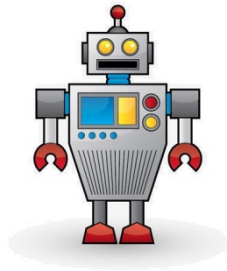
Escuela de Ciencias de la Computación e Informática



Planificación de Trayectoria

CI-2657 Robótica

M.Sc. Kryscia Ramírez Benavides

LEGO mindstorms

LEGO DIGITAL DESIGNER

LEJOS Java for LEGO Mindstorms

Problemas de Navegación de los Robots

- ¿Dónde estoy?
  - Localización.
- ¿Dónde he estado?
  - Mapa de decisiones.
- ¿A dónde voy?
  - Planificación de misiones.
- ¿Cuál es la mejor manera de llegar?
  - Planificación de trayectoria.

Trayectoria

- Una secuencia de configuraciones del robot desde una configuración inicial a una configuración final.
- Debe ser continua.
- Debe estar en un orden específico.
- Por lo general, algunos de los valores de costo se aplican al robot para cambiar configuraciones.

Planificación de Trayectoria

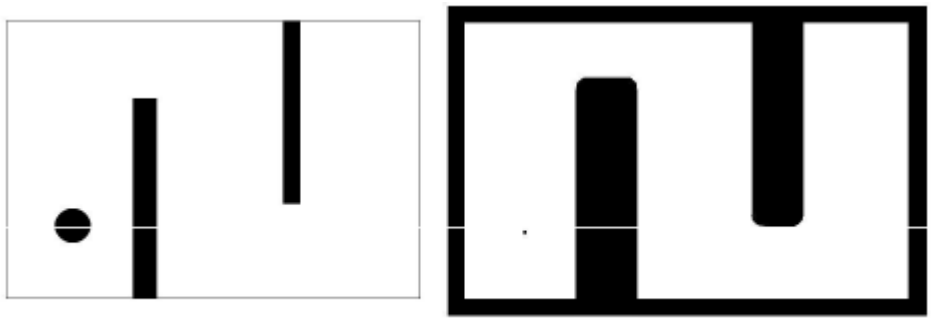
- También se llama descubrimiento del camino o hallazgo del camino.
- Por lo general, se prefiere un camino libre de colisiones con el mínimo costo.
- El costo puede ser distancia, tiempo, etc.
- Tanto el problema de optimización y búsqueda.

Complejidad de la Planificación de Trayectoria

- En el espacio de trabajo 3D, la búsqueda de una solución exacta es NP-HARD. [Xavier92]
- Planificación de trayectoria es PSPACE-HARD. [Reif79]
- La complejidad aumenta exponencialmente con:
  - Número de DOF. [Canny88]
  - Número de agentes.

Configuración Espacial

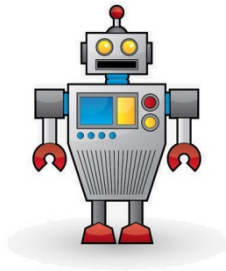
- También se llama *CSpace*.
- El conjunto de todas las configuraciones posibles de un robot.
- El número mínimo de parámetros necesarios para especificar completamente la configuración del objeto.



Espacio de Trabajo Configuración espacial

Algoritmos

- Algoritmos simples.
  - Algoritmos *Bug* (pulga).
  - Métodos Wavefront.
  - Planes de trabajo.
    - Mapas de Meadow.
    - Gráficos de visibilidad.
    - Gráficos generalizados de Voronoi.
    - Descomposición de celdas.
- Algoritmos basados en muestreo.
  - Planes de trabajo probabilísticos.
  - Explorando rápidamente un árbol aleatorio.
  - Variantes basadas en:
    - Estrategias de muestreo.
    - Estrategias de conexión.
    - Algoritmos perezosos.
    - Post-procesamiento.



