Estructuras de Datos y Algoritmos I

Trabajo Práctico Final: Path Planning

Agustín López

Carrera: **Licenciatura en Ciencias de la Computación**

Facultad de Ciencias Exactas Ingeniería y Agrimensura

1 Actividad 1: Algoritmo de Navegación en Entorno Desconocido

En esta actividad, se presenta el desarrollo e implementación de un algoritmo de navegación para un robot en un entorno desconocido. El objetivo es explorar y mapear el entorno de manera eficiente utilizando estrategias para evitar obstáculos y explorar nuevas áreas de forma sistemática. A continuación, se detallan los pasos del algoritmo y su implementación.

Problem Descripción del Algoritmo
El algoritmo para la navegación del robot sigue los siguientes pasos:

1. **Paso 1:** El robot comienza su movimiento en una dirección diagonal hacia la posición objetivo. Este enfoque inicial permite que el robot avance rápidamente y cubra una gran parte del entorno, reduciendo el tiempo necesario para la búsqueda inicial y evitando movimientos repetitivos en áreas ya conocidas.

2. **Paso 2:** Cuando el robot llega a una posición en la que no puede avanzar más, se dirige hacia una celda no visitada previamente. Esto asegura que el robot explore nuevas áreas del mapa y evita el movimiento redundante en regiones ya descubiertas.

3. **Paso 3:** Si no existen celdas sin visitar, el robot retrocede por la ruta previamente recorrida. Esta estrategia permite al robot volver a celdas anteriores solo cuando es necesario, optimizando así la cobertura del área explorada y evitando caminos redundantes.

Este enfoque permite al robot explorar el entorno de manera eficiente y adaptativa, maximizando la cobertura y minimizando el tiempo de búsqueda.

Solution El código para la actividad 1 está organizado en varias partes clave, cada una con una función específica en la simulación del robot. A continuación, se detalla cada componente del código:

1. **Lectura de Datos:** El archivo `main.c` incluye funciones dedicadas a leer y validar un archivo de configuración que describe el mapa, la posición inicial del robot y el destino. La función `leer_archivo` carga estos datos en una estructura `FileData`, mientras que `validar_formato` se encarga de verificar que el archivo tenga el formato correcto y sea válido para la simulación.

2. **Creación del Mapa y Robot:** La función `mapa_crear` inicializa el mapa con las dimensiones y coordenadas proporcionadas. La función `robot_crear` inicializa el robot con su posición inicial y destino. Además, las funciones `robot_destruir` y `mapa_destruir` liberan la memoria asociada al robot y al mapa, respectivamente, asegurando una correcta gestión de los recursos.

3. **Simulación del Movimiento:** En `main.c`, el movimiento del robot hacia su destino se simula mediante la función `robot_ir_a_destino`. Esta función contiene la lógica necesaria para el movimiento del robot, considerando tanto la exploración de nuevas celdas como el retroceso cuando es necesario.

4. **Visualización y Depuración:** La función `mostrar_robot_mapa` imprime el estado actual del robot y el mapa, facilitando la depuración y el seguimiento del progreso del robot durante la simulación. Esta función es crucial para identificar y solucionar posibles problemas durante la ejecución del código.

Estructura de Archivos: La estructura de archivos se diseñó para reflejar una lógica orientada a objetos, donde cada archivo tiene una responsabilidad específica. La organización es la siguiente:

1. `main.c`: Contiene la función principal `main` que orquesta la simulación del robot.

2. `robot.c`: Contiene las funciones relacionadas con la creación, destrucción y movimiento de la estructura `Robot`.

3. `mapa.c`: Contiene las funciones relacionadas con la creación, destrucción y visualización de la estructura `Mapa`.

4. `simulacion.c`: Contiene las funciones que gestionan la interacción entre las estructuras `Robot` y `Mapa`.

Uso de Tabla Hash para Celdas Visitadas: Para registrar las celdas visitadas, se empleó una tabla hash que mapea las coordenadas de las celdas a un valor booleano. Esta estructura permite una búsqueda eficiente de celdas visitadas y evita el movimiento redundante en áreas ya exploradas. La elección de una tabla hash se debe a su eficiencia en la búsqueda y la inserción de elementos, proporcionando un acceso rápido a la información de celdas visitadas. Dado que no se requiere eliminar elementos en esta implementación, la tabla hash es una opción adecuada.

El enfoque adoptado en esta actividad proporciona una solución práctica y efectiva para la navegación en un entorno desconocido, equilibrando exploración y retroceso para cubrir el área de manera óptima.

Este informe cubre los aspectos esenciales del algoritmo de navegación del robot, mostrando cómo se implementa y se pone en práctica para enfrentar un entorno desconocido de manera eficiente.

2 Actividad 2: Algoritmo de Navegación en Entorno Parcialmente Desconocido

En esta actividad, se implementa un algoritmo de navegación para un robot en un entorno parcialmente desconocido utilizando una variante del algoritmo A*. El robot se desplaza en un mapa representado por una matriz de dimensiones $N \times M$ y debe llegar a un destino específico desde una posición inicial, evitando obstáculos y utilizando un sensor que proporciona información limitada sobre su entorno.

2.1 Descripción del Código

El código desarrollado se divide en varias funciones clave:

1. **Estructuras de Datos:** Se definen estructuras para representar el estado del robot (`_Robot`) y la información de cada celda del mapa (`CeldaInfo`). La estructura `_Nodo` se utiliza para manejar los nodos en la cola de prioridad, esencial para la implementación del algoritmo A*.

2. **Creación y Destrucción del Robot:** La función `robot_crear` inicializa la posición y el destino del robot, así como la memoria para la matriz que representa el mapa. La función `robot_destruir` libera la memoria utilizada por el robot, asegurando una correcta gestión de los recursos.

3. **Uso del Sensor:** La función `usar_sensor` simula la interacción con un sensor que proporciona las distancias a los obstáculos en las cuatro direcciones cardinales (arriba, abajo, izquierda, derecha). Esta función actualiza la matriz del mapa con la información obtenida, marcando las celdas exploradas y los obstáculos detectados.

4. **Algoritmo A*:** La función `calcular_ruta` implementa una variante del algoritmo A* para encontrar el camino más corto desde la posición actual del robot hasta el destino. Se emplea una cola de prioridad para explorar las celdas con el menor costo estimado y se actualiza la información de cada celda en función de los costos de movimiento y la heurística (distancia Manhattan al destino).

5. **Movimiento del Robot:** La función `ir_a_destino` reconstruye el camino óptimo encontrado por el algoritmo A* y mueve el robot a lo largo de ese camino, actualizando su posición en el mapa y gestionando las celdas visitadas.

El enfoque adoptado en esta actividad permite al robot navegar en un entorno parcialmente desconocido utilizando un algoritmo eficiente y adaptativo, maximizando la cobertura y minimizando el tiempo de búsqueda.