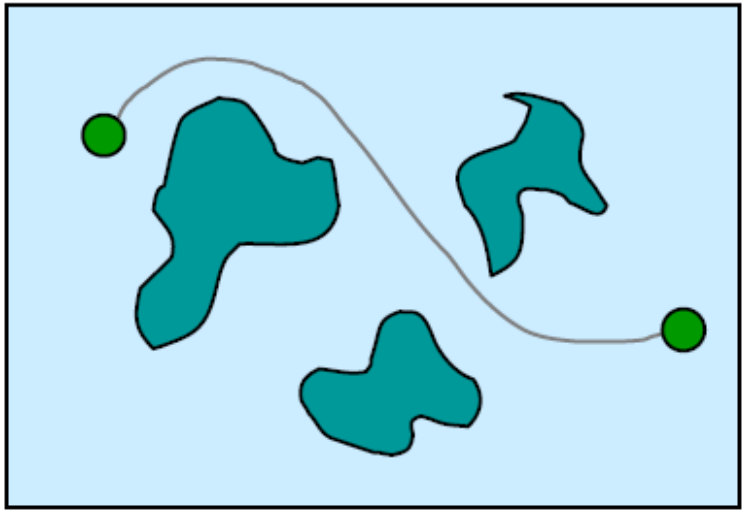
3. Path Planning



Outline

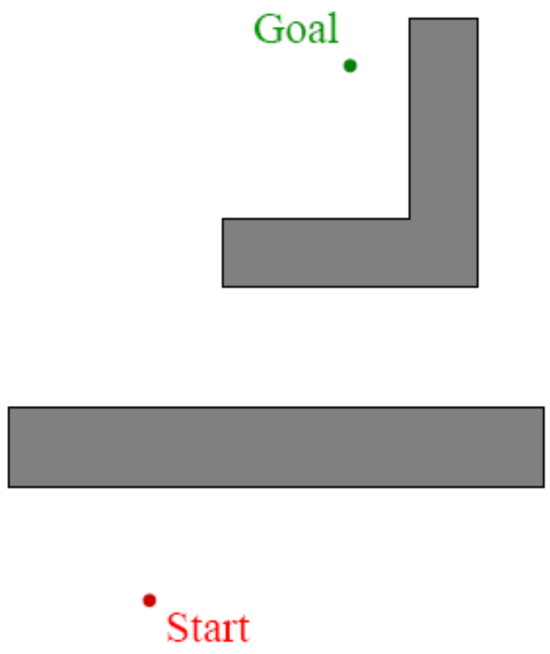
- Bug algorithms
- Configuration space
- Potential functions – Wavefront planner
- Topological maps – Visibility graph
- Graph search - A* algorithm
- Cell decompositions
- Sampling-based algorithms

Bug algorithms

- They are inspired from insects
- Simple Bug behaviours:
  - follow a wall
  - move toward a goal
- Assumptions:
  - the direction to the goal is known
  - tactile sensors

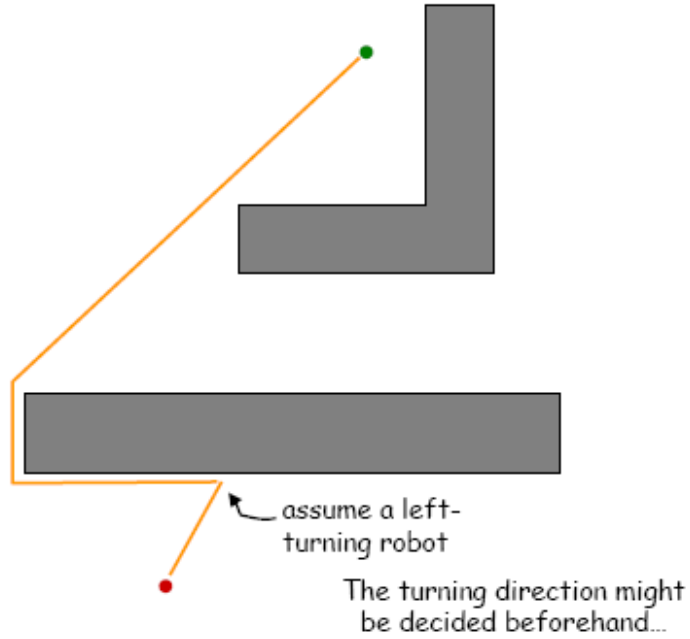
Bug algorithms

- Bug 0 algorithm:
  1. head toward goal
  2. follow obstacle (left or right) until you can head toward the goal again
  3. continue



Bug algorithms

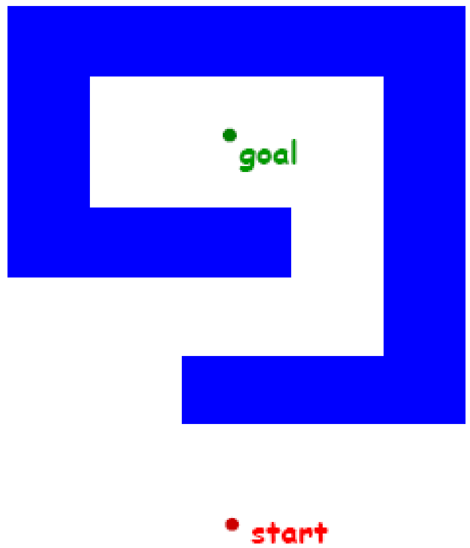
- Bug 0 algorithm:
  1. head toward goal
  2. follow obstacle (left or right) until you can head toward the goal again
  3. continue



Bug algorithms

- Bug 0 algorithm:
    1. head toward goal
    2. follow obstacle (left or right) until you can head toward the goal again
    3. continue

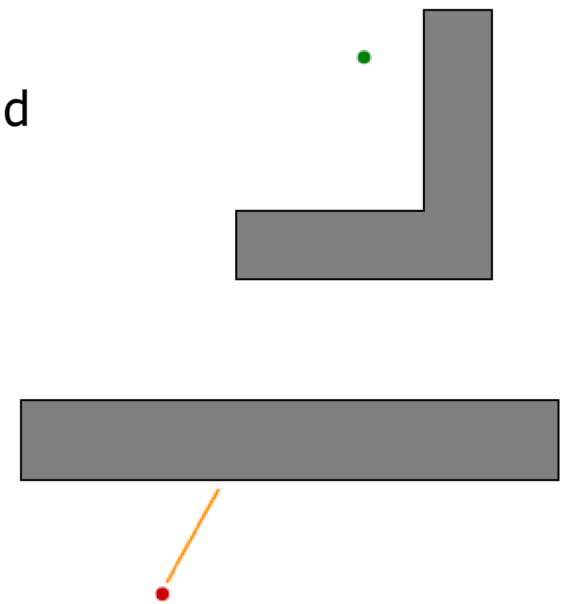
What is the trajectory in this environment?