Algoritmos Simples

Algoritmos *Bug*

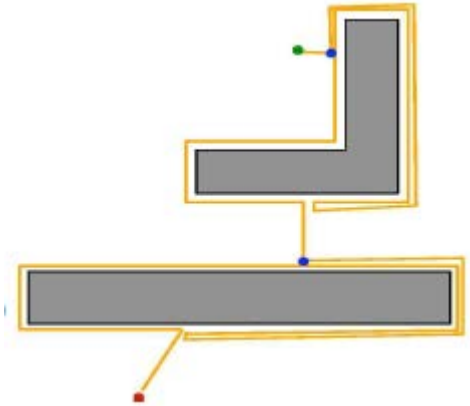
- Muchos algoritmos de planificación asumen el conocimiento global.
- Los algoritmos *Bug* asumen sólo el conocimiento local del medio ambiente y un objetivo global.
- Los comportamientos de errores son simples:
  - Seguir un muro (derecha o izquierda).
  - Moverse en línea recta hacia la meta.
- Bug 1 y Bug 2 asumen sensores esencialmente táctiles.
- Ofertas Tangent Bug con detección de distancia finita.

Algoritmos *Bug* (cont.)

- Dirección conocida del objetivo.
- El robot puede medir la distancia entre dos puntos x y y.
- Utiliza sensores locales para paredes/obstáculos y codificadores.
- Suposición razonable del mundo:
  - Un número finito de obstáculos en cualquier área finita.
  - Una línea cruzará un obstáculo un número finito de veces.
  - Espacio de trabajo está delimitado.

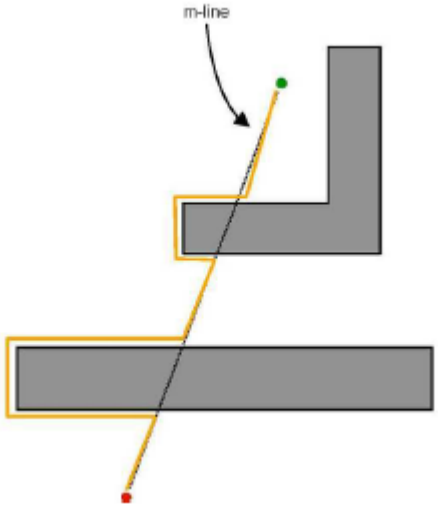
Algoritmo *Bug* 1

- Dirigirse hacia la meta.
- Si hay un obstáculo en el camino, circunnavegar y recordar lo cerca que se llega a la meta.
- Volver al punto más cercano (por la pared de seguimiento) y continuar.



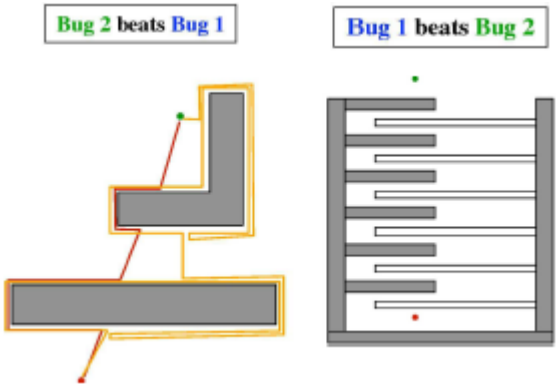
Algoritmo *Bug 2*

- Dirigirse hacia la meta en *m-line*.
- Si hay un obstáculo en el camino, seguir hasta encontrar la *m-line* de nuevo más cerca de la meta.
- Dejar el obstáculo y continuar hacia la meta.



