Navegación inteligente en robótica movil

Mateo Almeida Gómez

Robótica

Jaime Arango Castro

Agenda

Revisaremos la robótica movil desde la importancia de la navegación y demás conceptos teóricos relacionados a esta.

- Navegación reactiva
- Navegación basada en mapas

Introducción a la Navegación Robótica

¿Qué es la Navegación?

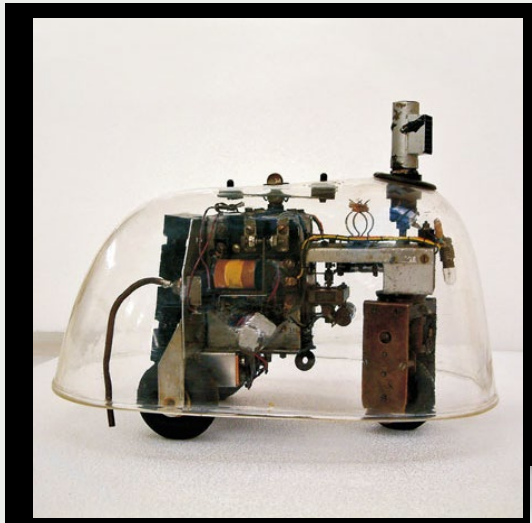
Permite a los robots moverse autónomamente de un un punto A a un punto B, punto B, evitando obstáculos y alcanzando alcanzando objetivos.

¿Por Qué es Importante?

Es fundamental para aplicaciones como exploración, entrega, servicio y operaciones autónomas en entornos entornos dinámicos.

Contexto del Curso Curso

Esta presentación intenta integrar la navegación con fundamentos previous del libro de Peter Corke.

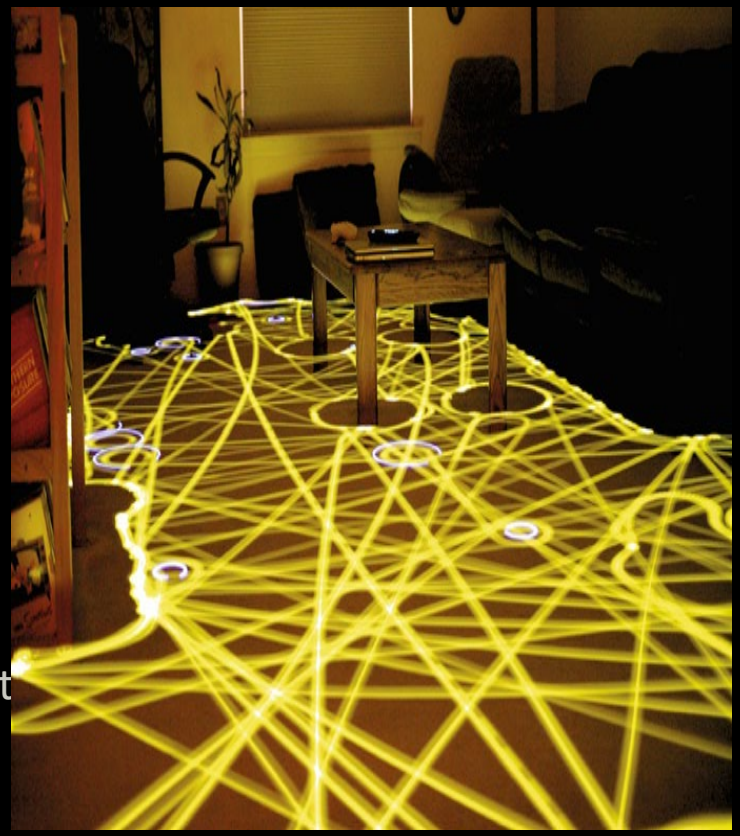


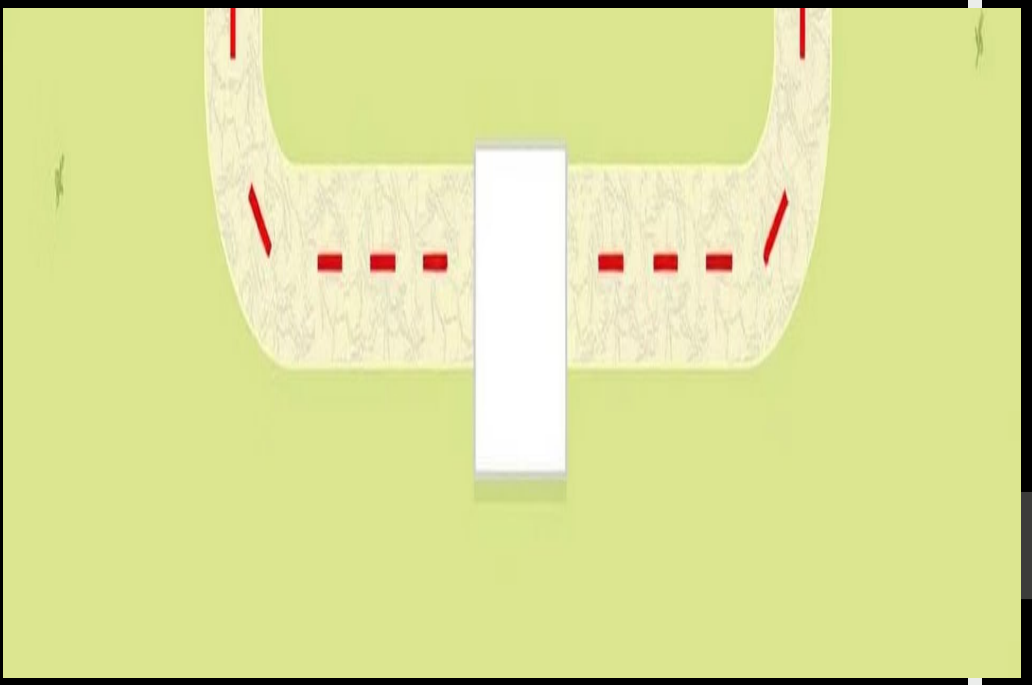
Tipos de Navegación

Navegación Reactiva (Local)

Responde a la información del sensor inmediato para evitar colisiones sin conocimiento global del entorno.

- Basada en reglas simples.
- Ideal para entornos dinámicos.
- Observamos en pantalla el lapso de tiempo en el cual se encuentra enmergido el robot





Navegación Reactiva: Braitenberg y Bug2

Vehículos Braitenberg

Conexión directa sensor-actuador; el robot responde a a estímulos simples sin planificación compleja

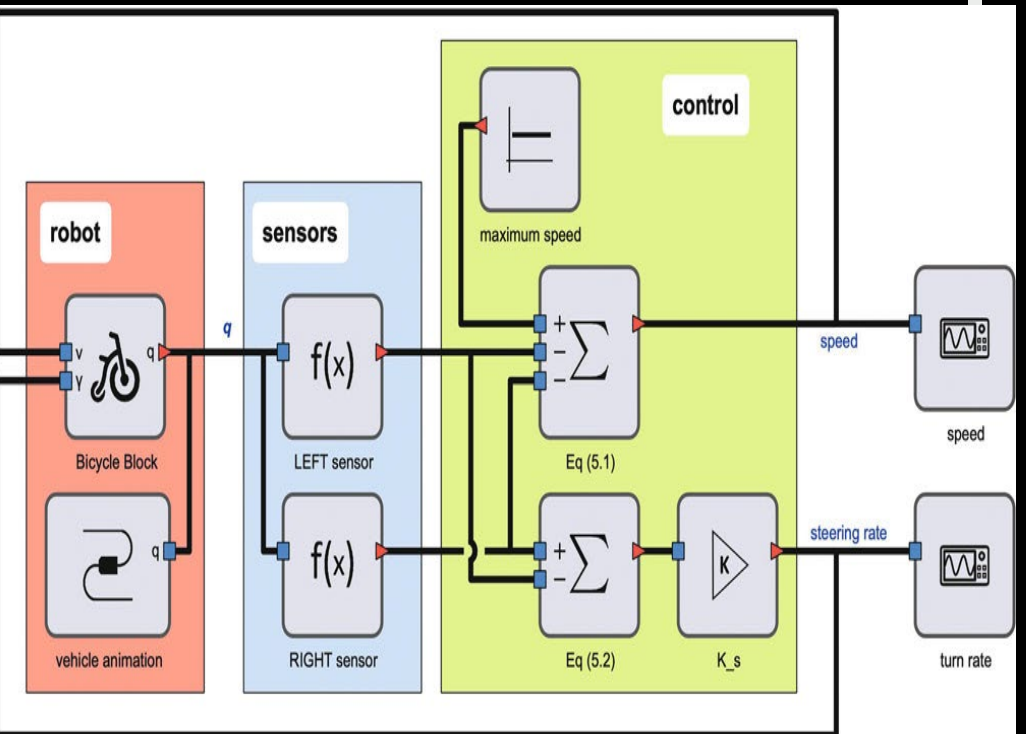
Su conexión sensorial actuadora actuadora es un plan implicito implicito

Algoritmo Bug2

Combina avance directo hacia la meta con rodeo de obstáculos. Sigue la línea m y se adhiere a contornos.

