Aplicación de Búsqueda Informada A* al Robot Aspiradora (PEAS)

Inteligencia Artificial

Autor: **Aaron Rodrigo Ramos Reyes**

Fecha de entrega: 29 de Octubre de 2025

Tarea 2 – Punto B: Aplicación de A sobre modelo PEAS.*

1. Introducción

En esta parte del trabajo se aplica el algoritmo de búsqueda informada **A* (A-star)** al agente **robot aspiradora inteligente**, de acuerdo con el modelo **PEAS** descrito en la tarea anterior.

El entorno se modela como una **cuadrícula de celdas** donde existen:

- Celdas libres (transitables).
- Celdas ocupadas por muebles (bloqueadas).
- Celdas sucias que el robot debe limpiar.

El robot se mueve en cuatro direcciones (arriba, abajo, izquierda, derecha) sin diagonales, y su objetivo es limpiar todas las celdas sucias de acuerdo con dos modos de funcionamiento:

1. **Modo rápido:** Minimiza el tiempo total de limpieza, buscando la distancia Manhattan más corta posible.
2. **Modo eficiente:** Minimiza el consumo energético, penalizando los giros innecesarios y movimientos redundantes.

Ambos modos son simulados visualmente mediante la librería **pygame**, donde se observa el recorrido del agente en tiempo real.

2. Modelo PEAS del agente

- **Performance (Desempeño):**
  - Minimizar el tiempo total de limpieza.
  - Minimizar el consumo energético (número de movimientos y giros).
  - Limpiar todas las celdas sucias sin chocar con muebles.
- **Environment (Entorno):**
  - Cuadrícula de tamaño $N \times N$ (en este caso 8×8).
  - Celdas pueden ser:
    - 0 – Libre.
    - 1 – Mueble (obstáculo).
    - 2 – Sucia (debe ser limpiada).
  - Posición inicial aleatoria del robot.
- **Actuators (Actuadores):**
  - Moverse arriba, abajo, izquierda o derecha.
  - Aspirar (limpiar) la celda actual.
- **Sensors (Sensores):**
  - Detectar si la celda actual está sucia.
  - Detectar si una celda vecina contiene un obstáculo.
  - Detectar su posición dentro de la cuadrícula.

3. Algoritmo A* aplicado

El algoritmo A* busca la ruta óptima entre dos celdas mediante la función:

$$f(n) = g(n) + h(n)$$

donde:

- $g(n)$: costo real desde el inicio hasta el nodo actual.
- $h(n)$: heurística estimada desde el nodo actual al objetivo (distancia Manhattan).

En el contexto del robot:

- $g(n)$ incluye penalizaciones adicionales según el modo:
  - Modo rápido: $g(n) = 1$ (cada movimiento cuesta 1 unidad).
  - Modo eficiente: $g(n) = 1 + 1{,}5$ si hay cambio de dirección (penaliza giros).
- $h(n) = |x_1 - x_2| + |y_1 - y_2|$

El algoritmo se repite para limpiar todas las celdas sucias. Cada vez que el robot llega a una celda sucia, la limpia y busca la siguiente más cercana según el modo activo.

4. Generación del entorno

El mapa es una cuadrícula de 8×8 celdas generada aleatoriamente:

- 10 muebles (obstáculos) distribuidos al azar.
- 6 celdas sucias (de color verde).
- Las demás son celdas libres (gris claro).
- El robot inicia en una celda libre (amarillo).

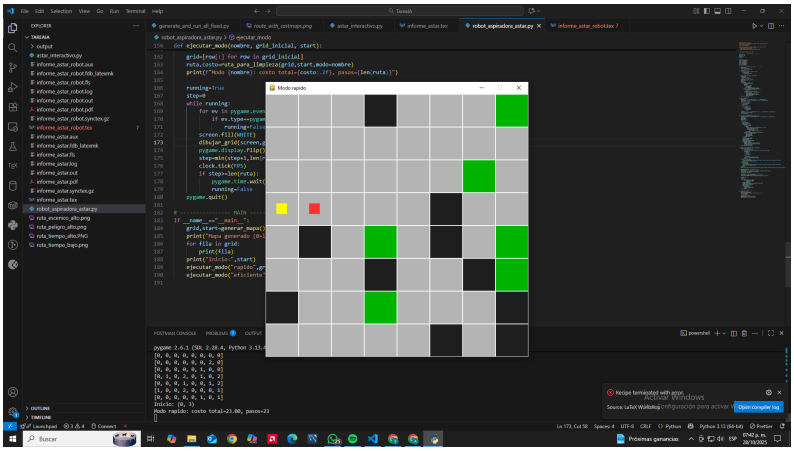

Figura 1: Mapa inicial con obstáculos (negro), suciedad (verde) y posición inicial (amarillo).

5. Modos de operación

5.1. Modo rápido

En este modo, el robot calcula la secuencia de movimientos que minimiza el tiempo total de limpieza. No se consideran penalizaciones por cambio de dirección, por lo que tiende a girar más, pero termina antes.

