DESARROLLO DE UNA INTERFAZ DE UNA ASPIRADORA DE BASURA

Velasco Reyes Juan Ramón, Gallegos Alvarez Jovani, García Canteros Ángel Josue
{*verj000624 , gaaj010320, garciacanterosangeljosue057*}*@gs.utm.mx*
Profesor: Dr. Juan Juárez Fuentes

I. INTRODUCCIÓN

Este proyecto se enfocará en la creación de una interfaz en Python, complementada por una visualización en Javascript para su ejecución en pagina web. El desarrollo y la documentación detallada de este proyecto serán cruciales para asegurar su funcionalidad y eficiencia, alcanzando así una representación eficaz del funcionamiento de una aspiradora de basura en un entorno simulado.

El desarrollo del proyecto requiere no solo una representación visual y funcional, sino también un conjunto de algoritmos eficientes que permitan el desplazamiento, la recolección de objetos y la gestión de obstáculos en el entorno simulado.

Para lograr estos objetivos de manera efectiva, se recurre a heurísticas, que son estrategias simplificadas para la toma de decisiones rápidas, en lugar de métodos exhaustivos. Las heurísticas permiten suposiciones fundamentadas y se emplean ampliamente en ciencias computacionales.

En el caso de esta aspiradora de basura, las reglas heurísticas permiten que los movimientos y decisiones del sistema autónomo sean ejecutados con rapidez, lo cual es esencial para una experiencia interactiva en la interfaz.

II. OBJETIVO DEL PROYECTO

II-A. Objetivo general

Desarrollar una interfaz gráfica mediante Python y Javascript que simule el funcionamiento de una aspiradora de basura.

II-B. Objetivos específicos

- Desarrollar un método de brusqueda aleatorio para la aspiradora para comparar su tiempo de ejecución con los demás métodos.
- Desarrollar un método de búsqueda inteligente que recoja la basura mas cercana al punto de la aspiradora.
- Integrar un método de búsqueda de fuerza bruta que haga un barrido completo en la matriz de espacio para limpiar la basura.
- Implementar un método de búsqueda alterno .^{es}piral"para comparar resultados con los demás métodos.

III. DESARROLLO DE LA INTERFAZ

El proyecto cuenta con 5 archivos de funcionamiento; **app.py, vaccum cleaner, short route, snake, spiral in out** y finalmente **interfaz**. Los archivos de backend `snake.py, short_route, spiral_in_out` y `vacuum_cleaner.py` definen las lógicas de movimiento y recolección de basura del robot, simulando su comportamiento en un entorno de matriz.

El archivo `Vaccum_cleaner.py` nos provee de un funcionamiento no inteligente que le otorga un movimiento aleatorio a la aspiradora. Este movimiento aleatorio claramente provoca un aumento considerable en el tiempo de limpieza de la basura, ya que no se trata de un método de búsqueda como tal.

En cambio, el archivo `Short_route.py` le da la capacidad a la aspiradora de escoger los puntos mas cercanos de cada elemento para minimizar el tiempo de búsqueda de la aspiradora. Este método integrado es el mas rápido en la interfaz y también el más eficiente ya que solo recorre ciertas celdas con el fin de alcanzar un punto cercano entre la basura actual y la siguiente.

En el archivo `snake.py`, el propósito es implementar un modo de movimiento en el que el robot

sigue un patrón en ”serpiente”para recorrer toda la matriz, maximizando la cobertura del entorno en su búsqueda de basura. En su código principal, este archivo inicializa una matriz de tamaño 10x10 con celdas vacías, define posiciones aleatorias para la basura y las almacena en la matriz.

Tal como una busqueda exhaustiva, la aspiradora comienza desde una posición inicial en la esquina superior izquierda y se mueve en linea recta de izquierda a derecha bajando un recuadro cuando llega al limite de la matriz, todo esto mientras recoge basura, actualizando el contador de movimientos en cada cambio de posición. Las condiciones de parada del sistema están definidas de manera que finaliza cuando el robot ha recogido toda la basura presente en la matriz.