Ventajas y Desventajas

Rápida y simple, pero subóptima y sin previsión, pudiendo pudiendo quedar atrapado.



Braitenberg

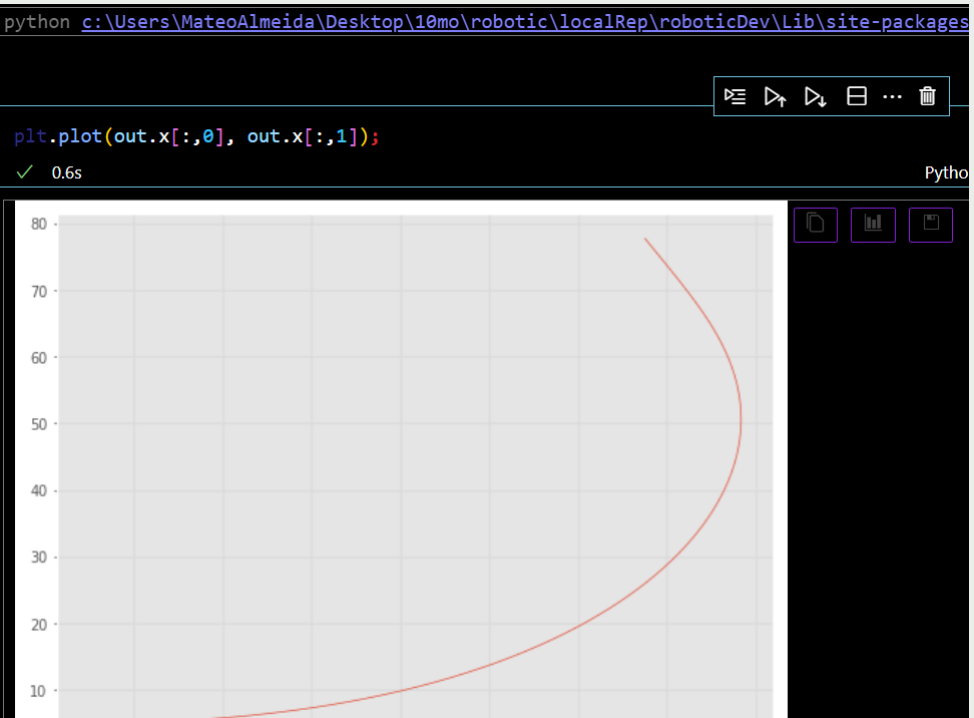
```python
def sensorfield(x, y):
  xc, yc = (60, 90)
  return 200 / ((x - xc) ** 2 + (y - yc) ** 2 + 200)
```

Conexión directa sensor-actuador; el robot responde a estímulos simples sin planificación compleja

Su conexión sensorial actuadora es un plan implicito

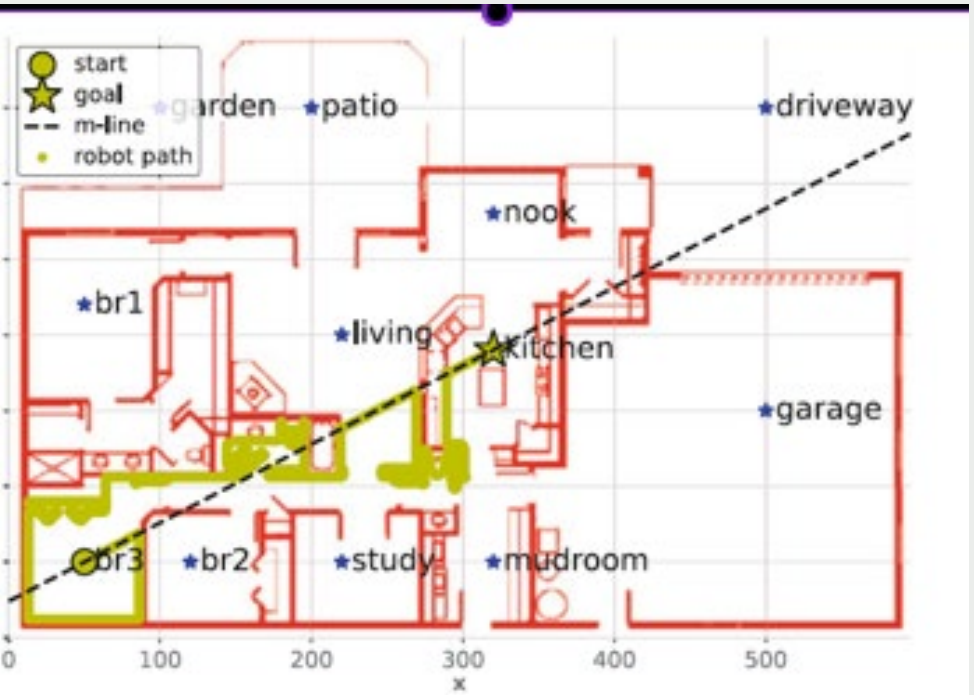
Esta función representa un campo escalar generado por un sensor (o una fuente) que está ubicado en el punto (60, 90). El campo disminuye con la distancia desde ese punto.

De esta forma, el robot tendrá su velocidad y angulo de giro, definidos



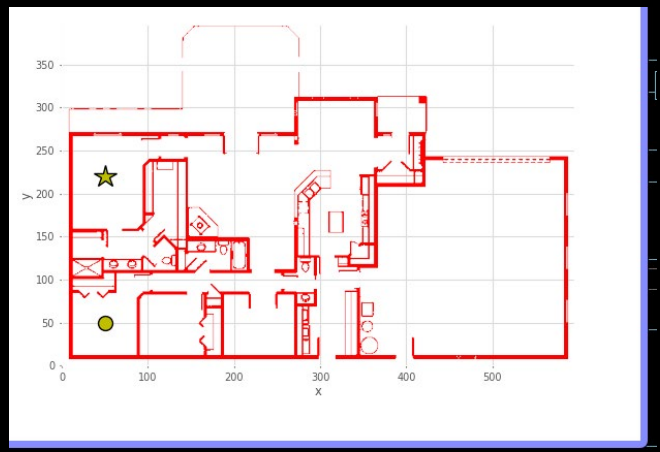
La gráfica muestra la trayectoria del vehículo Braitenberg en el plano XY, indicando cómo se mueve desde su posición inicial hacia el máximo del campo sensorial

BUGs algorithms



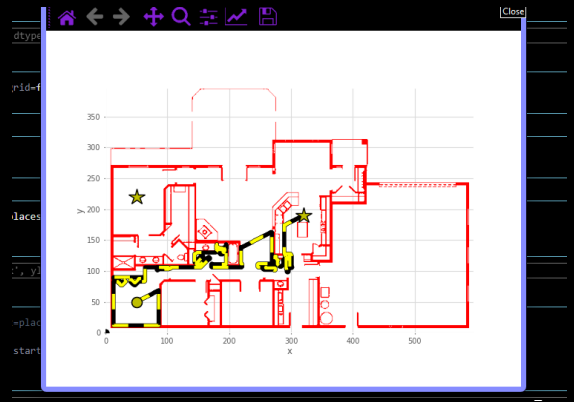
SIMPLE AUTOMATA

A diferencia de los vehículos de Braitenberg, los "bugs" **tienen una memoria mínima** y una **lógica de control más compleja**, por ejemplo, máquinas de estados.