Bug algorithms

Adding some memory, it is possible to improve Bug 0

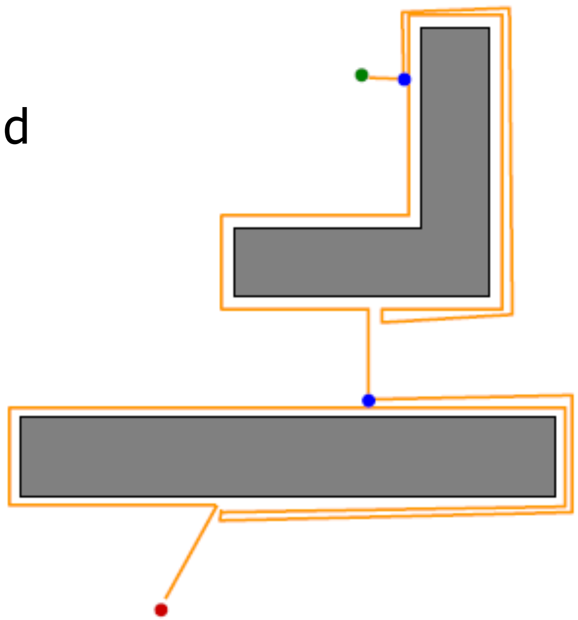
- Bug 1 algorithm:
  1. head toward goal
  2. if an obstacle is encountered circumnavigate it and remember how close you get to the goal
  3. return to that closest point and continue



Bug algorithms

Adding some memory, it is possible to improve Bug 0

- Bug 1 algorithm:
  1. head toward goal
  2. if an obstacle is encountered circumnavigate it and remember how close you get to the goal
  3. return to that closest point and continue



Bug algorithms

- Bug 1 algorithm:

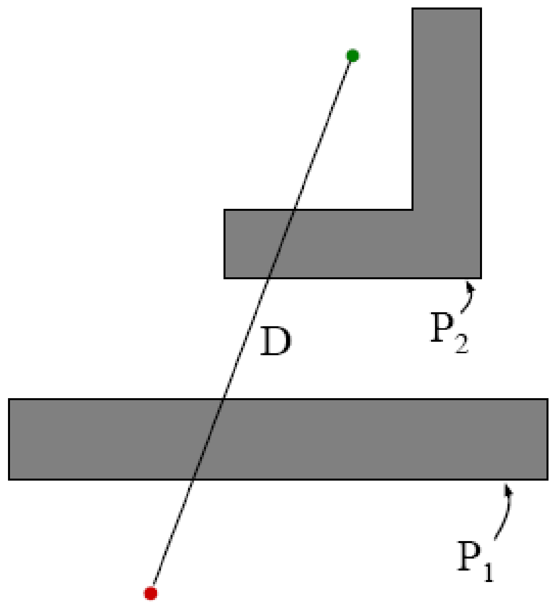
What are the upper/lower bounds on the path length that the robot takes?

D: straight-line distance from start to goal

P_i: perimeter of the ith obstacle

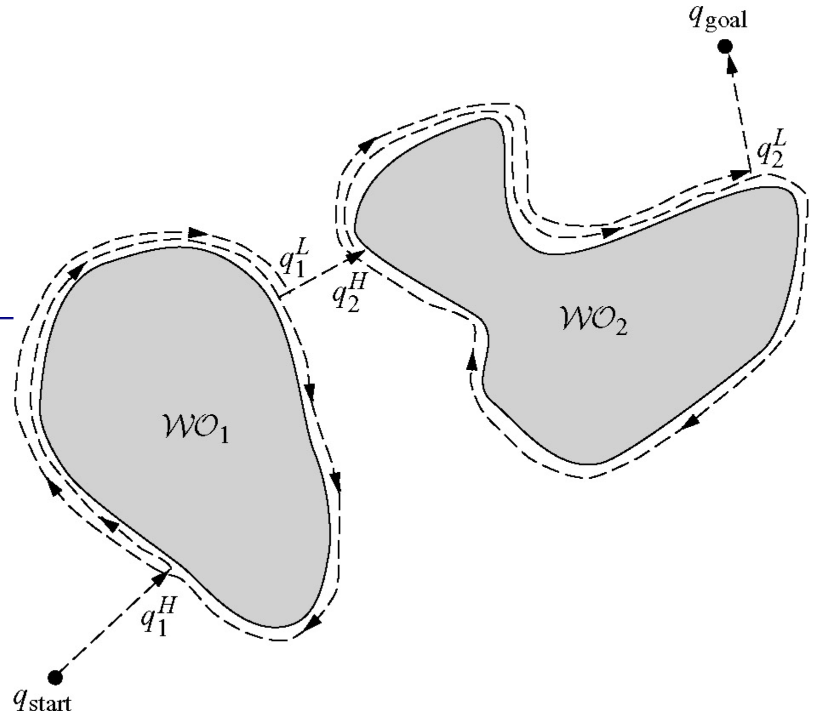
lower bound: D

upper bound: D + 1.5 $\sum_i P_i$

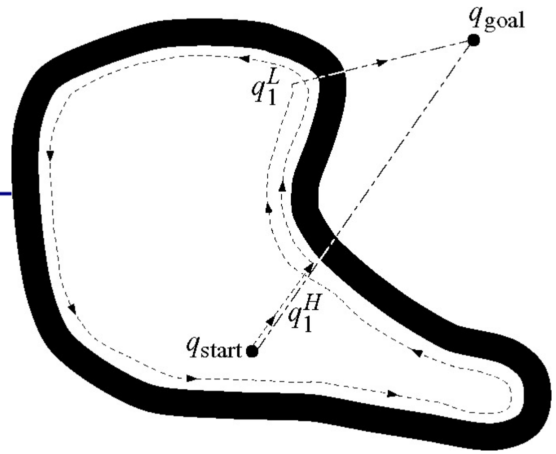


Bug algorithms

- Bug 1 algorithm:



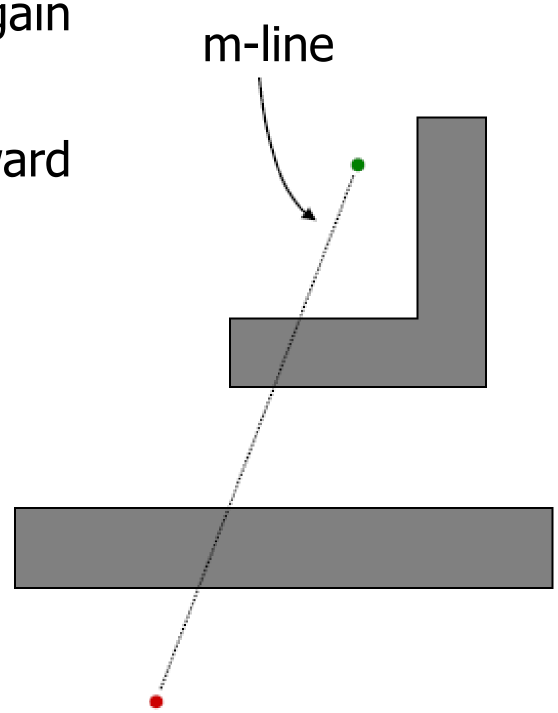

```
Input: A point robot with a tactile sensor
Output: A path to the q_goal or a conclusion no such path exists
while Forever do
  repeat
    From q^L_{i-1}, move toward q_goal.
  until q_goal is reached or an obstacle is encountered at q^H_i
  if Goal is reached then
     Exit.
  end if
  repeat
    Follow the obstacle boundary.
   until q_goal is reached or q^H_i is re-encountered.
   Determine the point q^L_i on the perimeter that has the shortest distance to the goal.
   Go to q^L_i.
   if the robot were to move toward the obstacle then
     Conclude q_goal is not reachable and exit.
   end if
end while
```