Finalmente, el archivo `Spiral_In_Out.py` es un método de movimiento de patrón espiral de afuera hacia dentro. La aspiradora comienza en la posición inicial [0,0] y se mueve hacia el centro de la matriz recorriendo todas las orillas. se trata de un método de búsqueda de fuerza bruta que también recorre todas las celdas de la matriz hasta recoger toda la basura.

Estos dos últimos métodos maximizan el espacio de recorrido de la aspiradora, sin embargo el tiempo de finalización se ve afectado considerablemente.

El archivo `interfaz.html` define la interfaz gráfica que el usuario emplea para visualizar y controlar el sistema de limpieza. Su estructura básica está creada en HTML, con una serie de botones y elementos interactivos para controlar la aspiradora. Se utiliza Bootstrap para dar estilos básicos a los componentes y hacerlos responsivos. Además, se ha añadido un efecto de ”neón.ªl diseño para una experiencia visual atractiva, especialmente en los elementos de control.

En cuanto a las funcionalidades de JavaScript en `interfaz.html`, estas incluyen un temporizador que inicia en cada recorrido del robot, mostrando el tiempo transcurrido desde el inicio de la simulación. La función `updateMatrix(data)` permite actualizar el entorno visual de la matriz en el navegador según el movimiento del robot. Además, un menú desplegable permite al usuario seleccionar el modo de movimiento del robot (aleatorio, zigzag, espiral o ruta corta).

Como parte de nuestro diseño creativo, se añadió un efecto de animación que simula la escritura y el borrado del título de la página, así como botones de control para iniciar, pausar y reiniciar el movimiento del robot en la simulación. La página emplea un diseño en blanco y negro para mejorar la visibilidad de los elementos interactivos, con las celdas de la matriz representando visualmente la basura, la aspiradora y las áreas limpias. El botón de selección de modo y los contadores de basura están resaltados para facilitar su acceso.

La lógica de funcionamiento del sistema comienza con la inicialización de la matriz de 10x10 y la colocación de posiciones aleatorias de basura. La posición inicial de la aspiradora también se asigna aleatoriamente. Dependiendo del modo seleccionado, la aspiradora sigue la lógica de movimiento correspondiente (serpiente, aleatorio, ruta corta o espiral), y cada vez que se posiciona en una celda con basura, esta se marca como limpia y el contador de basura se actualiza. En cuanto a la interacción con el usuario, este puede seleccionar modos de limpieza, iniciar, pausar y reiniciar la simulación. Al finalizar la recolección de basura, el sistema notifica al usuario que la limpieza ha sido completada.

IV. RESULTADOS

Después de tener cada uno de los archivos de métodos de búsqueda y la aplicación app.py, se ejecuta el programa, entonces podremos ver la interfaz de la aspiradora en una pagina web con dirección ”http://127.0.0.1:5000”, dentro de esta pagina podemos observar la interfaz mostrada en la figura 1.

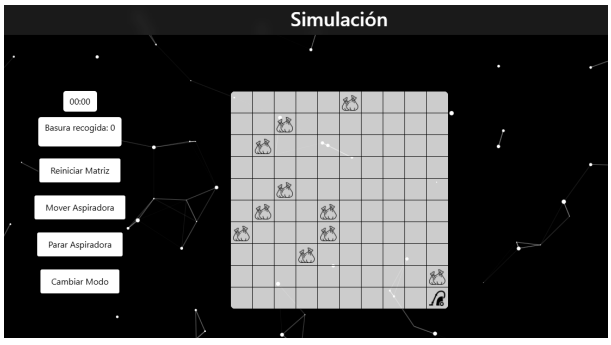

Figura 1. Interfaz principal de la aplicación

La interfaz nos provee de una matriz de 10x10 cuadros de espacio para la posible aparición de la basura y la aspiradora.