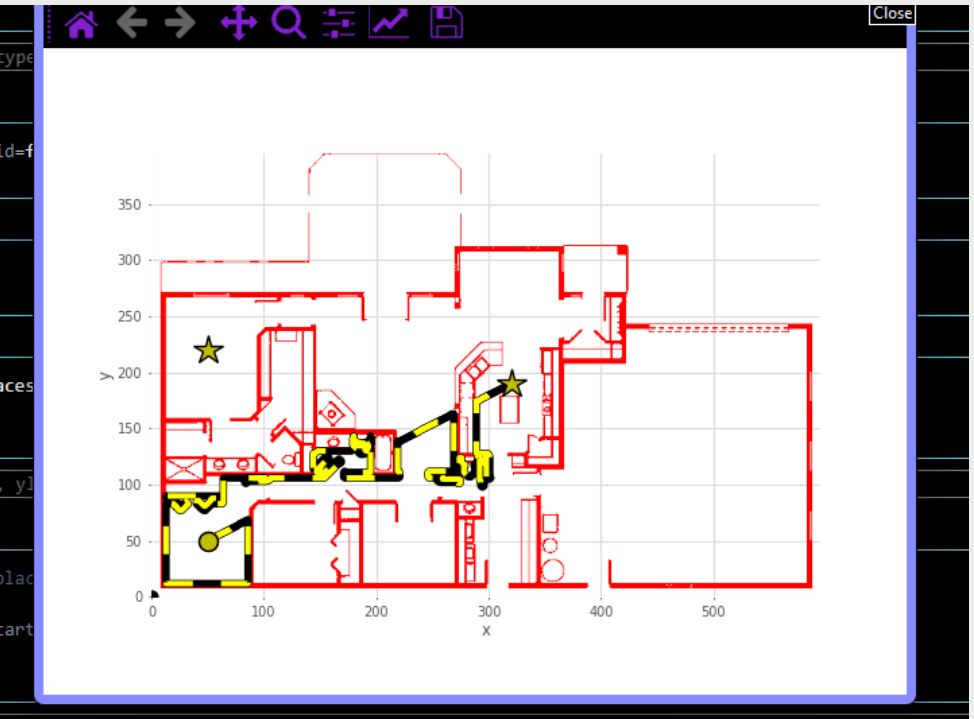


El proceso es sencillo:

1. Dibuja una línea recta (llamada "m-line") desde el punto de inicio al objetivo.
2. El robot se intenta mover a lo largo de la línea
3. Si se encuentra un obstaculo, lo rodea por la derecha, siempre mantiendose cerca del borde
4. Cuando Vuelve a intersectar la m-line, en un punto más cercano al objetivo que el anterior, regresa a esta y sigue avanbzado hacia el objetivo.
5. Repite el proceso hasta llegar al destino.



BUGs algorithms



SIMPLE AUTOMATA

De esta manera, vemos ;

El algoritmo no usa un mapa global, por lo que:

Puede tomar rutas subóptimas como pasar por armarios o baños

Toma decisiones locales sin saber si son contundentes a largo plazo

Esto es bueno puesto; que entrega simplicidad y no necesita un mapa completo pero también es malo, debido a que puede dar vueltas innecesarias y no garantiza la mejor ruta

Bug2 ilustra cómo un robot reactivo puede llegar a su meta sin un mapa previo, pero a costa de rutas más largas y menos eficientes. Este algoritmo muestra:

Cómo diseñar comportamientos sencillos basados únicamente en lecturas de sensores.

Las limitaciones que surgen al carecer de visión global del entorno.

El funcionamiento de las estrategias de navegación puramente locales.

Cinemática de Robots Móviles

Esenciales para la navegación reactiva y la ejecución de movimientos.
Por ejemplo: diferencial y Ackermann.

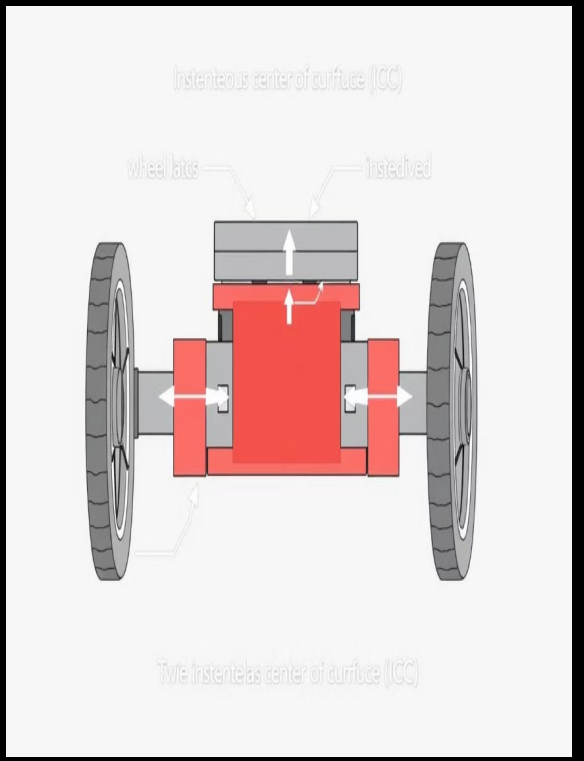
Movimiento en 2D

La navegación se basa en la comprensión del movimiento del robot en un plano. Las ecuaciones ecuaciones cinemáticas definen la velocidad y posición.

Los modelos cinemáticos, como los diferenciales para robots de dos ruedas y Ackermann para vehículos tipo coche son la base para ejecutar los movimientos planificados por los algoritmos de navegación.

Diferencial: presenta dos ruedas independientes y velocidad lineal.

Ackermann : Rueda delantera direccional y traseras motrices. Presenta un radio de giro.



Vemos entonces que Bug2 y Braitenberg, generarn sus trayectorias basándose en (x,y, theta) definidos por cada uno de estos modelos.

Finalmente, es clave tener presente que sin cinemática correcta, no se puede traducir la señal de control en movimiento real.

Rotaciones 2D, 3D y cuaterniones

ROTACIÓN 2D:

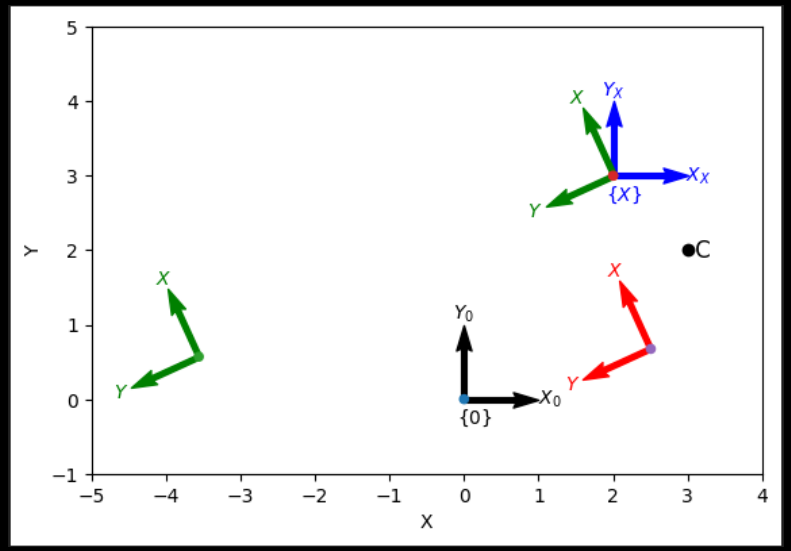
```
from sympy import Symbol, Matrix, cos, sin

theta = Symbol('theta')
R = Matrix([
    [cos(theta), -sin(theta)],
    [sin(theta), cos(theta)]
])
```