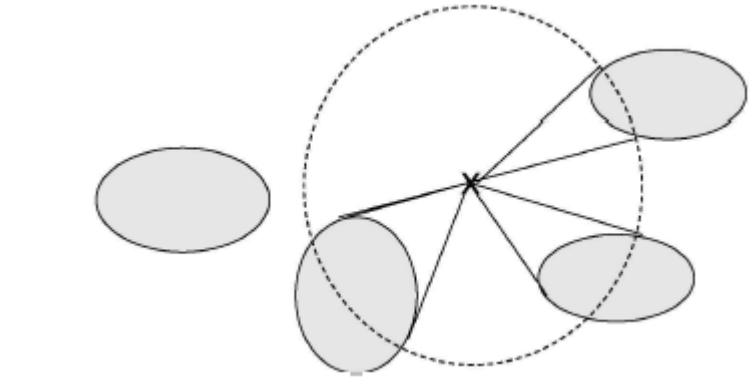
Algoritmo *Bug* 1 vs. Algoritmo *Bug* 2

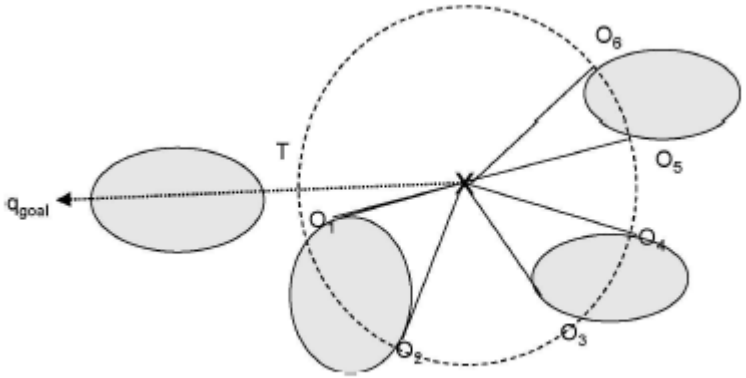
- *Bug* 1 es un algoritmo de búsqueda exhaustiva.
  - Examina todas las opciones antes de comprometerse.
- *Bug* 2 es un algoritmo voraz.
  - Toma la primera cosa que ve mejor.
- En muchos casos, *Bug* 2 supera a *Bug* 1, pero *Bug* 1 tiene un rendimiento global más predecible.



Algoritmo *Tangent Bug*

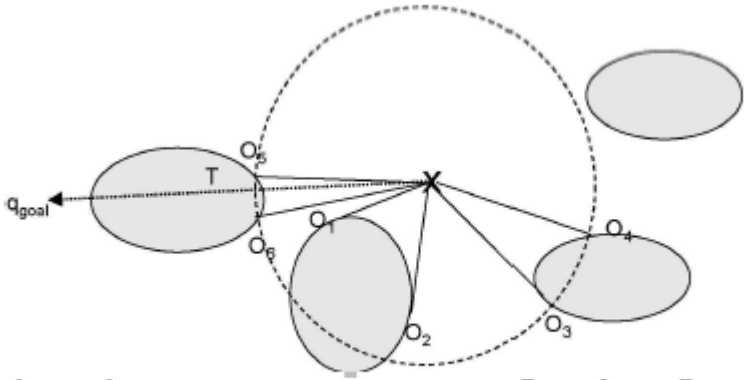
- *Tangent Bug* se basa en la búsqueda de puntos finales finitos, segmentos continuos de ρR.



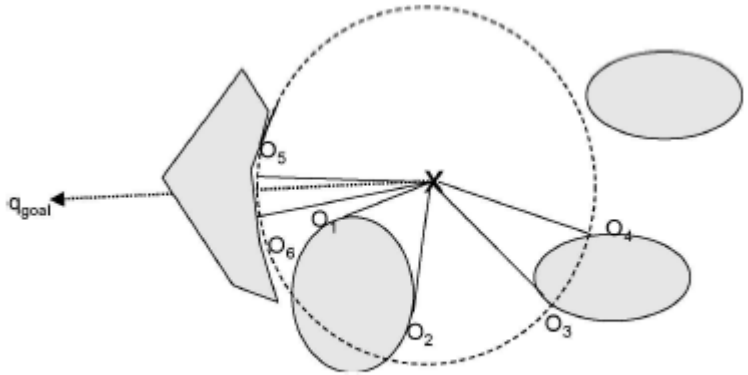


Actualmente, el bug piensa que puede llegar a la meta!

Algoritmo *Tangent Bug* (cont.)



Elige el punto *Oi* que minimiza *d*(*x*,*Oi*) + *d*(*Oi*, *q*goal)
"Distancia heurística"



Si esta distancia encuentra un obstácula, debe actuar como un BUG y seguir el límite

Ideas Básicas

- Un comportamiento de movimiento-a-objetivo, siempre y cuando el camino esté despejado o si hay un obstáculo encontrar punto límite visible que disminuye la distancia heurística.
- Un límite como consecuencia de comportamientos se invoca cuando se aumenta la distancia heurística.
- Terminar comportamiento del límite siguiente cuando $d_{reach} < d_{followed}$.
  - Donde:
    - $d_{followed}$: La distancia más corta entre el límite percibido y la meta.
    - d_{reach}: La distancia más corta entre el obstáculo y el objetivo (o distancia a la meta si ningún obstáculo que bloquee visible).

