Flash Point Fire Rescue

HIRAM ISRAEL MENDOZA LÓPEZ A01613963
SANTIAGO YAEL NIETO CASTILLO A01612348
GONZALO MORÁN SÁNCHEZ A01710494

SITUACIÓN PROBLEMA

Simular el juego de mesa Flash Point Fire Rescue utilizando multiagentes, el juego básicamente consiste en rescatar personas de una casa que se esta incendiando.

El objetivo del proyecto es aprender acerca del uso y manejo de sistema multiagentes, ademas de hacer un modelado en Unity.

IMPLEMENTACIÓN

¿DE QUE MANERA?

LO PRIMERO QUE TENEMOS QUE TENER EN CUENTA SON LAS REGLAS DEL JUEGO DE MESA, Y LAS CONDICIONES INICIALES DE LA SIMULACIÓN: PAREDES, FUEGOS INICIALES, PUNTOS DE INTERÉS, ENTRADAS Y PUERTAS

¿QUÉ SE HIZO?

SE IMPLEMENTO UNA CLASE LLAMADA BUILDING MODEL EN DONDE SE MODELA EL EDIFICIO TANTO SUS DIMENSIONES, LOS NÚMEROS DE AGENTES, ESTRUCTURAS DE LAS PAREDES, PUERTAS, FUEGO ENTRADAS Y PUNTOS DE INTERÉS. TODO ESTO TOMADO DE UNA DIFERENTES MATRICES DONDE SE INICIALIZA LAS POSICIONES DE LO YA MENCIONADO

IMPLEMENTACIÓN

¿QUÉ SE HIZO?

DE IGUAL FORMA SE DECLARA UNA CLASE BOMBERO COMO AGENTE, EN ESTE APARTADO SE DECALARA LAS INTERACCIONES QUE TIENE EL AGENTE CON SU ENTORNO, COMO ABRIR PUERTAS, DESTRUIR PUERTAS, MOVERSE POR EL TABLERO, ETC. PARA PODER REALIZAR ESTAS ACCIONES CUENTA CON UN CONTADOR LLAMADO “PUNTOS DE ACCIÓN” QUE REGULA LAS INTERACCIONES CON EL ENTORNO.

DE IGUAL MANERA SE CREARON DOS CLASES, LA PRIMERA LLAMADA NODO, QUE ES PARTE DE LA CLASE FILA QUE AYUDA A CONTROLAR LOS TURNOS DE LOS AGENTES EN CADA MOVIMIENTO.

Boceto del tablero de juego en Unity

METODOLOGÍA

- Los bomberos son modelados como agentes individuales que interactúan con el entorno de grilla (representado por MultiGrid). Cada agente es colocado en el grid y se gestiona a través de turnos por el método step().

- Los agentes son gestionados mediante una Fila, lo cual es una estrategia de coordinación entre agentes. Cada agente tiene una cantidad de puntos de acción (action_points), y el orden en que actúan está determinado por la fila.

- La dinámica del fuego se gestiona a través de la matriz_fuego, que se actualiza en cada paso del modelo. La propagación del fuego esta basada en tiradas de dados y la verificación de las celdas vecinas.

CONEXIÓN CON SERVIDOR

SERVER.PY: PROCESA LOS DATOS Y DEVUELVE RESPUESTAS JSON QUE INCLUYEN LAS POSICIONES Y ESTADOS DE LOS OBJETOS EN LA SIMULACIÓN.

WEBCLIENT.CS: ENVIA DATOS EN FORMATO JSON AL SERVIDOR Y RECIBE LAS RESPUESTAS, LAS CUALES SE UTILIZAN PARA ACTUALIZAR LA POSICIÓN Y ESTADO DE LOS OBJETOS EN LA SIMULACIÓN.

CONCLUSIONES Y AREA DE OPORTUNIDAD

Este reto representó un desafío importante, pero más allá de las dificultades, nos brindó una comprensión profunda sobre el manejo de sistemas multiagentes y gráficas computacionales modeladas en Unity. A pesar de que no alcanzamos todos los objetivos que nos propusimos al inicio, el proceso nos permitió adquirir nuevas habilidades y tener más claros conceptos que sin duda serán útiles en proyectos futuros.

¡Gracias por su atención!