✓ 0.0s

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

Gira un vector en el plano, conservando la longitud

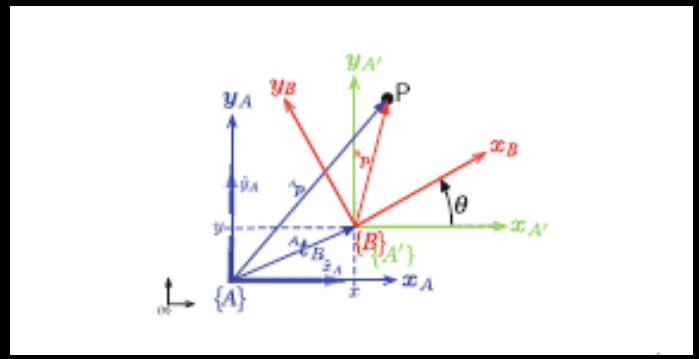


Rotaciones 2D, 3D y cuaterniones

TRANSFORMACIÓN HOMOGÉNEA

$$\begin{pmatrix} {}^{A}x \\ {}^{A}y \\ {}^{A}z \\ 1 \end{pmatrix} = \begin{pmatrix} {}^{A}\mathbf{R}_{B} & {}^{A}\mathbf{t}_{B} \\ \mathbf{0}_{1\times 3} & 1 \end{pmatrix} \begin{pmatrix} {}^{B}x \\ {}^{B}y \\ {}^{B}z \\ 1 \end{pmatrix}$$

Combina rotación y traslación en un solo paso



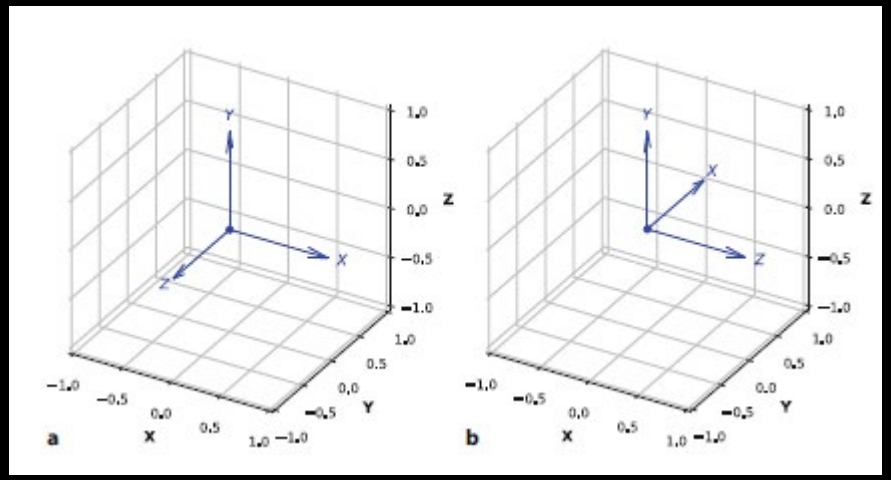
Rotaciones 2D, 3D y cuaterniones

ROTACIÓN 3D

$$\begin{pmatrix} {}^{A}p_x \\ {}^{A}p_y \\ {}^{A}p_z \end{pmatrix} = {}^{A}\mathbf{R}_B \begin{pmatrix} {}^{B}p_x \\ {}^{B}p_y \\ {}^{B}p_z \end{pmatrix}$$

$$\mathbf{R}_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix}$$

$$\mathbf{R}_y(\theta) = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}$$

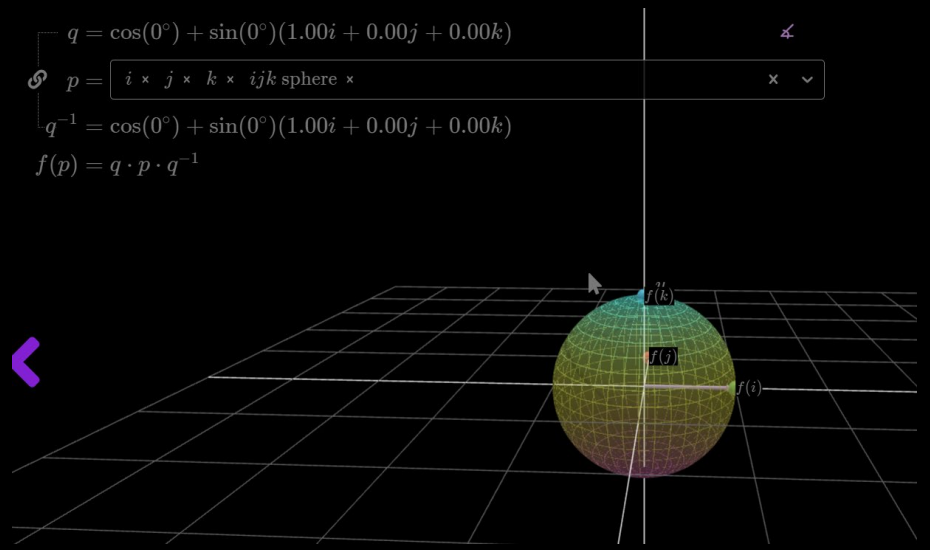


- “Ortogonal 3×3, det=1.
- Prevenir bloqueo de cardán

Rotaciones 2D, 3D y cuaterniones

CUATERNIONES

$$\breve{q} = s + v_x i + v_y j + v_z k \in \mathbb{H}$$



•Presenta una formula de rotación:

$$p' = q\,p\,q^{-1}.$$

– Para evitar el gimbal lock (al usar tres ejes de rotación secuenciales) usamos **Cuaterniones** o **matrices de rotación**: Estas son representaciones que no tienen esta singularidad y permiten una interpolación suave entre orientaciones (SLERP = interporlación esférica de cuaterniones)

Navegación Basada en Mapas (Global)

Utiliza un mapa preexistente o construido para planificar rutas

rutas óptimas.

Considerando :

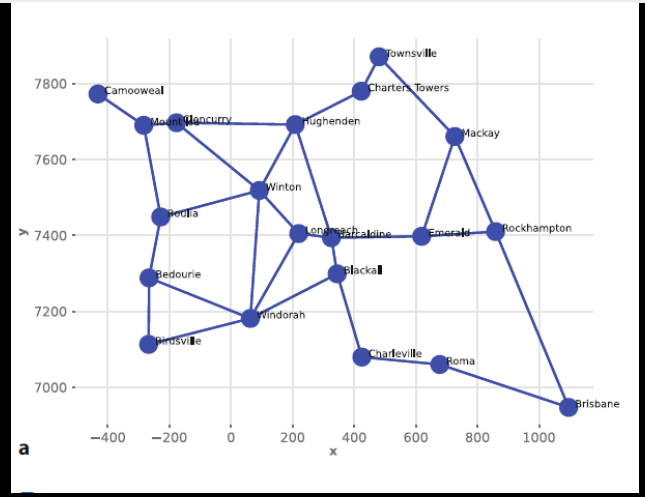
La más corta

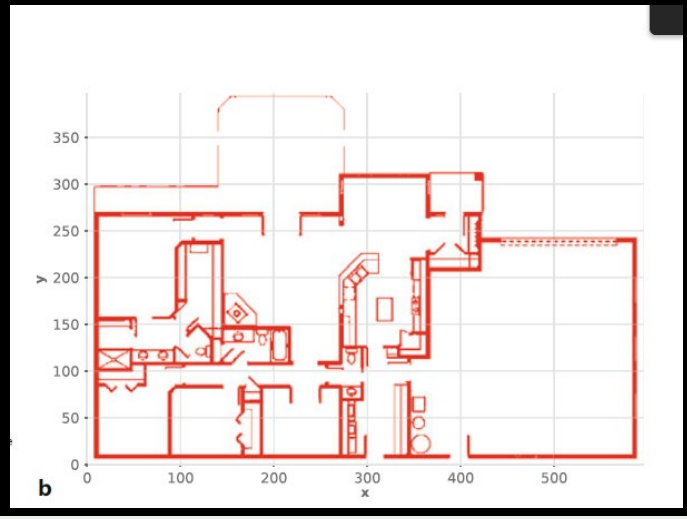
La más rápida

LA más Segura

Podemos encontrar:

- Grafos: Nodos y aristas.
- Celdas: Cuadrículas de ocupación.
- PRM: Mapas de ruta probabilísticos.





Navegación Basada en Grafos

1

Representación

El entorno se modela como nodos (lugares) y aristas (caminos). Los costos
Los costos asociados indican distancia o tiempo. También es importante
importante denotar que el uso de transformaciones homogéneas y
homogéneas y cuaterniones permite definer marcos de refencia locales
locales para la navegación. E.g, cuando se transforman las coordenadas
coordenadas del grafo a un marco del robot usando SE(3)

2

BFS

Búsqueda en anchura: encuentra la ruta más corta en número de pasos.

3

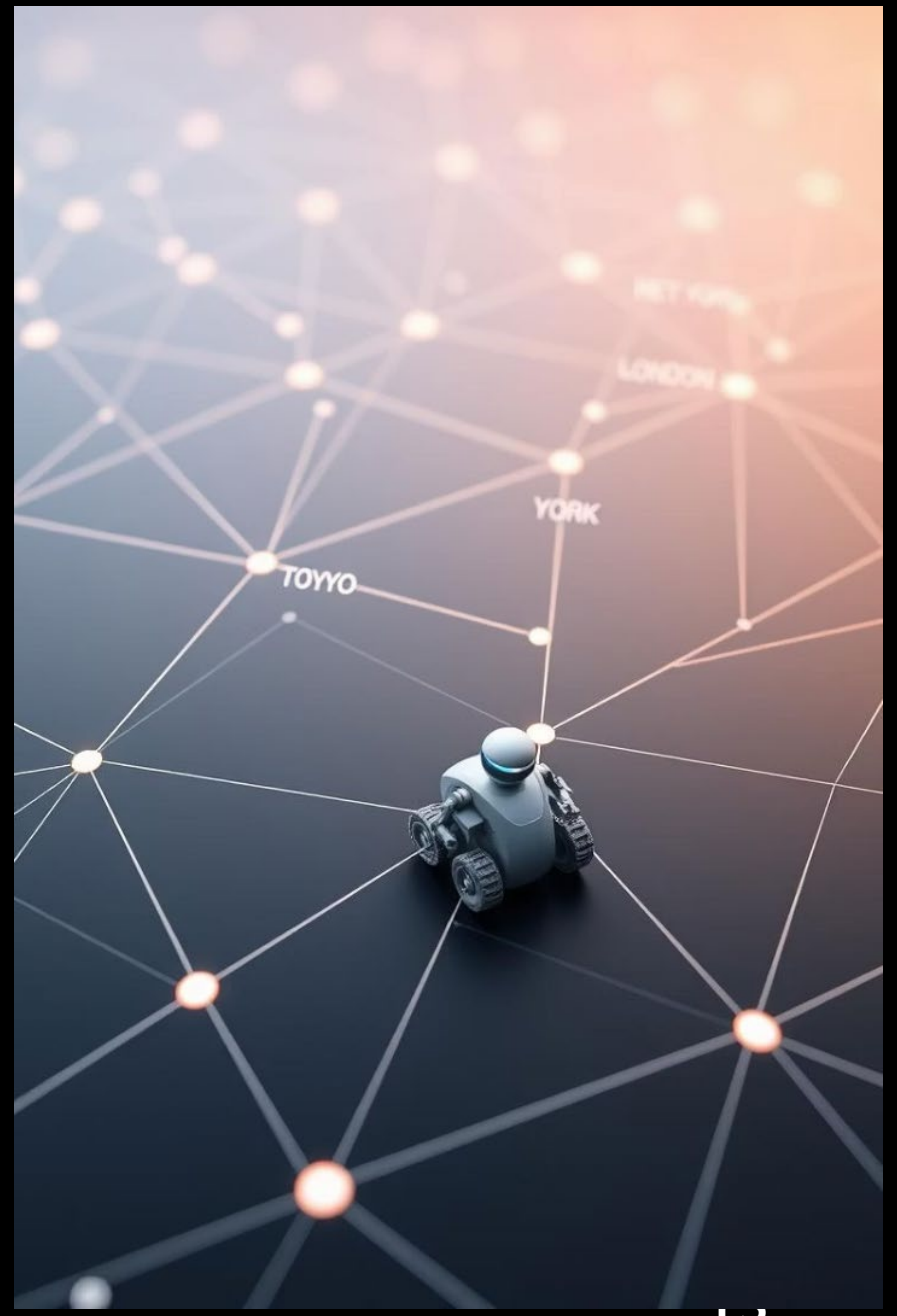
UCS

Búsqueda de costo uniforme: garantiza la ruta de menor costo total.

4

A*

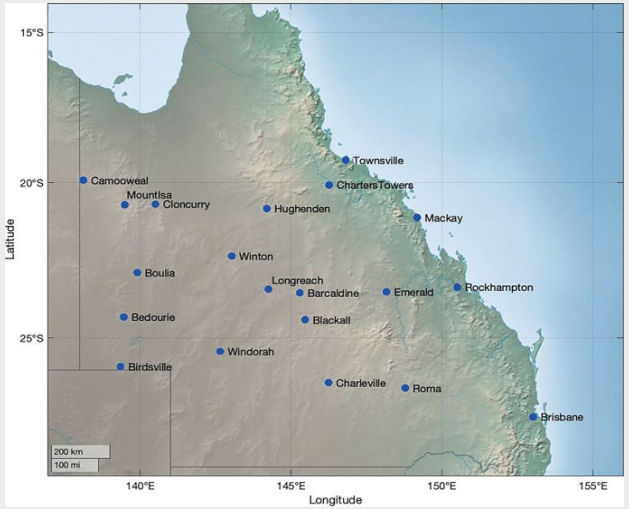
Búsqueda A estrella: utiliza heurística para encontrar la ruta óptima de
de forma más eficiente.



Navegación Basada en Grafos

Representación

Vemos también que, los vertices de los grafos representan lugares definidos por coordenadas espaciales.



BFS Búsqueda en anchura: encuentra la ruta más corta en número de pasos.

UCS Búsqueda de costo uniforme: garantiza la ruta de menor costo total.

A* Búsqueda A estrella: utiliza heurística para encontrar la ruta óptima óptima de forma más eficiente.

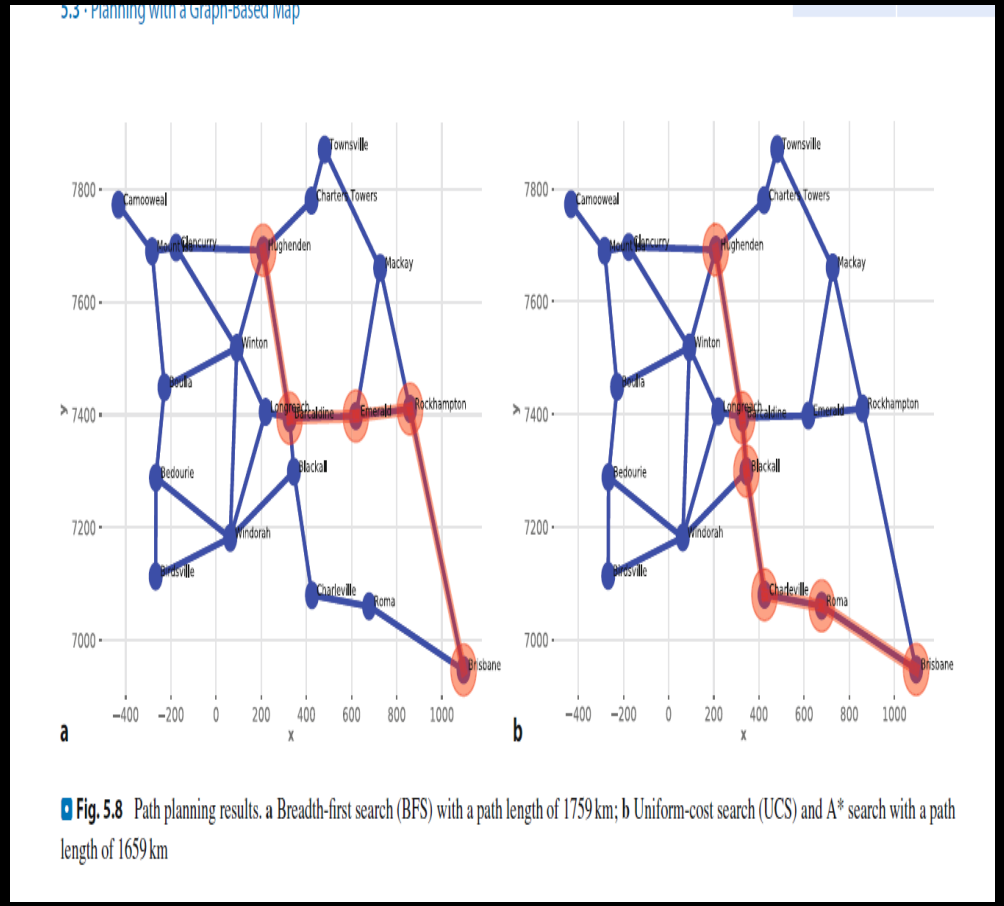


Fig. 5.8 Path planning results. a Breadth-first search (BFS) with a path length of 1759 km; b Uniform-cost search (UCS) and A* search with a path length of 1659 km