Métodos Wavefront

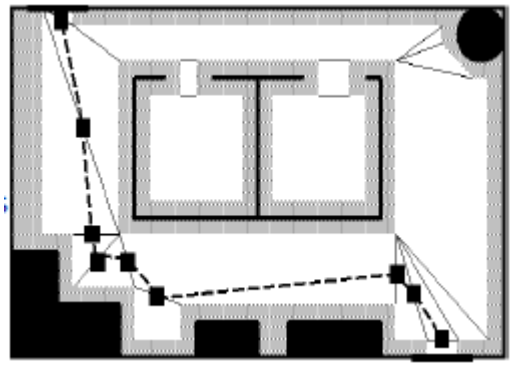
- Dividir el mapa en una cuadrícula.
- Dar un número a cada celda de la cuadrícula de acuerdo con su distancia, siga los números en orden decreciente.

Planes de Trabajo

- Algoritmos de representación *CSpace*.
- Mapas de Meadow.
- Gráficos de visibilidad.
- Gráficos generalizados de Voronoi.
- Descomposiciones de celdas.

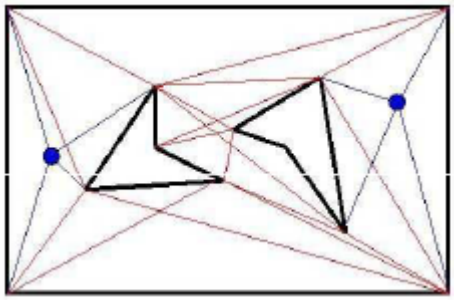
Mapas de Meadow

- Ampliar obstáculos tan grandes como el robot, transformar desde el espacio de trabajo al espacio de configuración
- Encontrar esquinas de los objetos.
- Conectar las esquinas.
- Usar centros de estos bordes como puntos de referencia.



Gráficos de Visibilidad

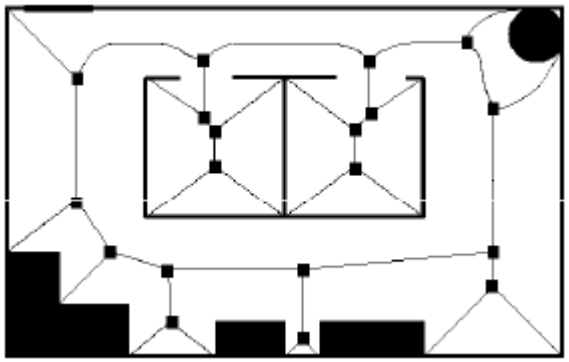
- Conectar cada par de:
  - Puntos importantes.
  - Vértices de los obstáculos.
  - Los puntos inicial y final.



- Los bordes no deben estar en colisión con otros objetos.
- Graficar los bordes y nodos de representación *CSpace*.
- Las búsquedas se realizan en este gráfico.

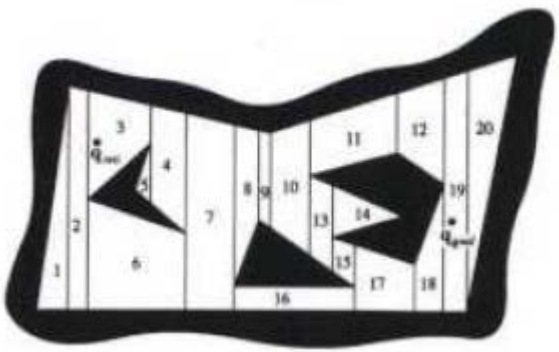
Gráficos Generalizados de Voronoi

- Los bordes de la gráfica de Voronoi:
  - Los puntos en el espacio de trabajo, que tienen la misma distancia de los obstáculos circundantes. [Aur91].
- Los nodos de la gráfica de Voronoi:
  - Las intersecciones de estas líneas.