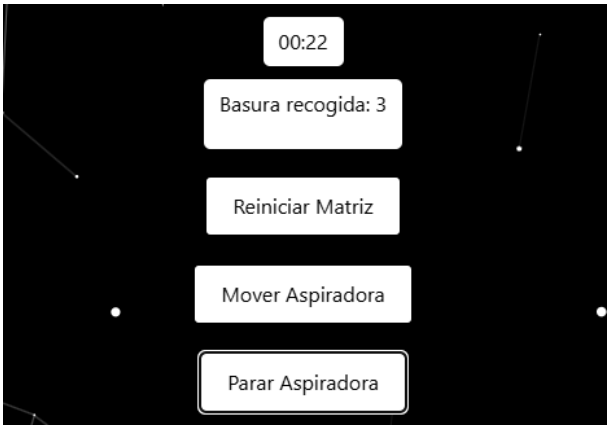

Figura 2. Botones de funcionamiento

Del lado izquierdo de la interfaz podemos observar el **contador**, el cual toma el tiempo final en el cual la aspiradora termina de limpiar el área. Del mismo lado también nos encontramos con el botón de **reiniciar matriz**, este botón nos permite hacer una de la basura y de la aspiradora en tiempo real, ademas de reiniciar el contador de tiempo. Inmediatamente debajo se encuentra el botón **mover aspiradora**, el cual inicia el recorrido de la aspiradora dentro de la matriz. Justo debajo se encuentra el botón **parar aspiradora**, el cual detiene el movimiento de la aspiradora y detiene el contador. Estos elementos se muestran en la figura 2.

Finalmente nos encontramos con el botón cambiar modo (figura 3), este botón posee un slider que permite seleccionar al usuario el modo de búsqueda de la aspiradora, teniendo disponibles las opciones de búsqueda **Random, Shrot Route, Snake y Spiral.** De forma predeterminada, la aspiradora comienza en **modo de búsqueda Random** y después de presionar el botón de inicio, esta comenzará su recorrido hasta terminar de recoger toda la basura, en ese momento el contador se detendrá y **un mensaje emergerá anunciando el final del recorrido de la aspiradora** (figura 4).

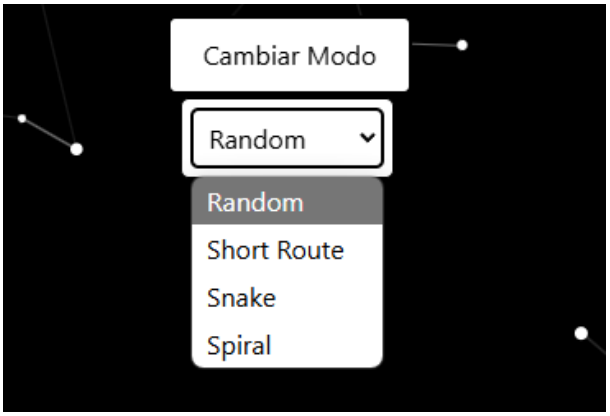

Figura 3. Slider de selección para modo de búsqueda

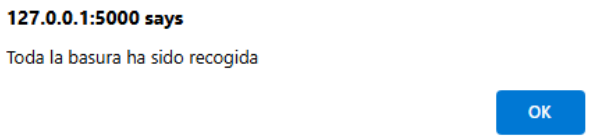

Figura 4. Mensaje de finalización de programa

V. Conclusiones

El sistema automatizado de limpieza simulado cumple con los objetivos planteados al ser capaz de recorrer la matriz y recolectar basura de forma autónoma en diversos modos de movimiento. La integración de Flask permite una comunicación eficiente entre el backend y la interfaz, logrando una actualización visual en tiempo real. Además, la interfaz resulta amigable y funcional, permitiendo al usuario controlar la simulación de manera intuitiva.

El desarrollo de este proyecto ha facilitado la comprensión de algoritmos de movimiento autónomo y de comunicación entre backend y frontend en tiempo real, habilidades esenciales en el desarrollo de sistemas inteligentes.