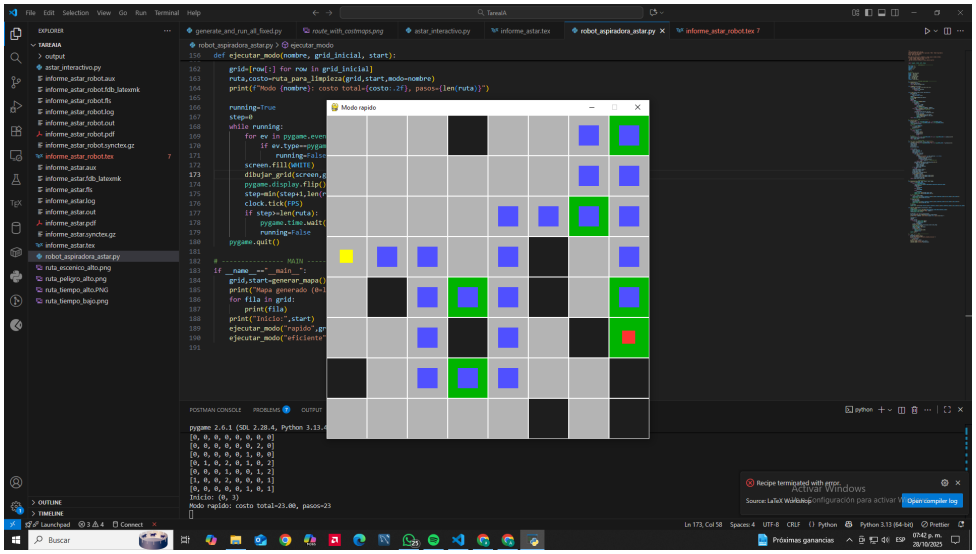

Figura 2: Ruta obtenida en modo rápido. Se prioriza la distancia total mínima.

5.2. Modo eficiente

En este modo, el robot penaliza los giros con un costo adicional de +1,5 por cada cambio de dirección. Esto hace que busque trayectorias más suaves y rectas, reduciendo el gasto energético aunque aumente la distancia recorrida.

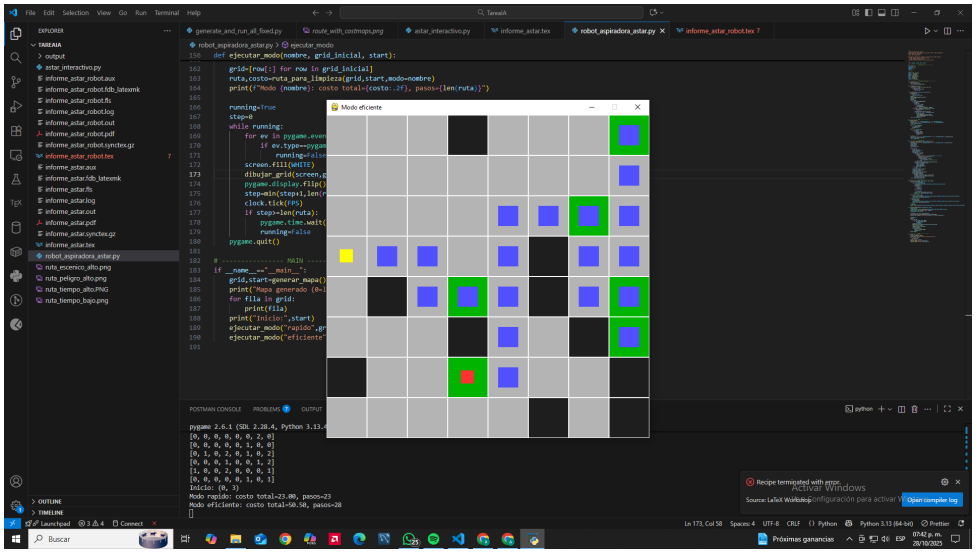

Figura 3: Ruta obtenida en modo eficiente. Se penalizan los giros y movimientos innecesarios.

6. Comparación entre modos

Modo	Costo total (A*)	Características observadas
Rápido	Menor tiempo (menos pasos)	Más giros, mayor gasto energético.
Eficiente	Mayor tiempo	Menos giros, movimientos más directos.

La diferencia entre los modos es visible cuando el mapa tiene varios obstáculos: el modo rápido toma atajos, mientras que el modo eficiente elige trayectorias más suaves aunque sean más largas.

7. Conclusiones

Este ejercicio demuestra cómo el algoritmo A* puede adaptarse a distintos criterios de optimización según la definición del modelo PEAS del agente.

- En el modo rápido, el agente prioriza el **tiempo**, encontrando rutas más cortas pero con más giros.
- En el modo eficiente, el agente prioriza la **energía**, penalizando los giros y evitando trayectorias abruptas.
- El modelo PEAS permite definir formalmente el comportamiento inteligente del agente, separando los componentes de desempeño, entorno, sensores y actuadores.

En síntesis, este proyecto ilustra el uso de A* como base de un comportamiento racional adaptable a distintos objetivos dentro de un entorno simulado.

Fin del documento.