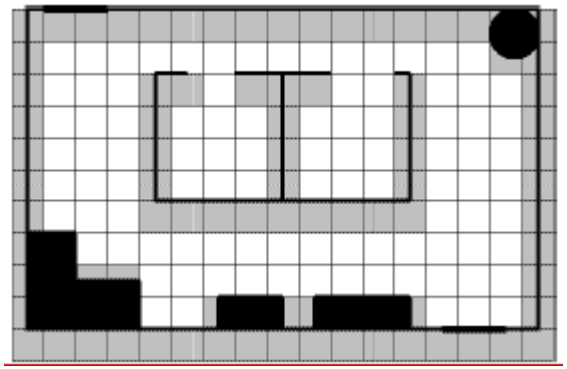
Descomposición de Celdas Trapezoidal

- Iniciar una nueva línea para cada punto crítico
  - Puntos de esquina.
- Dividir el espacio en trapecios.
- Desplazarse de uno al siguiente.



Celdas Regulares

- Dividir el mapa en celdas regulares.
  - Colisión libre, chocando y regiones de semichoque.
  - Similares a los píxeles agrandados.
- Es completa:
  - Si existe un camino, se encontró.



Enfoque Campo Potencial

- La localización de la meta genera un **potencial atractivo** - tirando el robot hacia la meta.
- Los obstáculos generan un **potencial repulsivo** - jalando el robot lejos de los obstáculos.
- El **gradiente negativo del potencial total** se trata como una fuerza artificial aplicada al robot.
- Se realiza la suma de las fuerzas de control del robot.

Enfoque Campo Potencial (cont.)

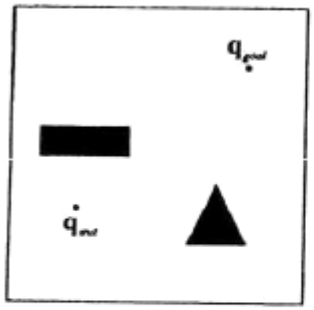
Artificial **Potential**

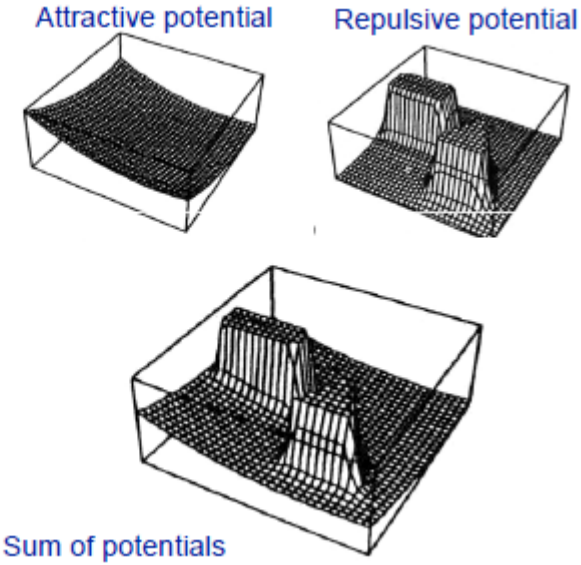
$$U(q) = \underbrace{U_{goal}(q)}_{\text{attractive potential}} + \underbrace{\sum U_{obstacles}(q)}_{\text{repulsive potential}}$$