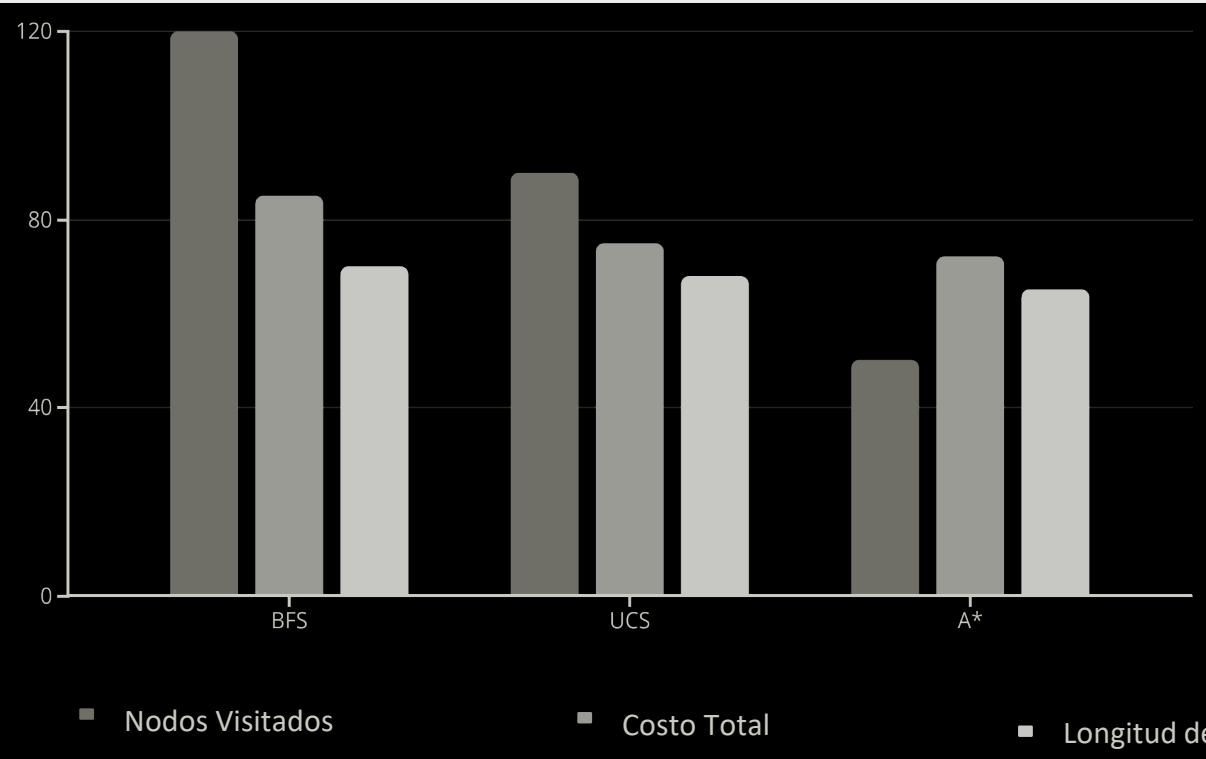
Algoritmos
para busqueda de rutas.

1. **Breadth-First Search (BFS)**
Prioriza caminos con **menos nodos**.
No considera la distancia real.
Devuelve una ruta rápida pero no óptima.

2. **Uniform-Cost Search (UCS)**
Expande el nodo con menor **costo acumulado (g(v))**.
Devuelve la ruta de **menor distancia real**.
Usa una cola de prioridad y explora más nodos que BFS.

3. **A* Search**
Usa: f(v) = g(v) + h*(v)
g(v): costo acumulado.
h*(v): heurística admisible (ej: distancia Euclidiana).
Explora menos nodos que UCS.
Más eficiente si la heurística es buena.

Simulación y comparativa de algoritmos



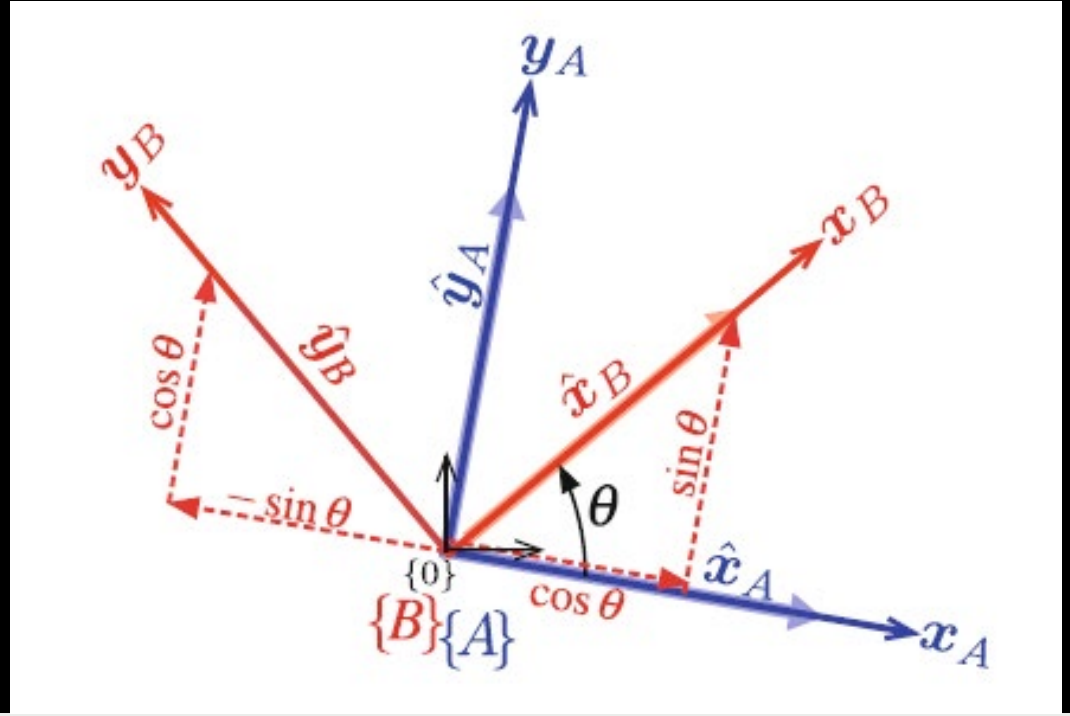
Vemos que UCS y A* pueden considerer tiempo como costo (distancia / velocidad)

También está presente la planeacion de trayectorias, mediante interpolación entre nodos usando polinomios cúbicos/quínticos.

Es posible realizar un seguimiento temporal con controladores , mediante la velocidad y aceleración, e.g de motores.

La simulación muestra cómo A* visita menos nodos y encuentra caminos óptimos, superando a BFS y UCS en eficiencia para trayectorias complejas.

Posición y orientación



Tanto para 2D como para 3D, representamos marcos de coordenadas usando transformaciones que combinan **rotación** y **traslación**.

En el plano, usamos **matrices de rotación 2D**, mientras que en 3D empleamos **matrices de rotación 3x3**, **ángulos de Euler** o **cuaterniones** para describir la orientación de un robot o sensor.