Bug algorithms

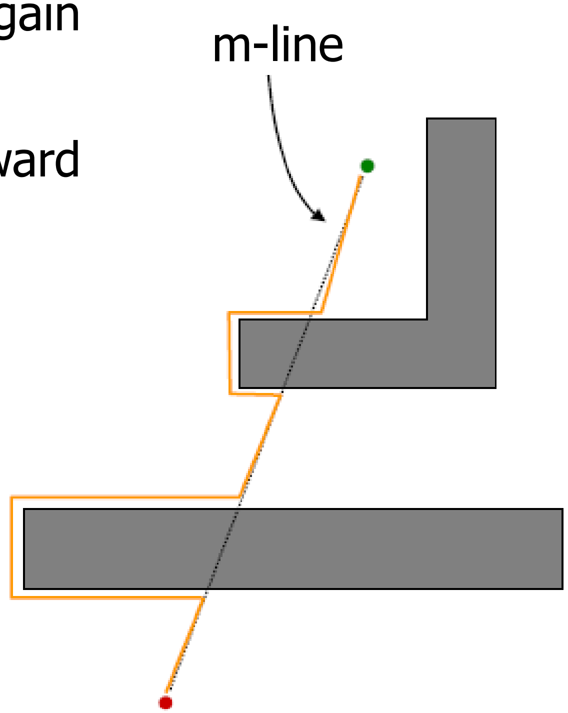
Another possibility

- Bug 2 algorithm:
  1. head toward goal on the m-line
  2. if an obstacle is in the way, follow it until you encounter the m-line again closer to the goal
  3. leave the obstacle and continue toward the goal



Another possibility

- Bug 2 algorithm:
  1. head toward goal on the m-line
  2. if an obstacle is in the way, follow it until you encounter the m-line again closer to the goal
  3. leave the obstacle and continue toward the goal



Bug algorithms

- Bug 2 algorithm:

What are the upper/lower bounds on the path length that the robot takes?

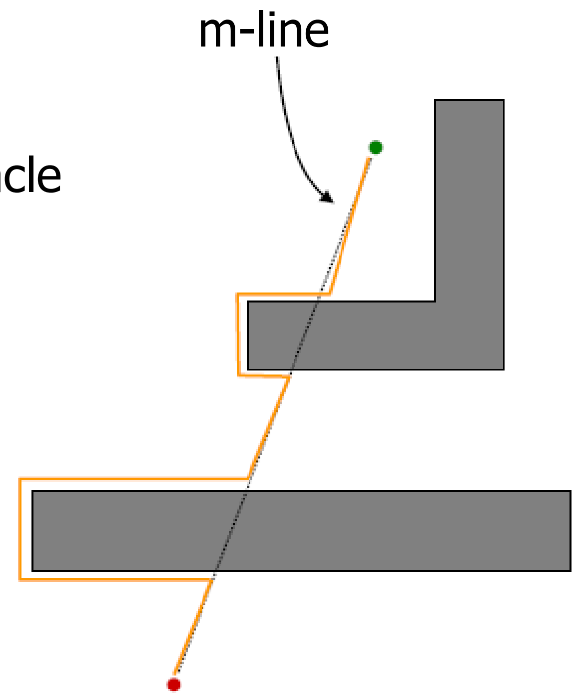
D: straight-line distance from start to goal

P_i: perimeter of the ith obstacle

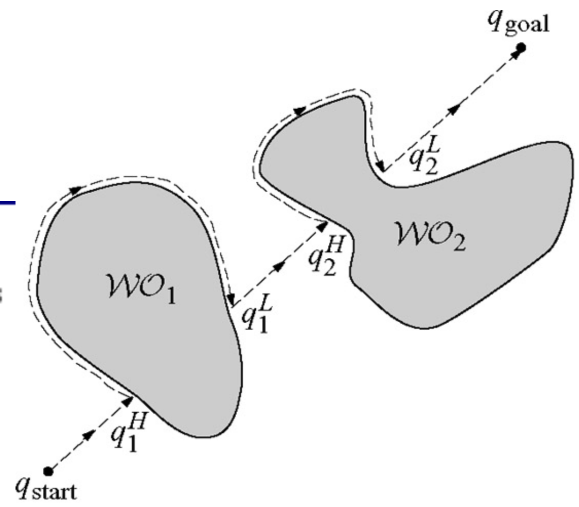
n_i: # of m-line intersections of the ith obstacle

lower bound: D

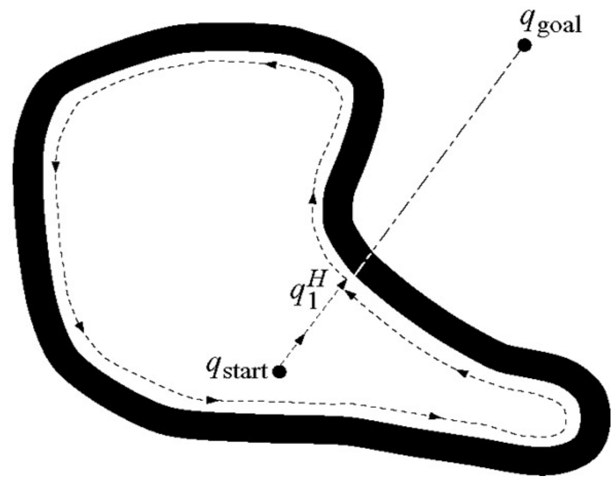
upper bound: $D + \sum_i (n_i \cdot P_i / 2)$



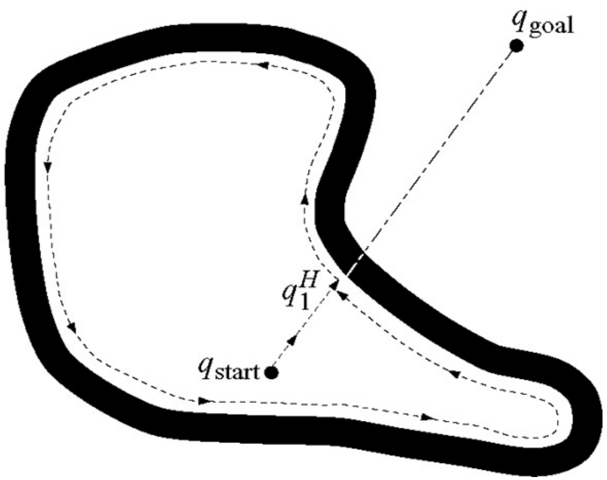
- Bug 2 algorithm:

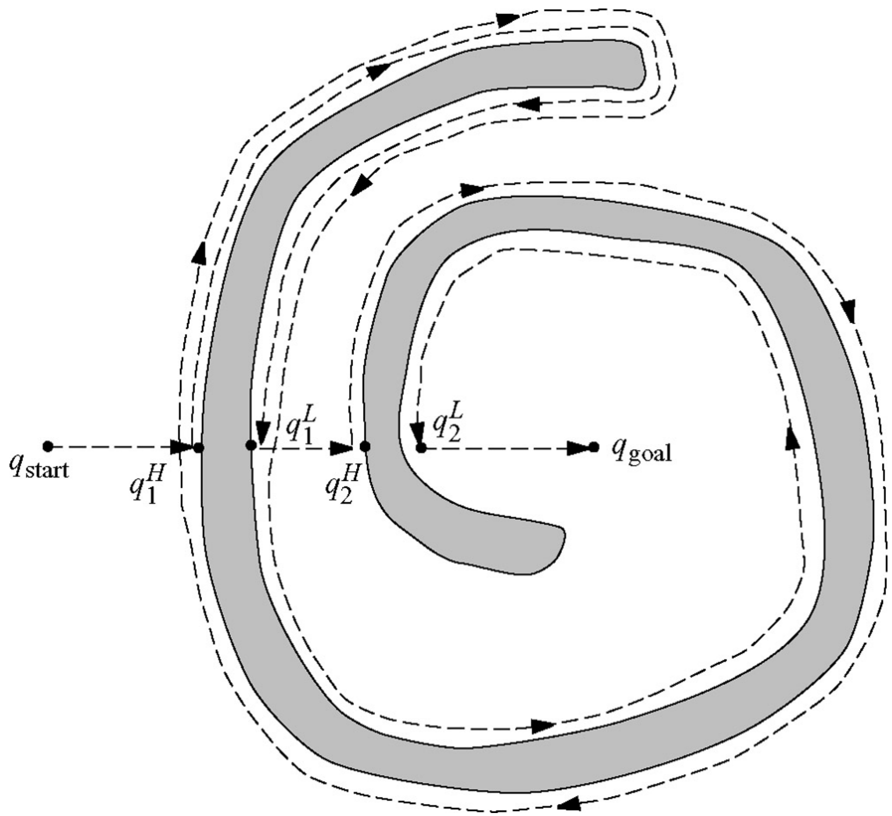


Input: A point robot with a tactile sensor
Output: A path to q_{goal} or a conclusion no such path exists
while True **do**
  repeat
    From q^L_{i-1}, move toward q_{goal} along m-line.
  until
q_{goal} is reached **or**
an obstacle is encountered at *hit point* q^H_i.
  Turn left (or right).
  repeat
    Follow boundary
  until
   q_{goal} is reached **or**
    q^H_i is re-encountered **or**
    m-line is re-encountered at a point m such that
     $m \neq q^H_i$ (robot did not reach the hit point),
     $d(m, q_{goal}) < d(m, q^H_i$ (robot is closer), and
     If robot moves toward goal, it would not hit the obstacle
    Let $q^L_{i+1} = m$
    Increment i
end while

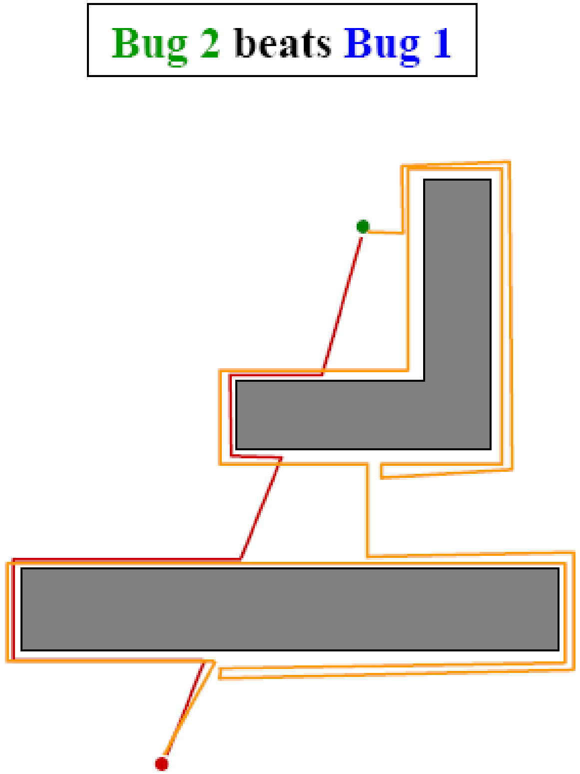


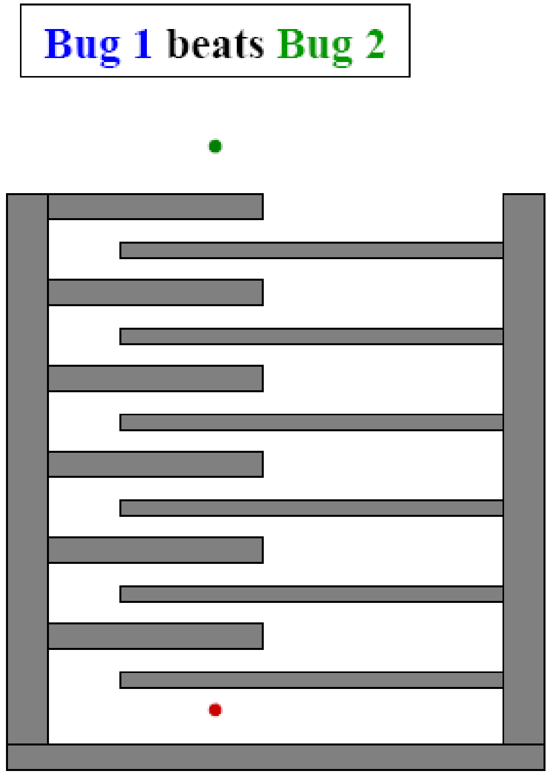
- Bug 2 algorithm:





Bug algorithms



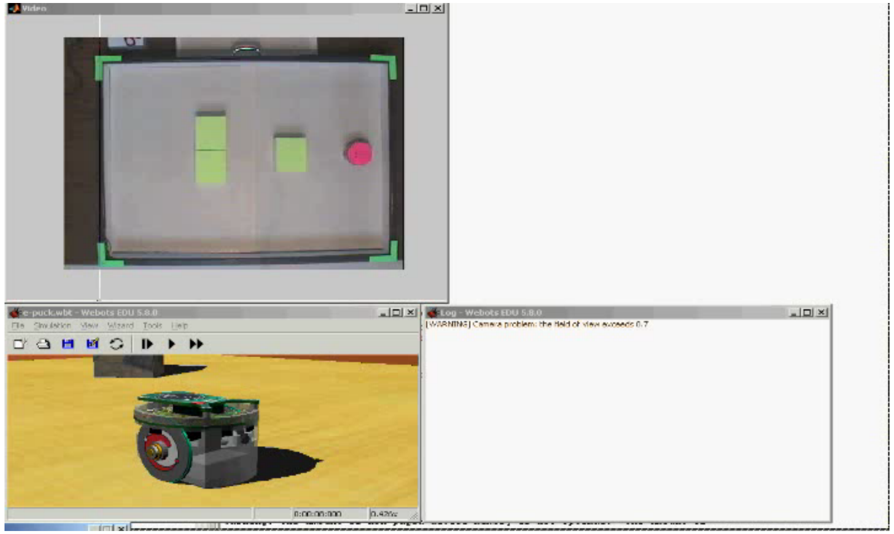


Bug 1 is an exhaustive search algorithm: *it looks first all choices*

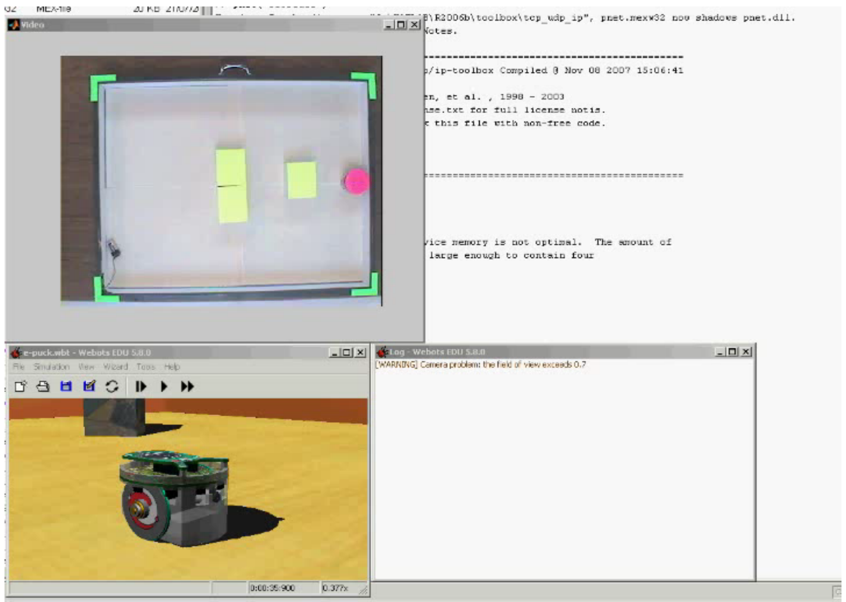
Bug 2 is a greedy algorithm: *it takes the first thing that looks better*

