Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

Reto

Integrantes:

Rodrigo Galván Paiz | A01721158

Andrés Aguirre Rodríguez | A01284373

Andres Fernando Garza Garcia | A01138704

Daira Adriana Chavarria Rodriguez | A01274745

TC2008B Grupo 104

Docentes:

Jorge Mario Cruz Duarte

Maria Angelica Barreda Beltran

1 de septiembre del 2022

Contents

Diagrama de clases

Diagrama de clases Python

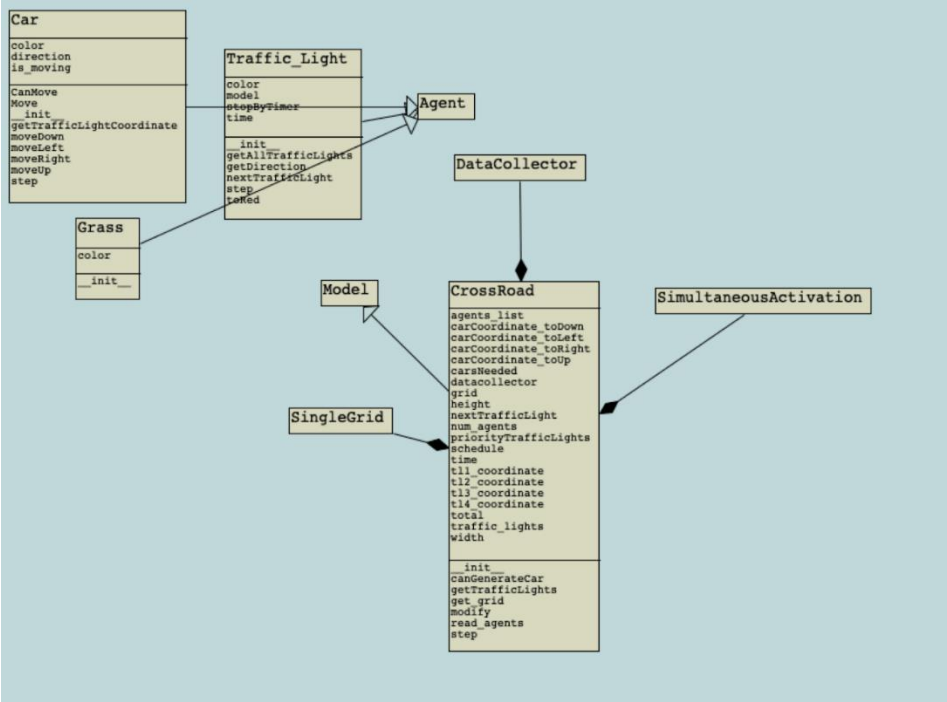

Figura 1 Diagrama UML Python

Diagrama de clases Unity

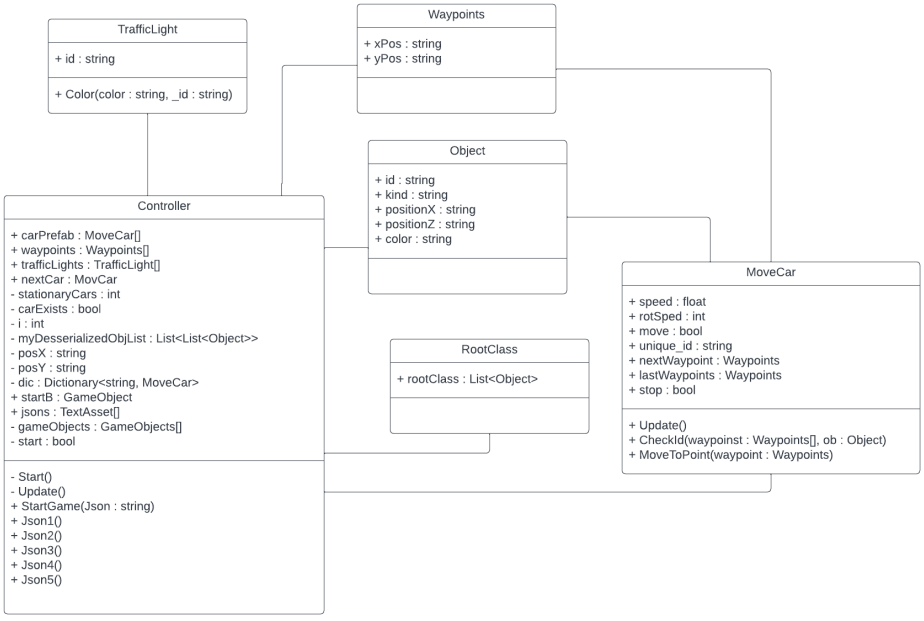

Figura 2 Diagrama UML Unity

Protocolos de interacción

Funcionalidad de Autos:

Cada auto cuenta con una dirección inicial (derecha, izquierda, arriba y abajo). Dependiendo de ella, definirá su posición inicial en el mapa, recordando que se trata de un cruce con 4 semáforos.

El auto avanzará sólo si existe un espacio disponible frente al mismo, es decir, que no haya otro coche estorbando su paso. Además de ello, si detecta un semáforo en rojo como parte de sus vecinos, tampoco avanzará. Para que esta última condición funcionara, se colocaron los semáforos de manera estratégica para que sólo los autos con determinada dirección fueran afectados. En resumen, el carro tiene que checar por dos condicionales y estas son las siguientes. Si hay un carro enfrente y si es que tiene un semáforo vecino si este está en rojo o verde. Por último, se tiene una condicional dentro de los cuadros en el centro de la intersección en donde el carro toma un valor del 1 al 6 y si esta valor es 1 o 2 entonces toma una vuelta a cierta dirección. Esto permite que los carros tomen la vuelta dentro de las intersecciones.

