indica que se pueden mover los coches dependiendo de su color.	No tiene percepción alguna.	No es proactivo debido a que el tiempo de cambio de calor es indicado por los programadores.	No cuenta con métricas de desempeño.
Destino	Eliminar al agente Carro que esté sobre él.	Elimina al coche que esté en su misma casilla.	Percibe únicamente lo que hay en la misma casilla.	No es proactivo debido a que solo realiza una acción y no puede tomar una decisión propia.	No cuenta con métricas de desempeño.
Obstáculo	Existir.	Esta casilla no es accesible por lo que no tiene ninguna capacidad efectora.	No percibe nada.	No es proactivo debido a que solo está existiendo y no tiene ninguna acción.	No cuenta con métricas de desempeño.
Calle	Indicar al agente auto la dirección de movimiento que deberá tomar.	La capacidad efectora es la asignación de dirección en la que se moverá el agente Carro.	No percibe ninguna casilla o agente, es percibida por el coche.	Las calles que son intersecciones toman una decisión aleatoria para asignar la dirección por lo que se podría	No cuenta con métricas de desempeño.

				considerar proactivo	

La arquitectura de subsunción de los agentes.

Car(Agent):

- Conoce la posición del agente Destination y Road (Sensores de posición)
- Conoce sus vecinos (Sensores de proximidad)
    - Para poder pararse si tiene otro tipo de agente

Mecánicas:

- Encontrar su destino. Conoce su destino
    - El agente carro no tiene capacidad efectora, sigue el comportamiento de los otros agentes los cuales sí tienen capacidad efectora.
- Movimiento con respecto al agente Road.

Traffic_Light(Agent):

Mecánicas:

- Dos tipos de estados:
    - Estos estados son de variables booleanas las cuales pueden tener True y False. Si es True cambia de color a verde y si es False Roja.
- Detienen al agente Car.
    - Si se encuentra en estado False y hay un agente de tipo Car por encima de este hace que se detenga.
    - Si cambia a True deja que el el agente Car siga con su movimiento
    - Conoce su estado actual.

Destination(Agent):

Mecánicas:

- Detienen al agente Car.
    - Si un tipo de agente Car se para encima de él hace que el agente se detenga.
- Desaparece el agente Car.
    - Cuando sabe que tiene un agente de tipo Car encima de él lo desaparece del modelo.

Obstacle(Agent):

Mecánicas:

- Existe en el modelo solo para adornar y ser evitado por el agente

Road(Agent):

- Toma una decisión
    - Si el agente tiene la posibilidad de elegir entre dos opciones tomará una decisión y tendrá un valor de dirección
- Asignan movimiento al agente
    - Asignan el tipo de movimiento al agente cuando uno pasa encima de este.

Características del ambiente.

El ambiente en el que existe el agente es accesible debido a que tiene un movimiento libre desde que es inicializado hasta el punto destino al que decida acceder, teniendo un comportamiento de acuerdo a lo que recibe cuando interactúa con los demás agentes, a su vez, es un entorno episódico gracias a las decisiones que toman algunos agentes Road son tomadas en el instante en que lo necesita. Otra característica del ambiente es que es estático, ya que no es modificado en ningún momento, el que sufrirá modificaciones es el agente, sin dejar de lado que es discreto gracias a que las acciones son finitas. La última característica que tiene nuestro ambiente es determinista, ya que los agentes que son parte del entorno terminan siendo los que determinan las acciones que realizarán los agentes Car.

Integración de Unity:
El modelo generado gráficamente en unity fue posible gracias al API que creamos y probamos en Thunder Client, y básicamente usamos un código llamado Agent Controller con las bases dadas por los profesores para poder realizar el movimiento de los agentes, esta muy extenso el código debido a que en él realizamos los desplazamientos con matrices, las interpolaciones, la instanciación de todos los tipos de GameObjects que se ven reflejados en el modelo, cuyos prefabs fueron generados manualmente en blender, desde la creación de cada una de las caras hasta los materiales son creados por los integrantes del equipo con la finalidad de desarrollar habilidades de diseño que pudieran hacer que tuviera esa esencia el el juego, un estilo que fuera nuestro. De igual forma en este Agent Controller se hacen las llamadas al API para obtener la información que necesitamos para poder aplicar transformaciones o cambios de estado a los agentes. En el repositorio se encuentra un video de la simulación corriendo durante un minuto con un Time de 0.5, generando coches cada 10 steps y guardando la información de cuántos coches han llegado a un destino cada 100 steps. No fue mencionado en el orden que debía pero se utiliza Flask para poder hacer esta conexión entre Unity y el modelo de Mesa

Conclusiones.

- El control absoluto del comportamiento de todos los agentes Car, realmente no ayudan en ningún aspecto a que se logre resolver la problemática debido a que con inteligencia artificial, no se puede tener el factor humano de las emociones, pensamientos intrusivos, estrés, entre otras.
- La modelación gráfica de un sistema puede ayudar a ejemplificar de manera clara y concisa un escenario que pretenda mostrar un comportamiento y analizar las situaciones diversas que estas puedan generar.
- La inteligencia artificial es la clave para la creación de modelos capaces de reflejar casos hipotéticos en su mayoría, logrando evaluar de manera gráfica los resultados del caso propuesto.