Posición y orientación

Los vértices de los grafos representan lugares definidos por coordenadas espaciales

El uso de transformaciones homogéneas y cuaterniones permite definir marcos de referencia locales para navegación

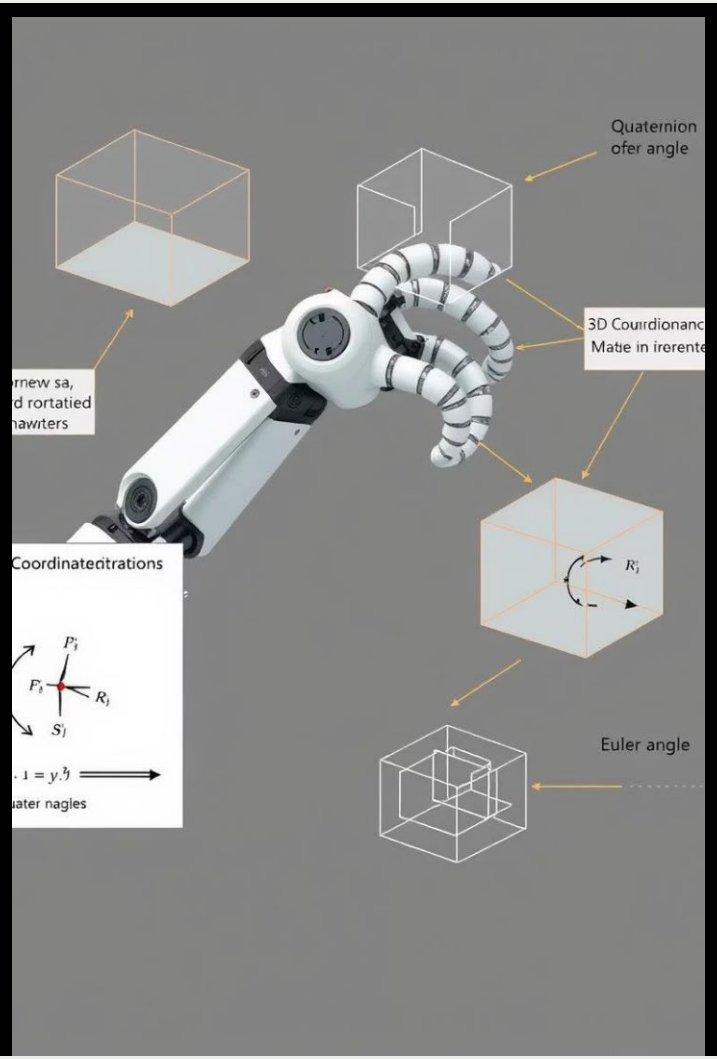
Ejemplo: Transformar coordenadas del grafo a un marco del robot usando SE(2) o SE(3)

Estas herramientas nos permiten :

*Cambiar entre marcos de referencia, por ejemplo de la base del robot a la herramienta

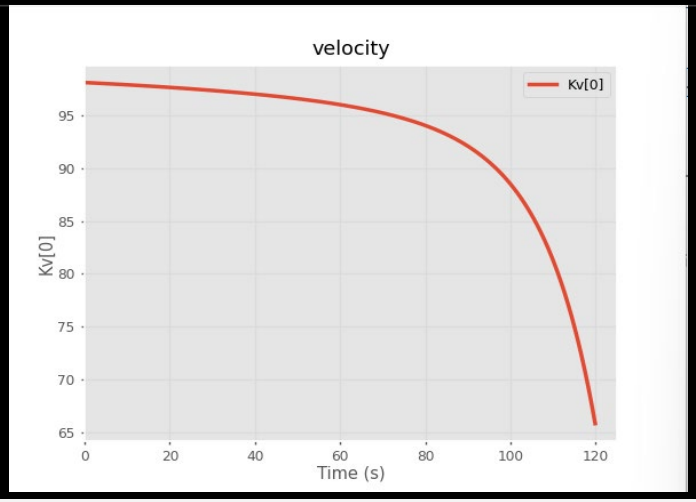
* Integrar la orientación del robot en sistemas de navegación

* Facilita la representación espacial en algoritmos de mapeo y localización

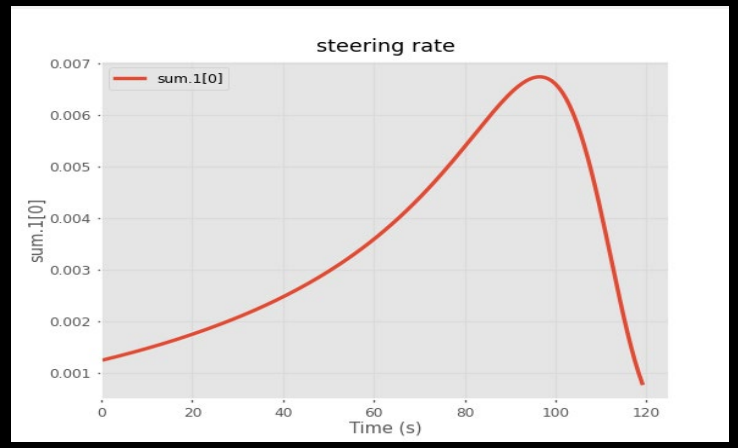


Transformaciones Homogéneas	Representan posición y orientación orientación en 3D.	Esenciales para la localización del robot y la transformación de coordenadas de sensores.
Cuaterniones vs. Euler	Métodos para representar rotaciones.	Los cuaterniones evitan el bloqueo de cardán, crucial para la orientación estable del robot.
Trayectorias	Definen el camino temporal del del robot (polinomios, interpolación).	La planificación de trayectoria asegura movimientos suaves y seguros.
Velocidad y Aceleración	Dinámica del movimiento del robot.	Limitaciones de velocidad y aceleración deben considerarse para movimientos controlados.

Posición, orientación y tiempo



La gráfica de velocity (velocidad) muestra cómo varía la velocidad del vehículo Braitenberg a lo largo del tiempo durante la simulación.



La gráfica "steering rate" muestra el comportamiento de la tasa de giro del vehículo Braitenberg (γ) en función del tiempo durante la simulación. La tasa de giro (γ) es la velocidad angular con la que el vehículo ajusta su dirección de movimiento para alinearse con el gradiente del campo sensorial.

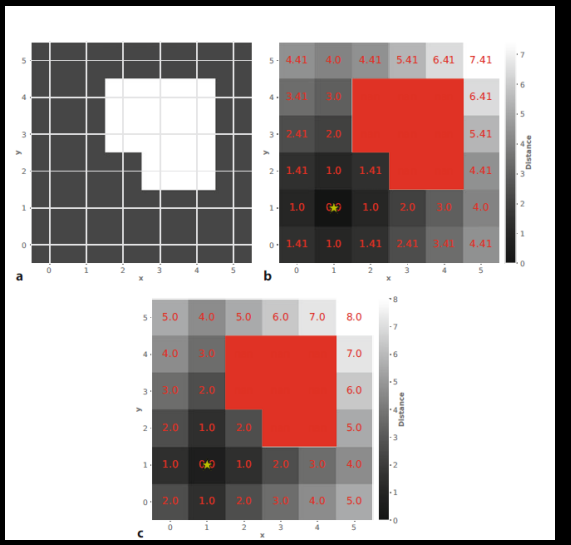
Posición, orientación y tiempo

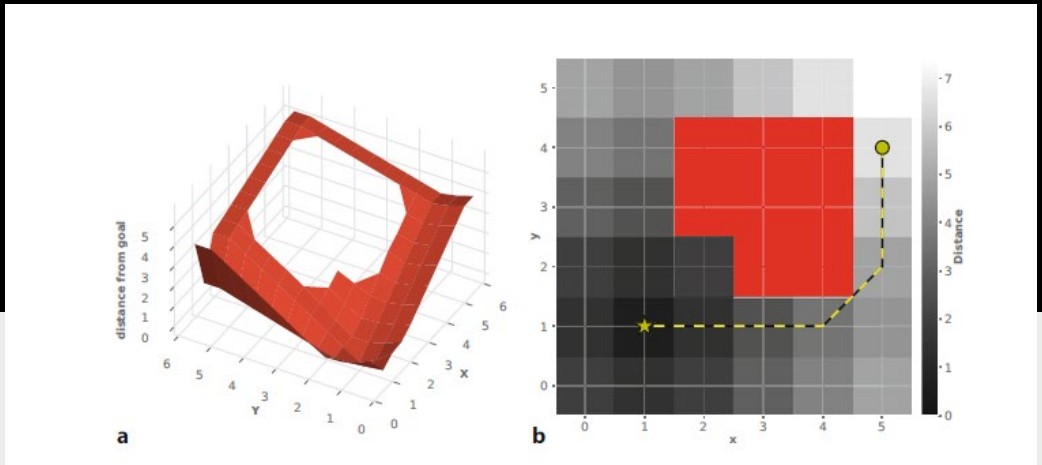
Celdas de ocupación y hojas de ruta (PRM)

Celdas de Ocupación

El entorno se modela como una cuadrícula. Cada celda indica si está libre (0) u ocupada (1).