Por otra parte, cada coche generado elige un color aleatoriamente para poder ser identificados más fácilmente. Esto en si no tiene ningún impacto sobre los carros. Además, cabe mencionar que en cada paso de la simulación se intenta instanciar un nuevo carro, es decir si el espacio está vacío en el lugar predeterminado donde se instancian los carros entonces se instancia un carro nuevo. Cuando se llegan a instanciar estos carros se les da una dirección específica predeterminada dependiendo de en donde se llegó a instanciar dentro del mapa. Estas direcciones definen si el carro va hacia arriba, abajo, hacia la izquierda o hacia la derecha una vez que se instancia.

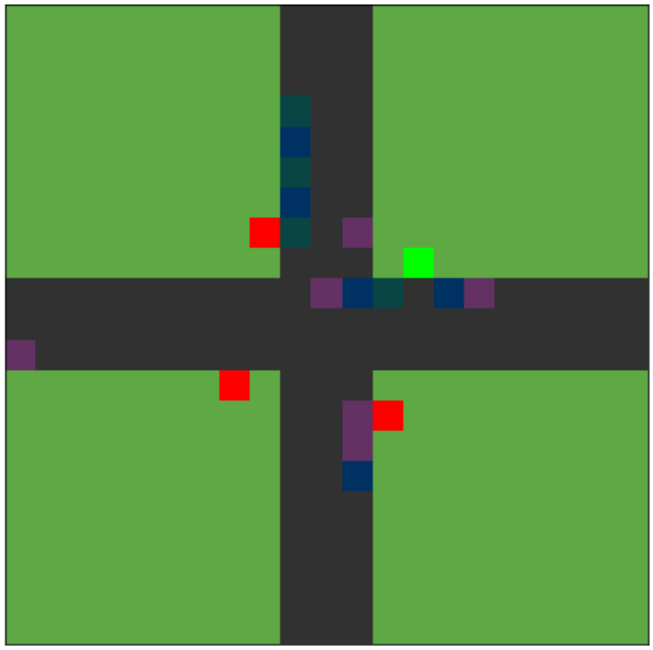

Figura 3 Carros en simulación

En la imagen superior podemos observar los coches en colores azul, verde oscuro y morado. Al tener el semáforo en verde, los coches que avanzan hacia la izquierda pueden moverse. Por otra parte, aquellos que van hacia arriba y abajo siguen formados en hilera en espera a su turno.

Protocolo de Semáforos:

Debido al diseño del cruce, se utilizan 4 semáforos en el modelo, cada uno responsable de su respectiva dirección. En la imagen inferior se puede observar la colocación de los mismos, siendo una forma estratégica de que los coches no detecten al semáforo incorrecto al estar avanzando.

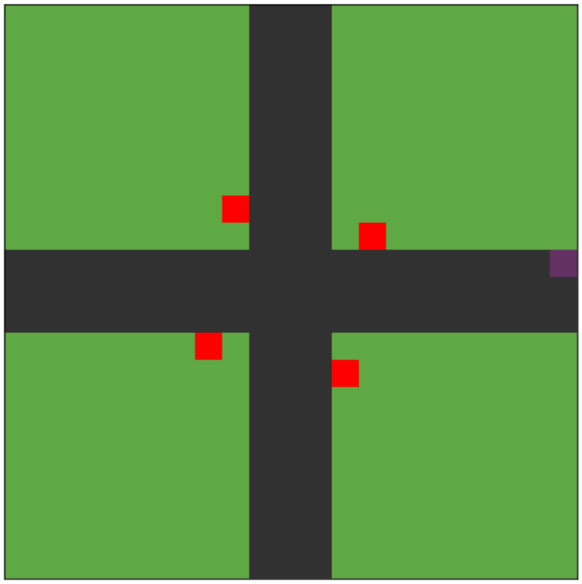

Figura 4 Semáforos en simulación

En cuanto a los semáforos estos cuentan con un límite de tiempo que si se encuentran mucho tiempo en verde este límite de tiempo los obliga a poner una luz roja. En cuanto a sus condicionales estas son simples ya que lo único que llegan a hacer es ver si hay un carro esperando al lado del semáforo y si los demás semáforos están en rojo. Si estas dos condicionales son verdaderas entonces el semáforo se pone de color verde. Con estas condicionales sencillas nada más los semáforos que tienen carros esperando se vuelven en verde y si llegan a detectar que ya no hay carros se ponen de color rojo. Esto garantiza un funcionamiento eficiente dentro de los cruces.

Proceso de Instalación:

En cuanto al proceso de instalación en este tan solo se tiene que descargar el ZIP que contiene el proyecto, cargar el proyecto en Unity, cargar la escena principal y correr el programa. Dentro de la simulación aparecerá un menú de opciones que presenta al usuario con cinco diferentes simulaciones precargadas. Tan solo se tiene que escoger una opción y proceder a visualizar la simulación.

GitHub

Link: https://github.com/RodrigoGalvan/MAS_CG_Reto.git

Video de simulación ejecutandose

Link: https://youtu.be/xP5Kdh0ON00