Robotics Programming Laboratory

Bertrand Meyer
Jiwon Shin

Lecture 5: Obstacle Avoidance
Obstacle avoidance: our perspective
Obstacle avoidance: robot’s perspective
Bug algorithms

- Known:
  - Goal position
  - Current position
  - Sensing ability to detect nearby obstacles
- Sense -> Act: does not store any past information
- Sensor:
  - Bug 0, Bug 1, Bug 2: tactile sensor
  - Tangent Bug: range sensor
Bug 0

1. **Move toward the goal:**
   1. If the goal is reached: Stop
   2. If an obstacle is in the way: Go to step 2

2. **Follow the obstacle boundary:**
   1. If no obstacle in the way, go back to step 1.
When does Bug 0 fail?
1. **Move toward the goal:**
   1. If the goal is reached: Stop
   2. If an obstacle is in the way: Go to step 2

2. **Follow the obstacle boundary:**
   1. Mark the closest
   2. After a complete loop: Go to the closest point to the goal then go back to step 1.

Lumelsky, V. & Stepanov, A. “Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape,” Algorithmica 2:403-430. 1987
Will Bug 1 fail?
How much would Bug 1 travel?

Given
- D: distance between start and goal
- $P_i$: Perimeter of i’th obstacle

Shortest travel distance?
- D

Longest travel distance?
- $D + 1.5 \sum_i P_i$
Bug 2

1. Move toward the goal:
   1. If the goal is reached: Stop
   2. If an obstacle is in the way: Go to step 2

2. Follow the obstacle boundary:
   1. If the goal line is crossed: Go to step 1.

Is crossing the goal line important?
How well does Bug 2 work?

Goal

Towards the goal

Closer to the goal
How well does Bug 2 work?
How well does Bug 2 work?
How much would Bug 2 travel?

Given

- D: distance between start and goal
- \( P_i \): Perimeter of \( i \)’th obstacle
- \( n_i \): number of times \( i \)’th obstacle crosses the goal line

Shortest travel distance?

- D

Longest travel distance?

- \( D + \frac{1}{2} \sum_i n_i P_i \)
Bug 1 vs Bug 2

**Bug 1**
- Exhaustive search: analyze all choices before committing
- More predictable performance

**Bug 2**
- Greedy search: take the first viable choice
- Generally outperforms Bug 1 but could be worse if the obstacles are complex
Can we do better if we can see more?
1. **Move toward the goal:**
   1. If the goal is reached: Stop
   2. If a local minimum is detected: Go to step 2

2. **Move along the boundary marking** $d_{\text{min}}$:
   1. If the goal is reached: Stop
   2. If $d(V_{\text{leave}}, \text{goal}) < d_{\text{min}}$: Go to step 3

3. **Perform the transition phase:**
   1. Move directly towards $V_{\text{leave}}$ until $Z$, where $d(Z, \text{goal}) < d_{\text{min}}$: Go to step 1

---

Visibility graph & tangent graph

Visibility graph

Tangent graph
Local tangent graph

Goal

$G_{node}$

Start

$R$
Local minimum detection

\[ d(V, \text{goal}) < d(x, \text{goal}) \text{ for all } V \]
Wall Following

\[ \mathbf{v}_{\text{wall}} = \mathbf{p}_2 - \mathbf{p}_1 \]

\[ \mathbf{v}_{\text{distance}} = (d_{\text{current}} - d_{\text{desired}}) \mathbf{v}_{\text{perpendicular}} \]

\[ \mathbf{v}_{\text{robot}} = d_{\text{desired}} \mathbf{v}_{\text{wall}} + \mathbf{v}_{\text{distance}} \]
Leave condition detection

\[ d(V_{\text{leave}}, \text{goal}) < d_{\text{min}} \]
Sensor range

Zero

Infinite
Unreachable goal
Loop closure

Challenging!

- Drift
- Limited sensor information