Logotipo, nombre de la empresa

Descripción generada automáticamente**Instituto Tecnológico y de Estudios Superiores de Monterrey**

**Campus Querétaro**

Modelación de Sistemas Multiagentes con Gráficas Computacionales (Gpo 102)

**RETO AD2023\_ROBOTS LIMPIADORES**

     Profesores:

Pedro Oscar Pérez Murueta

Alejandro Fernández

Denisse Lizbeth Maldonado Flores

Integrantes del equipo:

Ian Joab Padrón Corona | A01708940

Uri Jared Gopar Morales | A01709413

María Fernanda Moreno Gómez | A01708653

Miércoles 06 de septiembre de 2023

**RETO AD2023\_ROBOTS LIMPIADORES**

# Introducción:

Consiste en desarrollar la solución para que 5 robots limpien una oficina, la cual se encuentra llena de basura. Cada uno de estos robots cuentan con un almacenamiento máximo de 5 basuras, además de sensores que les permite determinar la basura que se encuentre en su posición, además de sensores de colisión alrededor de los mismos y brazos de recolección para la basura.

La oficina cuenta con una papelera con capacidad infinita y todos los robots conocen en qué lugar se encuentra, y por cada vez que los robots llenen su capacidad de almacenamiento estos se van a dirigir a la papelera para vaciarlo, para que de esta manera puedan seguir limpiado y acabar con su trabajo.

# Herramientas implementadas:

* Lenguaje: Python
* Simulación: Unity
* Librerías: Mesa, Matplotlib, Numpy.
* Espacio de desarrollo: Visual Studio Code

# Solución:

Para lograr la solución al presente problema, comenzamos por decidir que cada uno de los objetos representados en el mapa serían agentes de la librería Mesa (Litter, Wall, PaperBin, Robot). Esto con la finalidad de que tuvieran un atributo con su tipo de agente, y así mismo poder diferenciarlos de una forma más rápida. Además, nos facilita la exploración que programamos para los robots, para que supieran que es lo que se encuentra en las celdas visitadas o a su alrededor (para el caso de muros). Lógicamente, el agente con mayor cantidad de atributos son los robots, ya que tienen las tareas de exploración, recolección evitar los obstáculos, etc. Dadas estas consideraciones, dividiremos nuestra lógica en varias etapas.

# Etapa Uno: Exploración:

Para esta primera etapa decidimos que los robots comienzan con movimientos aleatorios por todo el mapa. Tomando en cuenta la descripción del problema, los robots no pueden estar en la misma celda ocupada por otro robot después de salir de su celda de origen y si se topa con un obstáculo deberán rodearlos. Los movimientos aleatorios que hacen los robots consideran esto. Al mismo tiempo, estos movimientos continúan por n steps, donde n es igual al número de celdas que existe en el mapa. A través de varias iteraciones descubrimos que, al terminar esta etapa, los robots exploraban entre el 60 y 70% del mapa.

Los robots cuentan con un mapa compartido en el modelo (el cual se inicializa en blanco) con la finalidad que, con cada nueva celda explorada por los robots, se revelara su contenido y se actualizara el mapa. Al inicializar los objetos representados en el mapa como agentes y ponerles de atributo el tipo el robot es capaz de saber cuál agente encontró (basura, papelera, pared, u otro robot).

Al finalizar los movimientos aleatorios, con ayuda de su mapa interno pueden saber las celdas que todavía faltan por explorar. Estas celdas serán alcanzadas por los robots con la ayuda de un algoritmo de búsqueda BFS (Breadth First Search), lo que hace es que de las posiciones donde se encuentran los robots buscan el camino más corto para estas celdas no exploradas. De esta manera son capaces de poder explorar todo el mapa y saber que existe en cada celda de este.

# Etapa Dos: Recolección

Acabada la etapa de exploración, entrara la etapa de recolección. Al tener el mapa explorado y los tipos de agentes que existen en cada celda, es momento que los robots se muevan a cada celda que concuerde con el tipo de agente de la basura. Para ello, el programa primero itera sobre toda la matriz de celdas exploradas y obtiene las posiciones donde los robots encontraron basura, para después agregarlas a una lista.

Una vez reconocidas las celdas que cuentan con basura, los robots invocarán una función para determinar cuál de estas celdas es la más cercana a su posición actual. Con esto podrán ir hacia esa celda y comenzarán a recolectar la basura que puedan (En base a la capacidad que tienen en ese momento). Para calcular estos movimientos los robots vuelven a hacer una búsqueda con nuestro algoritmo de BFS, trazando el camino óptimo hacia dichas celdas.

Posteriormente, los robots “eliminarán” los agentes del tipo basura de estas celdas y aumentarán su contador interno de basura (lo cual referimos como el proceso de recolección). Si los robots terminan de recolectar la basura de una celda antes de ir a la papelera, y todavía les queda espacio, irán a otra celda cercana por más basura. Una vez cumplan con su capacidad de 5 basuras, los robots van a la papelera para dejar toda la basura recolectada (en términos del código, es aquí cuando los robots resetean su contador de basura a 0).

Al llegar a la papelera, los robots verifican si todavía existe basura en la celda de la que provenían, lo cual significa que no lograron limpiar toda la basura de su celda. Si esto es cierto, estos volverán a esta celda para acabar con la basura. Esta funcionalidad es posible ya que cada robot tendrá una memoria de la celda en la que se encontraba y si se vació cuando recogió basura de esa celda, por lo que si llega a 0 esta memoria se borra. En caso de que no sea cierto, significa que ya terminó de recoger basura en la celda que tenía asignada ese robot, por lo que se le asignará una nueva celda de la lista que se planteó al inicio de esta etapa.

# Etapa Tres: Finalización

Finalmente, cuando un robot ya no puede ir a una celda con basura, (debido a que ya hay otros robots asignados a estas o ya no hay más en la lista de celdas con basura), estos comenzarán a moverse aleatoriamente por el mapa (sin pasar por la papelera). en espera de que sus compañeros robots terminen de dejar la basura restante a la papelera. Al finalizar todos los robots, el programa terminará, dejando saber al usuario la cantidad de steps que les tomó a los robots terminar su misión de limpieza.