Bug algorithms

- Bug 1 algorithm:



- Bug 2 algorithm:



Bug algorithms

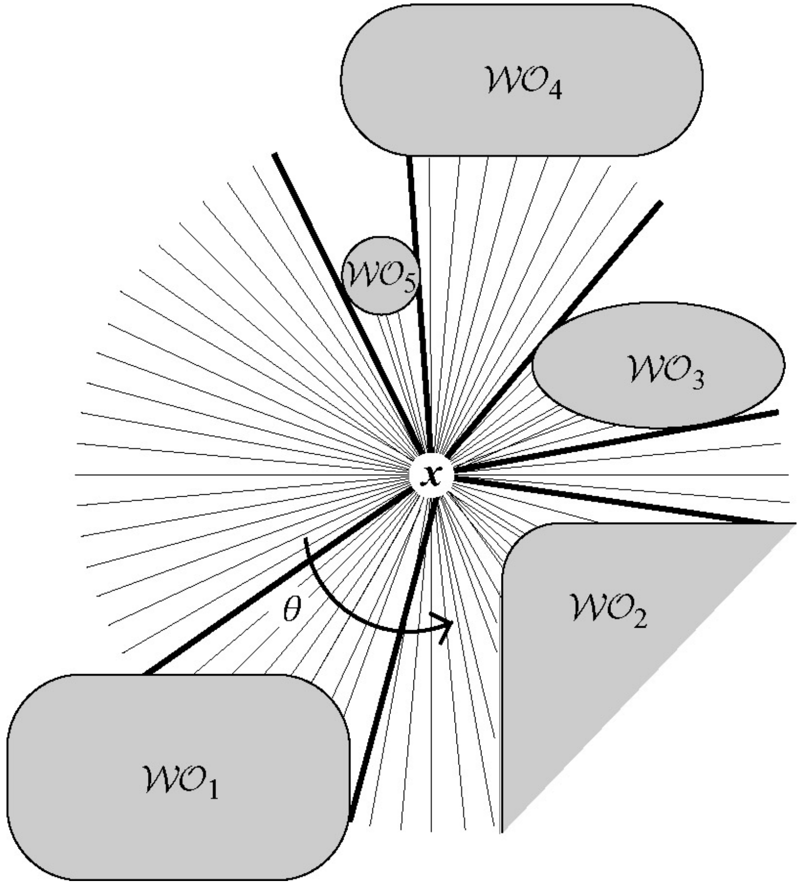
having range sensors...

- Tangent Bug algorithm:

$$\rho(x,\theta) = \min_{\lambda \in [0,\infty]} d(x, x + \lambda[\cos\theta, \sin\theta]^T),$$
$$\text{such that } x + \lambda[\cos\theta, \sin\theta]^T \in \bigcup_i \mathcal{WO}_i.$$

$$\rho_R : \mathbb{R}^2 \times S^1 \to \mathbb{R}$$

$$\rho_R(x,\theta) = \begin{cases} \rho(x,\theta), & \text{if } \rho(x,\theta) < R \\ \infty, & \text{otherwise.} \end{cases}$$



Bug algorithms

- Tangent Bug algorithm:

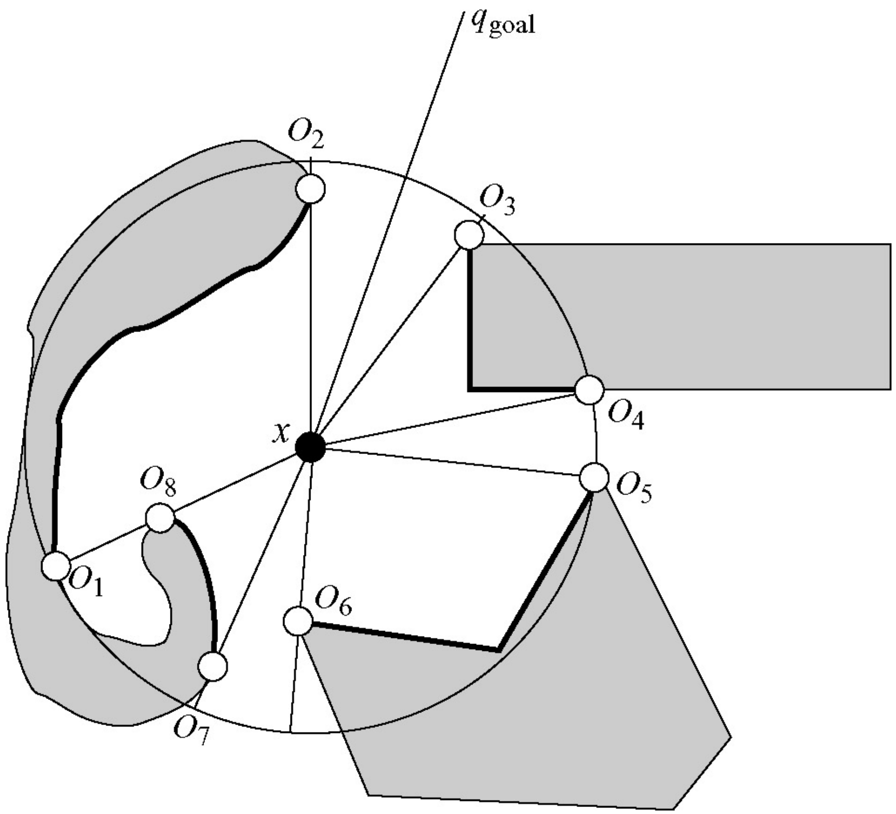
Discontinuity points:

$O_1, O_2, O_3, O_4, O_5, O_6, O_7, O_8$

Continuity intervals

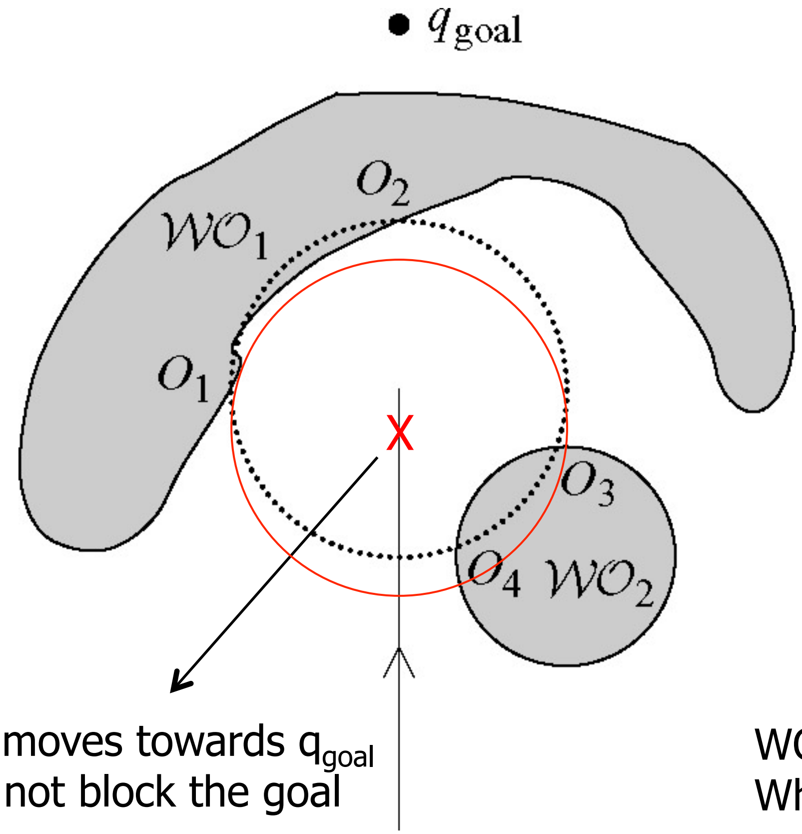
$O_1 \rightarrow O_2$, $O_3 \rightarrow O_4$