Artificial **Force Field**

$$F(q) = -\nabla U(q)$$

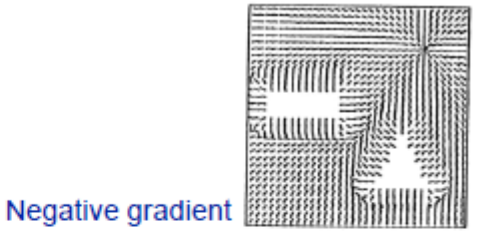
Negative gradient

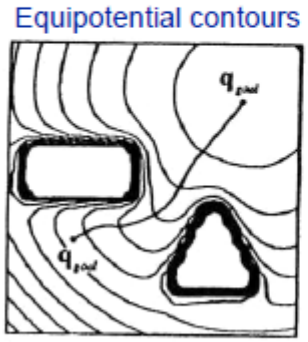


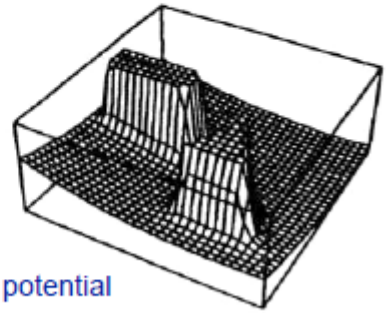


Enfoque Campo Potencial (cont.)

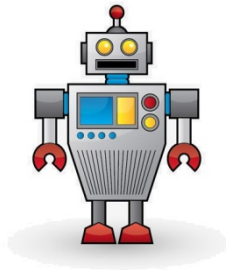
- Después de obtener el potencial total, generar campos de fuerza (gradiente negativo).
- Se realiza la suma de las fuerzas de control del robot.







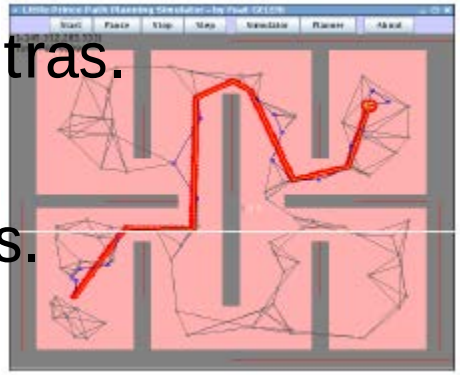
En gran medida, esto es computable a partir de las lecturas de los sensores



Algoritmos Basados en Muestreo

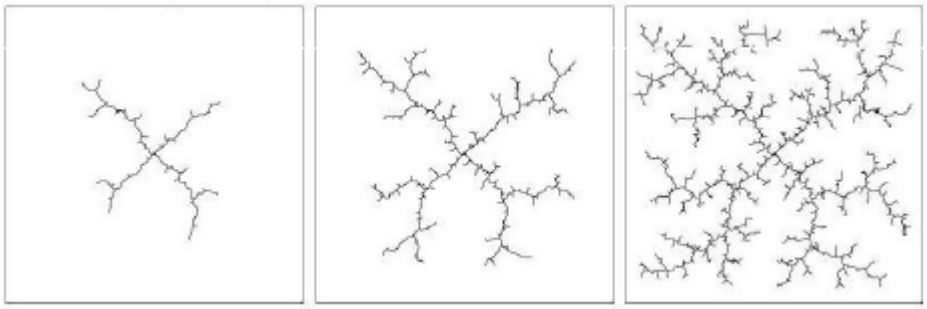
Planes de Trabajo Probabilísticos

- Construir un plan de ruta.
  - Configuración aleatoria de *N* muestras.
  - Colisión libre.
  - Trate de conectarse unos con otros.
  - A nodo más cercano *M*.
- Consultar una ruta de acceso.
  - Conectar q_{init} y q_{goal} a la gráfica.
- Buscar en el gráfico.
- Búsqueda rápida, pero necesita pre-procesamiento para la construcción del plan de ruta.



Explorando Rápidamente un Árbol Aleatorio (RRT)

- Seleccionar una configuración aleatoria q_{rand} en C_{free}.
- El nodo más cercano de configuración en el árbol se encuentra $q_{nearest}$.
- Tratar de conectarse de $q_{nearest}$ a q_{rand}.



Referencias Bibliográficas

- Fu, K.S.; González, R.C. y Lee, C.S.G. Robotics: Control, Sensing, Vision, and Intelligence. McGraw-Hill. 1987.


¿Preguntas?

¡Gracias!

M.Sc. Kryscia Daviana Ramírez Benavides
Profesora e Investigadora
Universidad de Costa Rica
Escuela de Ciencias de la Computación e Informática

Sitio Web: http://www.kramirez.net/
E-Mail: kryscia.ramirez@ucr.ac.cr
kryscia.ramirez@ecci.ucr.ac.cr

Redes Sociales: