Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

Actividad Integradora

Integrantes:

Rodrigo Galván Paiz | A01721158

Andrés Aguirre Rodríguez | A01284373

Andres Fernando Garza Garcia | A01138704

Daira Adriana Chavarria Rodriguez | A01274745

TC2008B Grupo 104

Docentes:

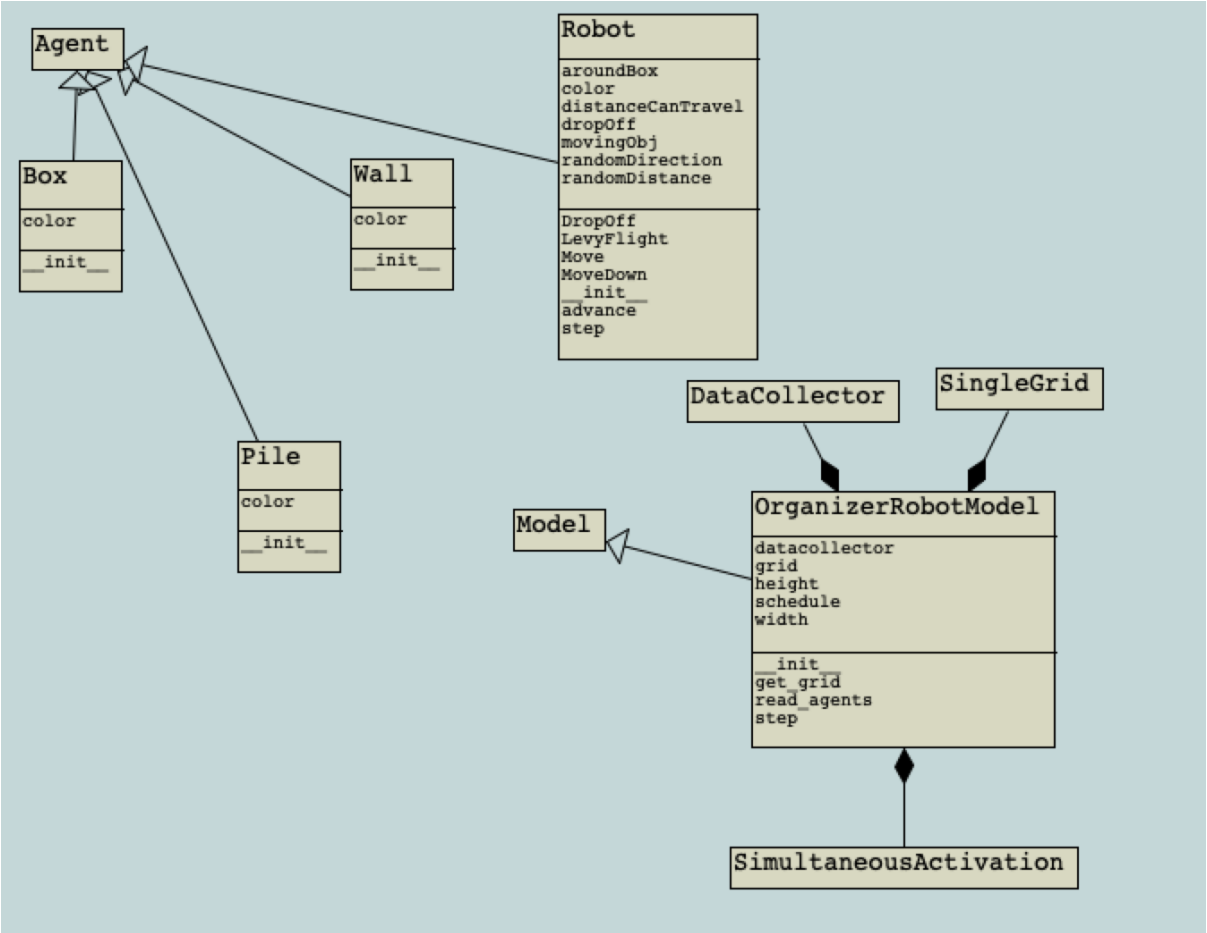
Jorge Mario Cruz Duarte

Maria Angelica Barreda Beltran

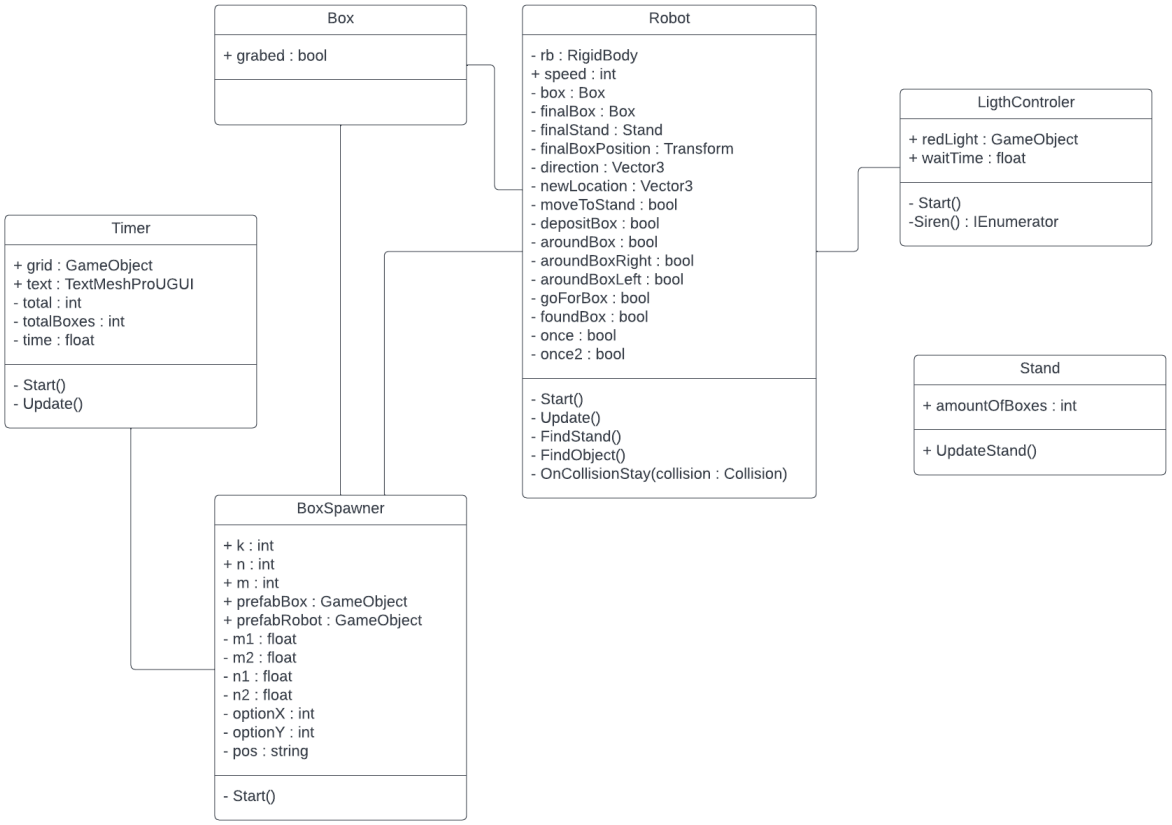
17 de agosto del 2022

Diagramas de clase:

Python:



Unity:



Estrategia de robot individual:

Python:

Al momento de iniciar la simulación, los robots se posicionan en lugares al azar, estando programados para no posicionarse en la fila de hasta arriba, ya que es donde se pondran las cajas despues de ser recogidas, o posicionarse en un lugar donde a hay otro robot o caja. Después de esto, cada robot se mueve de manera aleatoria, revisando que el espacio al que planea moverse este vacío, y cada paso revisa sus vecinos para verificar si hay una caja. En el caso de intentar moverse a la derecha y tener un obstáculo, se mueve un espacio abajo para evitar dicho obstáculo. Si no hay cajas cerca continúa moviéndose aleatoriamente hasta encontrar una. Al momento de encontrar una revisa si no tiene una caja, y si no tiene una la recoge. Después de recoger la caja su color cambia para tener una manera fácil de identificar su estado. Luego de eso los robots procuran moverse a la columna hasta la izquierda, o la fila debajo de las estanterías para tener un camino rápido para dejar sus cajas. Al llegar a las estanterías revisan la cantidad de cajas en la estantería posicionada más a la izquierda, y si hay espacio ahí la dejan, si no se mueven a la siguiente.

Unity:

En cuanto al programa en Unity los robots se instancian de manera alatoria en un lugar predeterminado al lado de los estantes. Estos robots llegan a buscar la caja más cercana a ellos y le hacen saber a los demás robots que va a ir por esa caja. Este robot procede a hacer un vector hacia la caja, es decir el camino más corto hacia la caja. Si este robot colisiona con otra caja o con otro robot entonces este mismo detecta si está más a la izquierda o a la derecha con el objeto que colisionó y se mueve dependiendo de esto. Una vez que tiene la caja procede a buscar al estante que tiene más proximo a él mismo y el robot asegura que este estante no tenga ya 5 cajas. Una vez encontrando el estante procede a hacer un vector hacia ese estante para dejar la caja. Este ciclo se repite hasta que el robot detecta que ya no se encuentra ninguna caja dentro del cuarto. Si es que ya no hay cajas dentro del cuarto entonces el robot va hasta la parte superior del cuarto a estacionarse para no estorbar a los otros robots.

Función de estanterías:

Python:

Las estanterías están posicionadas en la fila de hasta arriba en la simulación, teniendo un color diferente para que puedan ser identificadas fácilmente. Su funcionalidad es sencilla ya que cada cuadro es una estantería diferente, con capacidad de 5 cajas. Dependiendo de la cantidad de cajas el color de esta estantería cambia, tomando un color más oscuro mientras más cajas tiene. Siempre se revisa primero la estantería posicionada más a la izquierda al momento de que se quiere acomodar una caja.

Unity:

Las estanterías están posisionadas en la parte inferfior del cuarto. Estas cajas llegan a tener hasta 5 cajas y se despliegan en los estantes las cajas cada vez que un robot deja la caja. El estante también lleva cuenta de cuantas cajas tiene.

Comportamiento de cajas:

Python:

Las cajas aparecen de manera aleatoria en el grid, evitando las zonas de la columna izquierda, y las dos filas de arriba, ya que son los lugares de las estanterías, y los caminos que recorren los robots cuando ya tienen una caja en mano. El cuadro de caja desaparece del grid una vez que uno de los robots la recogen, se cambia el color del robot al tener una caja, y luego cambia el color de la estantería cuando tiene una caja o cajas.

Unity:

Las cajas se instancian alatoriamente dentro del grid y se aseguran que a la hora de aparecer las cajas no este ninguna otra caja en la posicion donde esta se va a poner.

Cómo mejorar el programa:

En cuanto a la simulacion de python la manera en la que esta se podria mejorar significativamente es haciendo que los robots no se muevan usando el patrón de Lévy Flight sino que busquen dentro del grid en el que están la caja más cercana a su posicion como se hizo dentro de Unity. Ademas de esto se podria mejorar el programa al hacer que los robots encuentren el estante más cercano para dejar la caja. Ademas se podria mejorar el programa haciendo que los robots primero vean todos los caminos posibles para evadir un objeto que está en su camino y en base a la distancias escoja el camino más corto para llegar a su destino. Todo esto reduciria lo que viene siendo el tiempo que le toma al programa en ejecutarse, lo cual depende enteramente de que tan rapido los robots pueden llegar a organizar las cajas. Cabe mencionar que como está programado ahorita los robots podrian llegar a tomar un tiempo muy largo para organizar las cajas ya que su movimiento está randomizado.