$O_5 \rightarrow O_6$, $O_7 \rightarrow O_8$

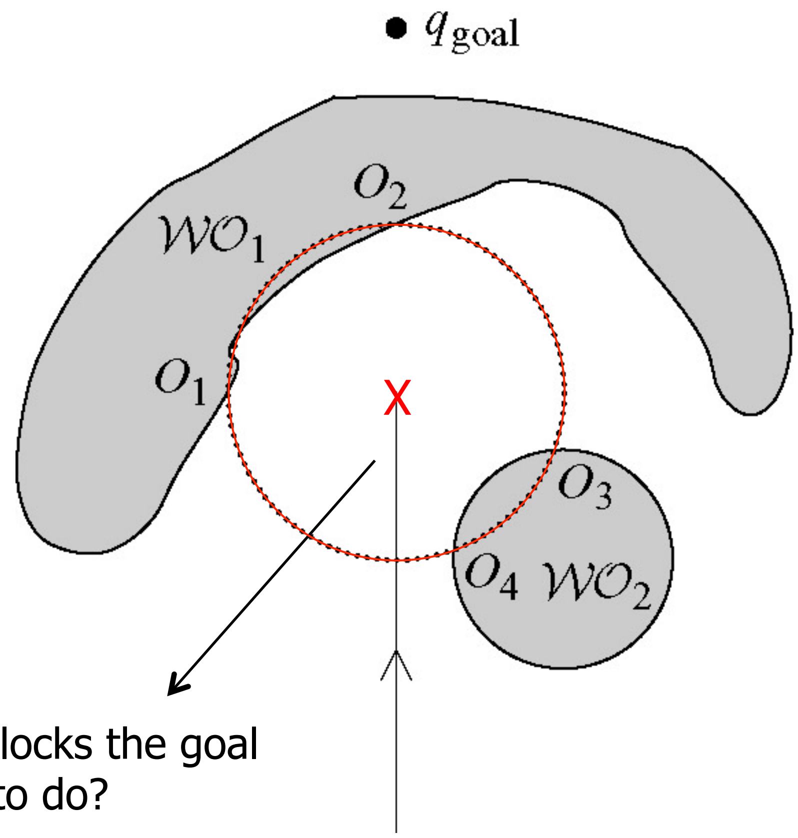


Bug algorithms

- Tangent Bug algorithm:



The robot moves towards q_{goal}
WO_2 does not block the goal



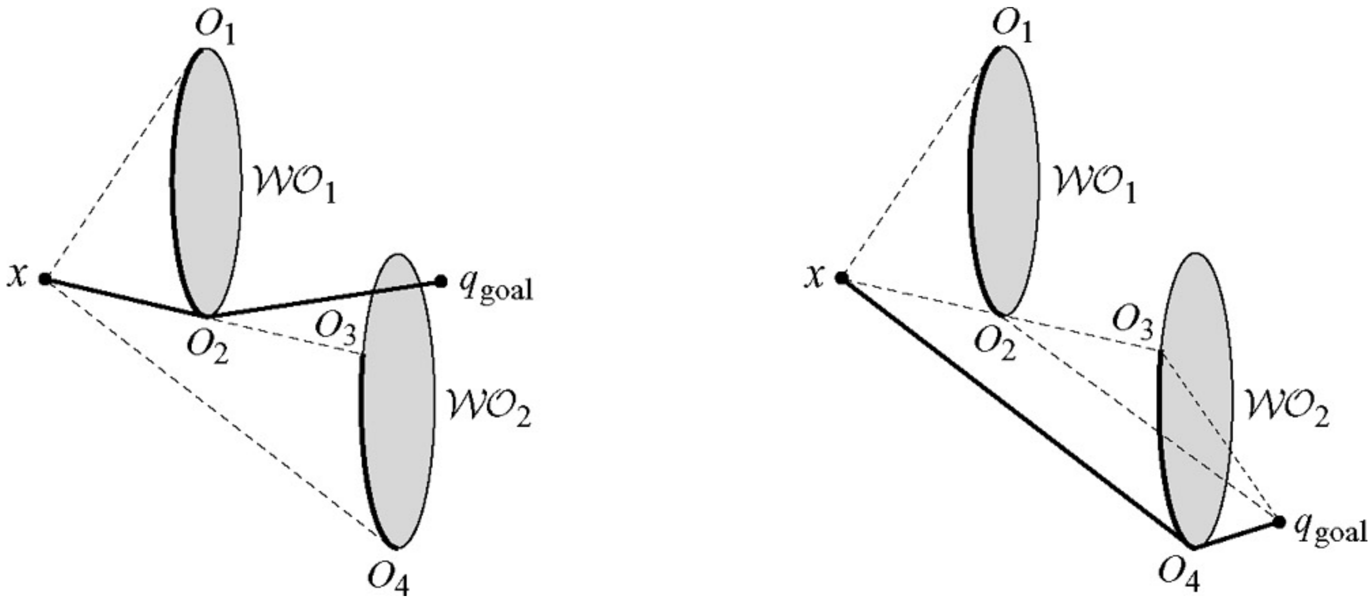
WO_1 blocks the goal
What to do?

Bug algorithms

- Tangent Bug algorithm:

The robot then moves toward the Oi that maximally decreases a heuristic distance to the goal.

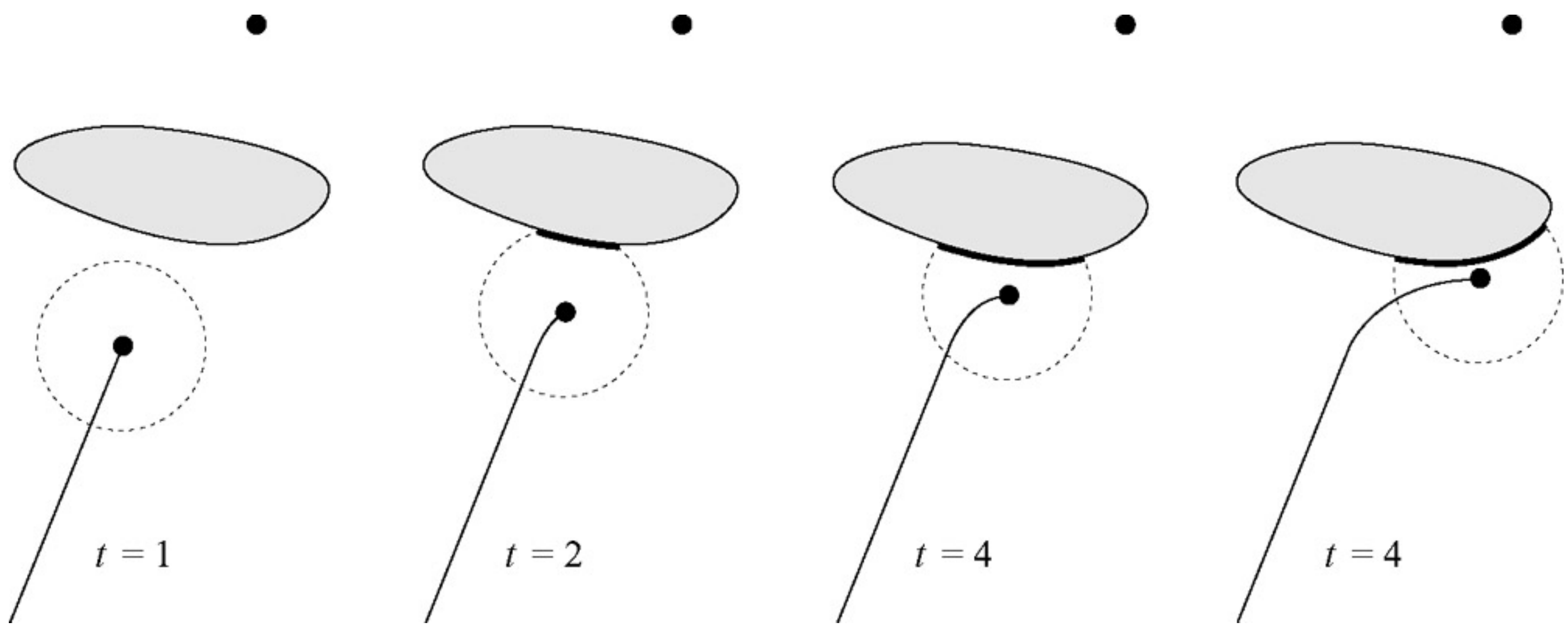
choose O_i that minimizes: $d(x, Oi) + d(Oi, q_{goal})$



- Tangent Bug algorithm:

Avoiding the obstacle:

PART 1: MOTION TO GOAL BEHAVIOUR



Bug algorithms

- Tangent Bug algorithm:

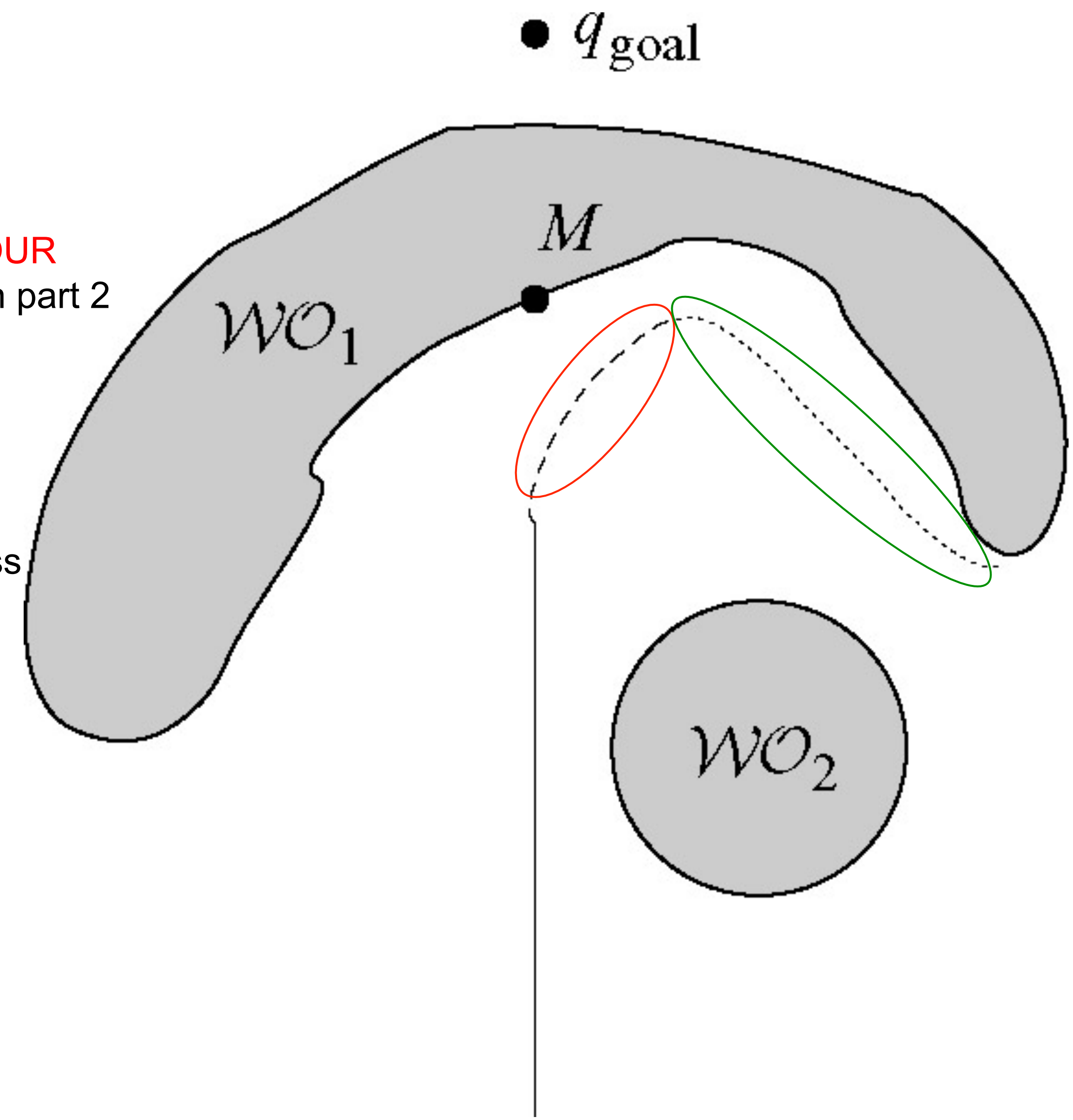
Avoiding the obstacle:

PART 1: MOTION TO GOAL BEHAVIOUR

... until d starts increasing, then part 2

PART 2: BOUNDARY FOLLOWING BEHAVIOUR

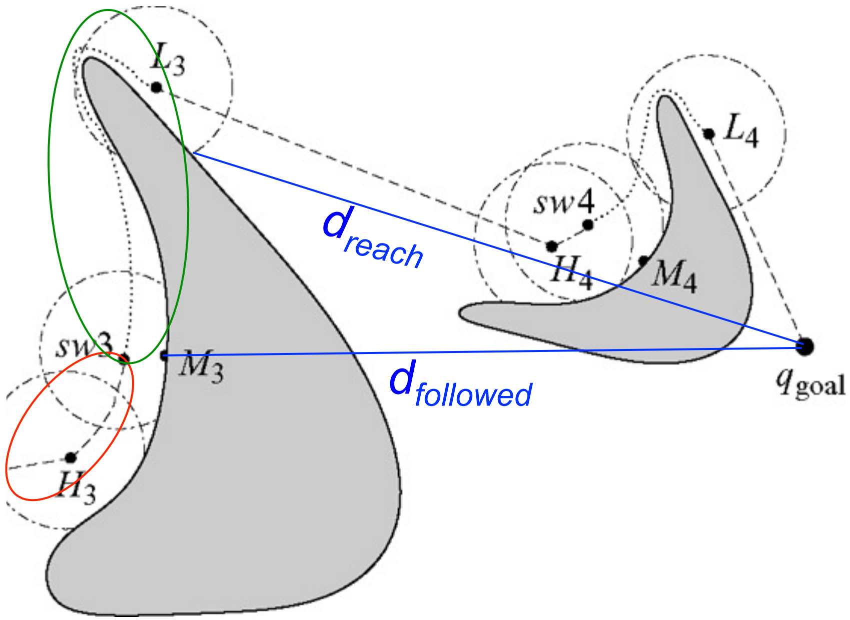
Follow the boundary until the distance to goal from one reachable point O_i (d_{reach}) is less than the distance to goal from any past followed point.
Then, part 1.