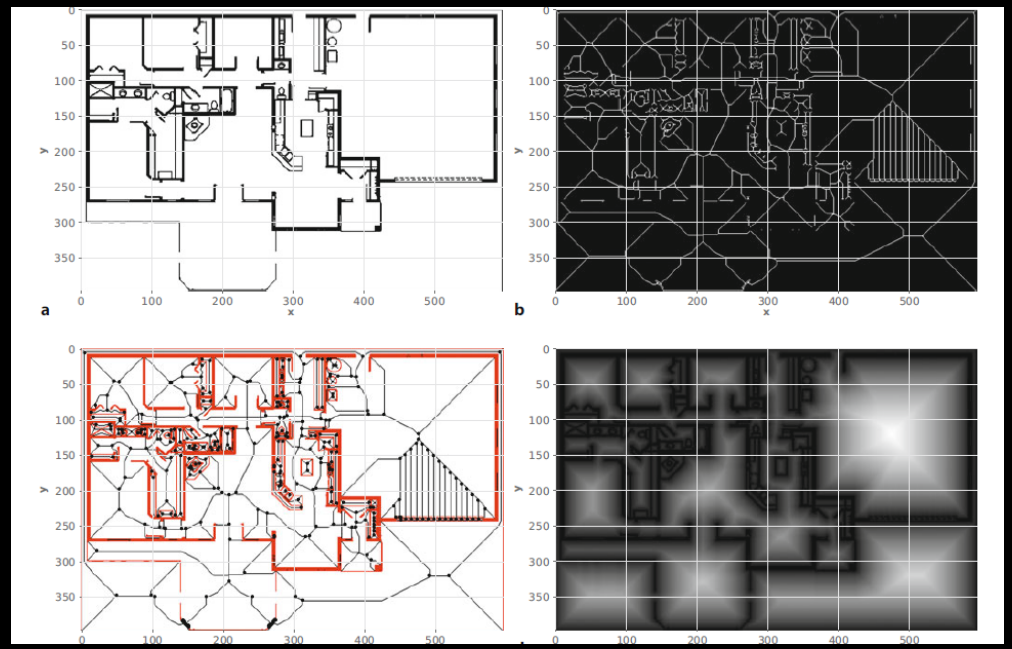
- La transformada de distancia (Wavefront/ brushfire) genera un campo de costes.
- El robot sigue el gradiente descendente hacia el objetivo.





Hojas de ruta

La calidad depende del muestreo.



■ Idea General

- Separar en dos fases:
  1. Planificación: construir una red de caminos libres (grafo).
  2. Consulta: buscar una ruta entre un punto de inicio y uno final.

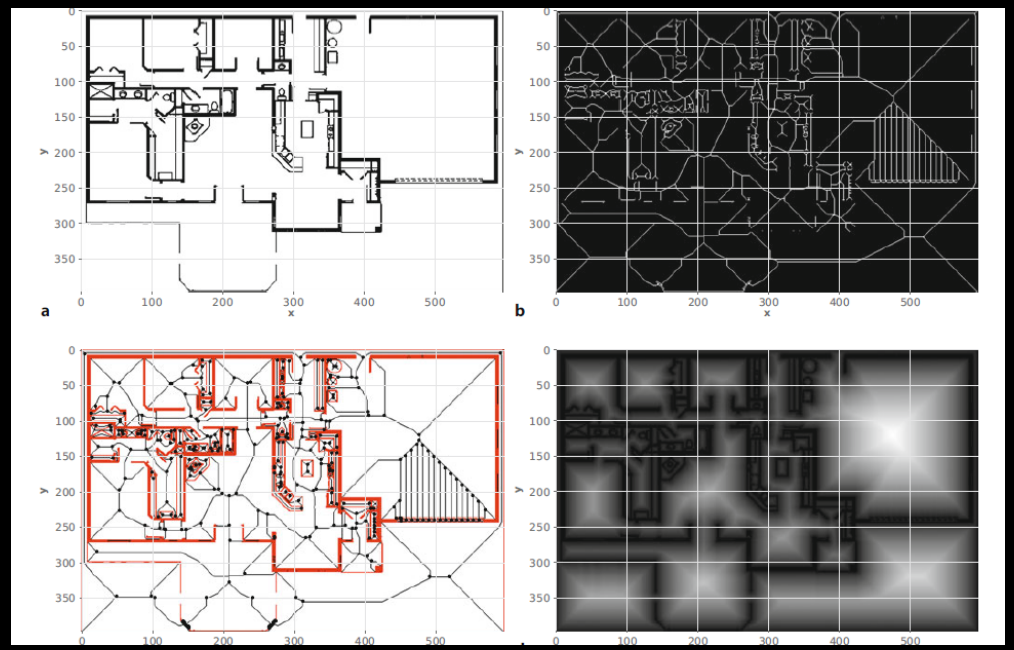
Por ejemplo, para los casos de Sekelton y Voronoi.

- Se genera una red de caminos usando:
  - Skeletonization: reduce el espacio libre a una red de píxeles centrales.
  - Voronoi Diagram: distancias a obstáculos generan líneas centrales equidistantes.

  - Finalmente se extraen intersecciones (junctions) y se procede a crear el grafo.

PRM

La calidad depende del muestreo.



- Eficiente para consultas rápidas de rutas.

Se realiza un muestreo aleatorio de puntos accesibles, conectados por líneas libres de obstáculos y de allí, se crea un grafo embebido.

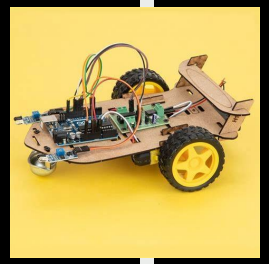
Sin embargo, es importante tener en cuenta algunas consideraciones ;

- Puntos aleatorios no cuban todo el mapa: esto requiere realizar múltiples intentos
- Rutas pueden ser menos suaves.
- Puede fallar en pasajes estrechos.

Comparativa de Estrategias de Navegación

MÉTODO	VENTAJAS	DESVENTAJAS
• REACTIVA	• RAPIDA Y SIMPLE	• SUBÓTIMA, SIN PREVISIÓN
• GRAFOS	• OPTIMA, HEURÍSTICA	• COSTOSA COMPUTACIONALMENTE
• CELDAS	• GLOBAL, PRECISA	• MUY LENTA EN MAPAS GRANDES
• PRM	• EFICIENTE EN CONSULTA	• Dependiente del muestreo

Comparativa de robots móviles

	Tipo de Robot	Modelo Cinemático	Navegación Típica	Compatibilidad Estrategias
	**Diferencial** (2 ruedas)	Simétrico, puede girar en su eje	Interiores, entornos planos	✓ Reactiva (Bug), ✓ Grafos, ✓ Celdas
	**Ackermann** (vehículos tipo auto)	Radio de giro limitado	Exteriores, caminos definidos	✓ Grafos, ✓ Celdas, ✗ Reactiva pura
	**Omnidireccional** (Mecanum, holonómico)	Movimiento libre en XY	Espacios reducidos, laboratorios	✓ Reactiva, ✓ PRM
	**Aéreo (Drones)**	6 DOF, control por orientación 3D	Navegación 3D, SLAM aéreo	✓ PRM, ✓ Grafos 3D, ✗ Celdas 2D

La navegación efectiva requiere la combinación de percepción, planificación y ejecución

Cada modelo tiene su aplicación ideal según contexto

El dominio de los conceptos de orientación, cinemática y tiempo permite adaptar soluciones a distintos entornos.