Entregable Personal

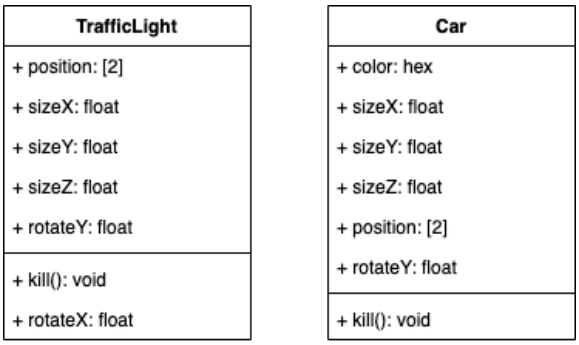


Diagrama de Clases (Gráfico)

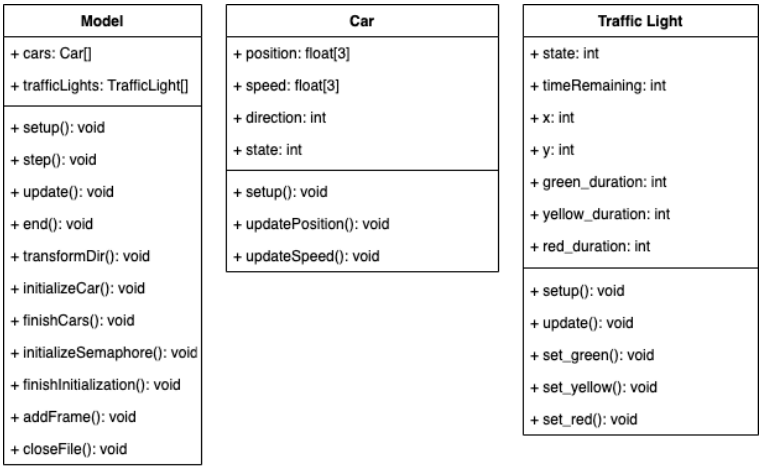
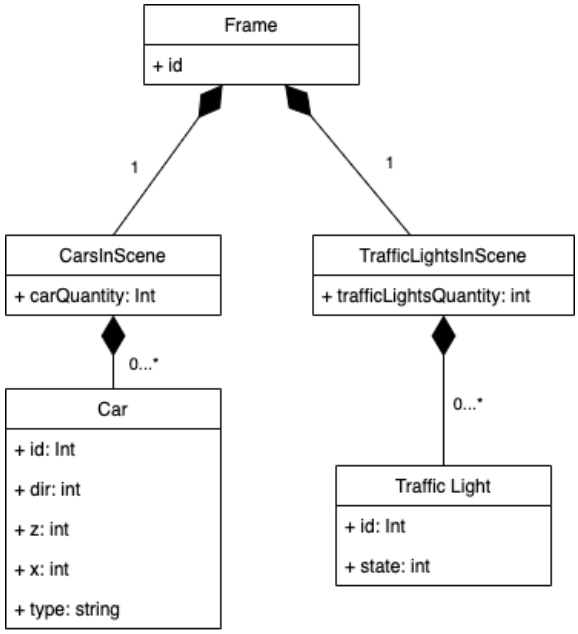


Diagrama de Clases (Multiagentes)



Modelo de Datos

1. **Describa cómo representarías el entorno en una retícula rectangular de NxM casillas.**

Para el caso de este entregable en lugar de usar una retícula rectangular (grid) usamos un Space el cual es continuo para usar de ambiente para la simulación de multiagentes. Para esta entrega entonces usamos un Space de 1,000 x 1,000 donde simulamos nuestras dos calles que van en direcciones opuestas en las que se desplazan los vehículos. Esto fue para hacer la simulación en la parte de multiagentes, para poder llevar estas coordenadas ya a la parte gráfica tuvimos que hacer adaptaciones puesto que en agentpy el Space solo es el primer cuadrante del plano cartesiano por lo que su (0, 0) es en la esquina inferior izquierda.

Mientras que en ThreeJs al ser un espacio 3D tiene todos los ejes y su centro es el (0, 0, 0). Entonces tuvimos que usar un valor de desplazamiento para adaptar las coordenadas de los agentes a la parte gráfica.

2. Enliste las diferentes situaciones (percepciones del estado del entorno) a las que se enfrentarían los conductores.

Los conductores deben de ser capaces de percibir dos cosas en su entorno, la primera es los otros vehículos que hay en la simulación y la segunda son los semáforos que hay en el ambiente de simulación. En el caso de los otros vehículos deben de estar al pendiente de ellos para mantener su distancia de ellos para evitar que lleguen a chocar con ellos. Y en el caso de los semáforos para saber el estado en el que se encuentra el semáforo de su carril y en base a eso saber si puede seguir avanzando si es verde, acelerarle si es que esta en amarillo y cerca del semáforo y en caso de que este en rojo detenerse hasta que cambie a verde.

Para el caso de la entrega final en la cual se va a tener un cruce ahí se tendrá un nuevo aspecto para el caso de los semáforos puesto que en esa simulación ya se tendrán semáforos para dar vuelta, así que en esos casos el vehículo tendrá que cambiar su dirección.

3. Defina las acciones que llevarían a cabo los conductores para cada una de las situaciones que consideraste en el punto anterior.

Como se mencionó se tomarán en cuenta los demás vehículos entonces los conductores buscan el vehículo que está frente a ellos y saben la distancia que hay entre ellos para de esa manera si se llegan a estar muy cercas empezar a disminuir velocidad y no chocar. En el caso de los semáforos depende de la luz que representa un estado. Verde avanzar y seguir con su velocidad, si esta cerca del semáforo y esta en amarillo acelerarle, si no esta cerca y es luz amarilla debe disminuir, y ya si esta en rojo pues disminuir velocidad y detenerse.

Para la entrega final en el caso de las vueltas deberá checar el semáforo y cambiar su dirección (si es muy realista debería de reducir la velocidad y luego cambiar

dirección).

4. Programe una simulación en Python para esta situación. Recopila información tal como velocidad a la llegada del semáforo, cantidad de autos detenidos cuando está en rojo el semáforo, etc.

Por el momento es el archivo: FinalSimulation.ipynb, el cual se encuentra en el repositorio de esta entrega.

5. ¿Qué pasaría en la simulación si el tiempo en que aparece la luz amarilla se reduce a 0?

Si esto pasa entonces la luz amarilla no tendría tiempo por lo que se cambiaría de estado (color de luz) de verde directamente a rojo, por lo que será más brusco el frenado que harán los carros porque deben de disminuir su velocidad rápidamente para frenar totalmente.