Bug algorithms

- Tangent Bug algorithm:

$d_{followed}$ is the shortest distance between the boundary which had been sensed and the goal.



d_{reach} is the distance between the goal and the closest point on the followed obstacle that is within line of sight of the robot

$$d_{\text{reach}} = \min_{c \in \Lambda} d(q_{\text{goal}}, c).$$

- Tangent Bug algorithm:

Input: A point robot with a range sensor
Output: A path to the q_{goal} or a conclusion no such path exists

while True **do**
  repeat
    Continuously move toward the point $n \in \{T, O_i\}$ which minimizes $d(x, n) + d(n, q_{goal})$
  until

- the goal is encountered **or**
- The direction that minimizes $d(x, n) + d(n, q_{goal})$ begins to increase $d(x, q_{goal})$, i.e., the

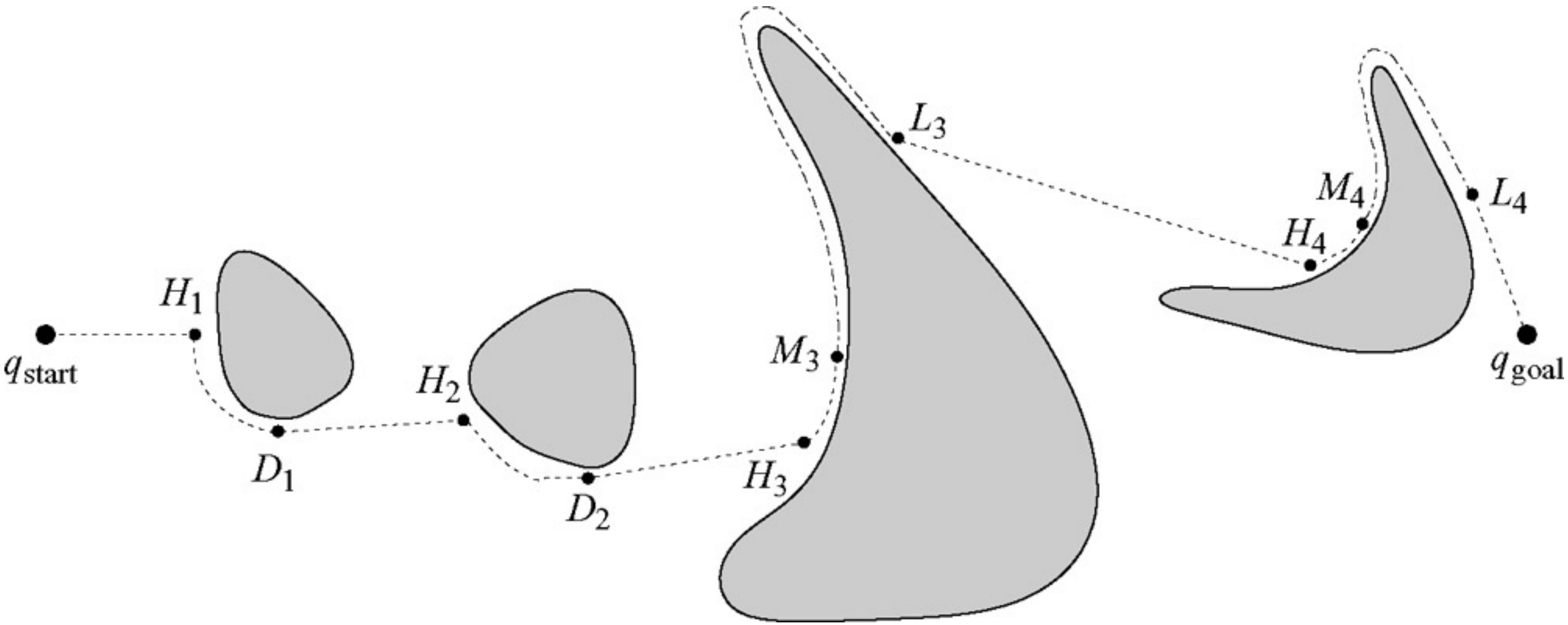
  Chose a boundary following direction which continues in the same direction as the most recent
  repeat
    Continuously update d_{reach}, $d_{followed}$, and $\{O_i\}$.
    Continuously moves toward $n \in \{O_i\}$ that is in the chosen boundary direction.
  until

- The goal is reached.
- The robot completes a cycle around the obstacle in which case the goal cannot be achieved.
- $d_{reach} < d_{followed}$

end while

- Tangent Bug algorithm:

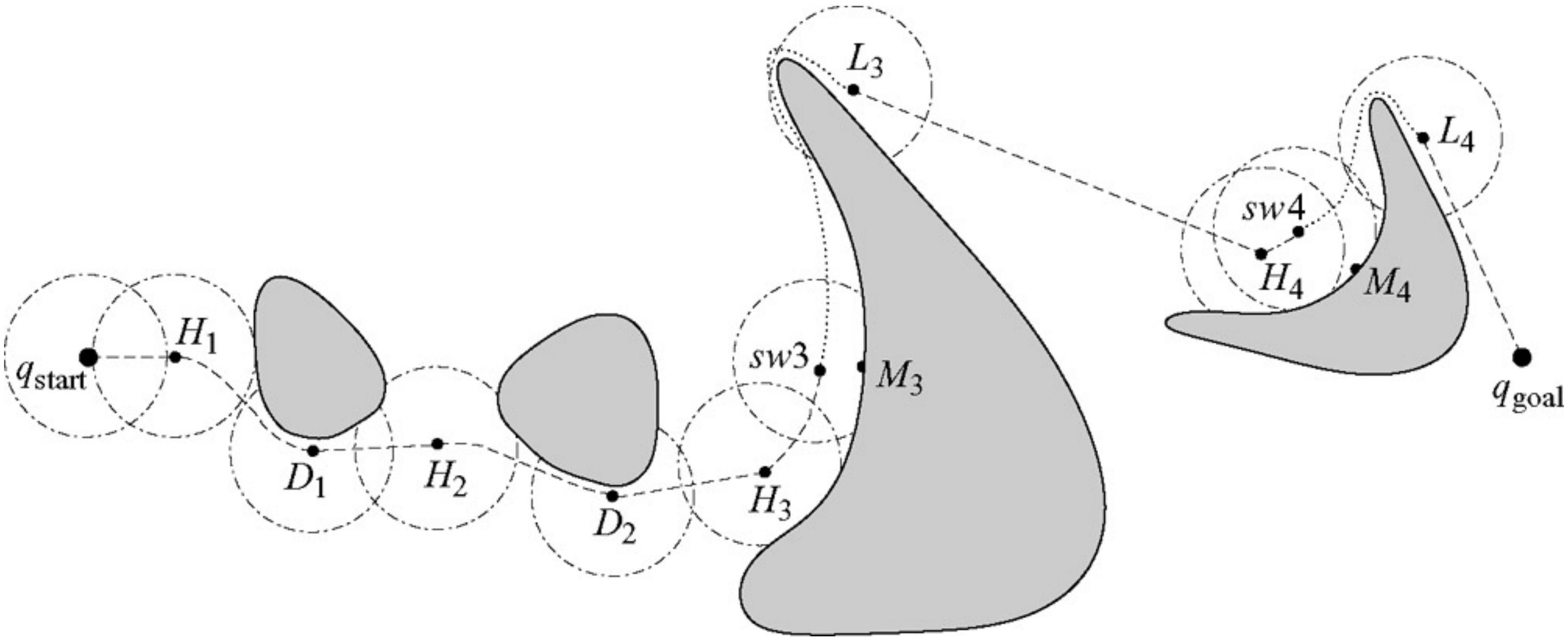
Tangent Bug with zero sensor range



Bug algorithms

- Tangent Bug algorithm:

Tangent Bug with finite sensor range



Bug algorithms

- Tangent Bug algorithm:

Tangent Bug with infinite sensor range