Path Planning for Point Robots

Jane Li

Assistant Professor
Mechanical Engineering Department, Robotic Engineering Program
Worcester Polytechnic Institute

Quiz (10 pts)

- (3 pts) Why human-robot handing-over is not a trivial problem?
- (3 pts) List three motion planning problems you may encounter in physical human-robot handing-over
- (4 pts) Use an example to describe how to combine motion planning with robot learning

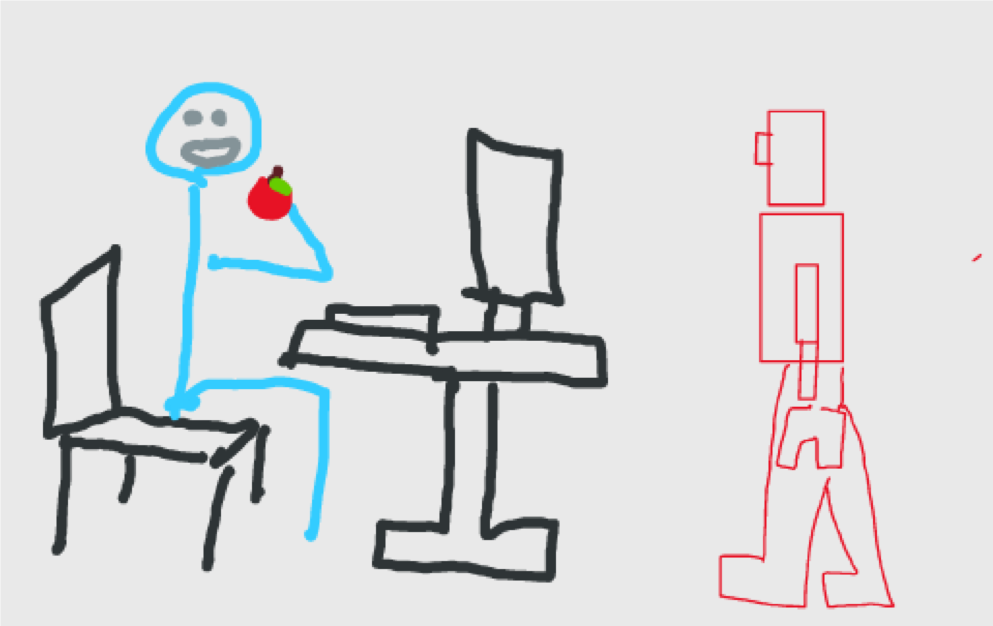
Not a trivial problem

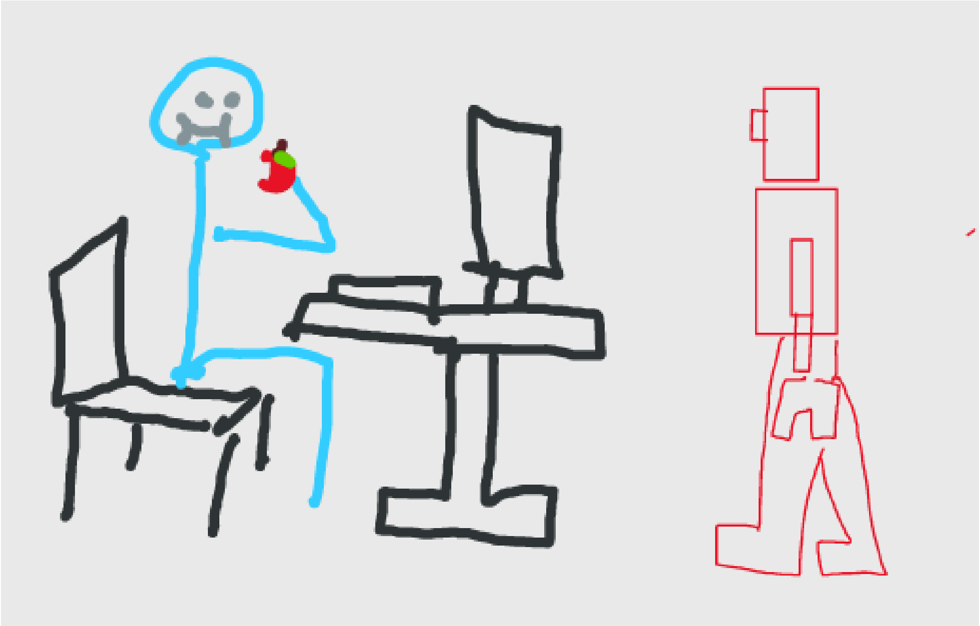
- Goal
  - **Fluent** and **natural** human-robot object handing-over

- Human-robot object handing-over is not a trivial problem
  - Infer human intent
  - Identify object affordance
  - Planning feasible and natural motion
  - Handle exception

Example

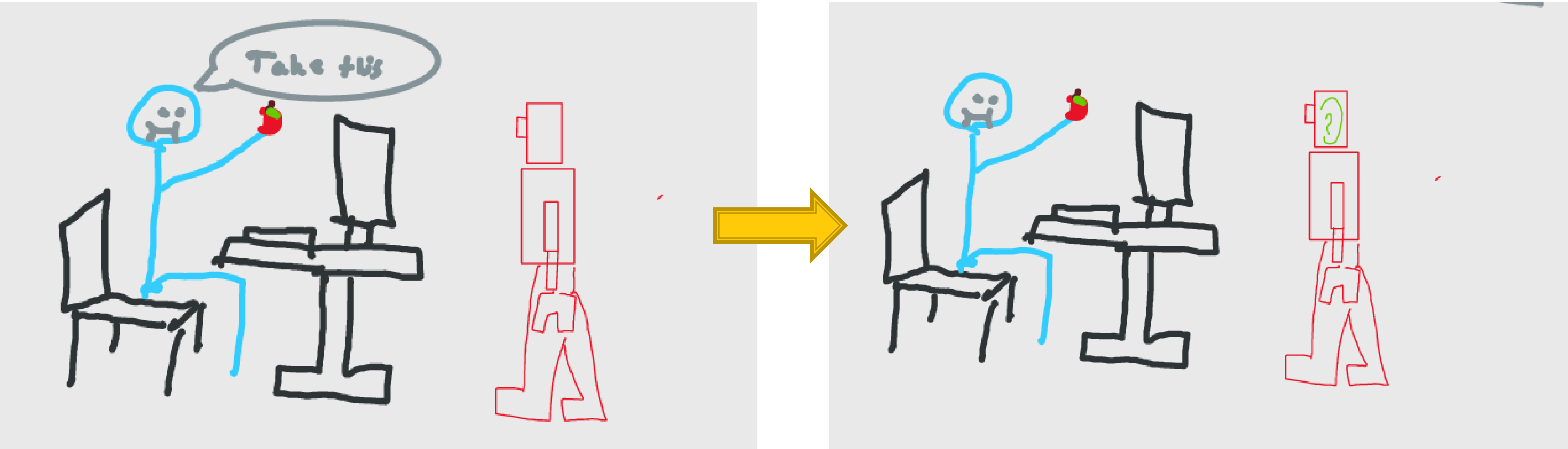
- Robot observing human manipulation





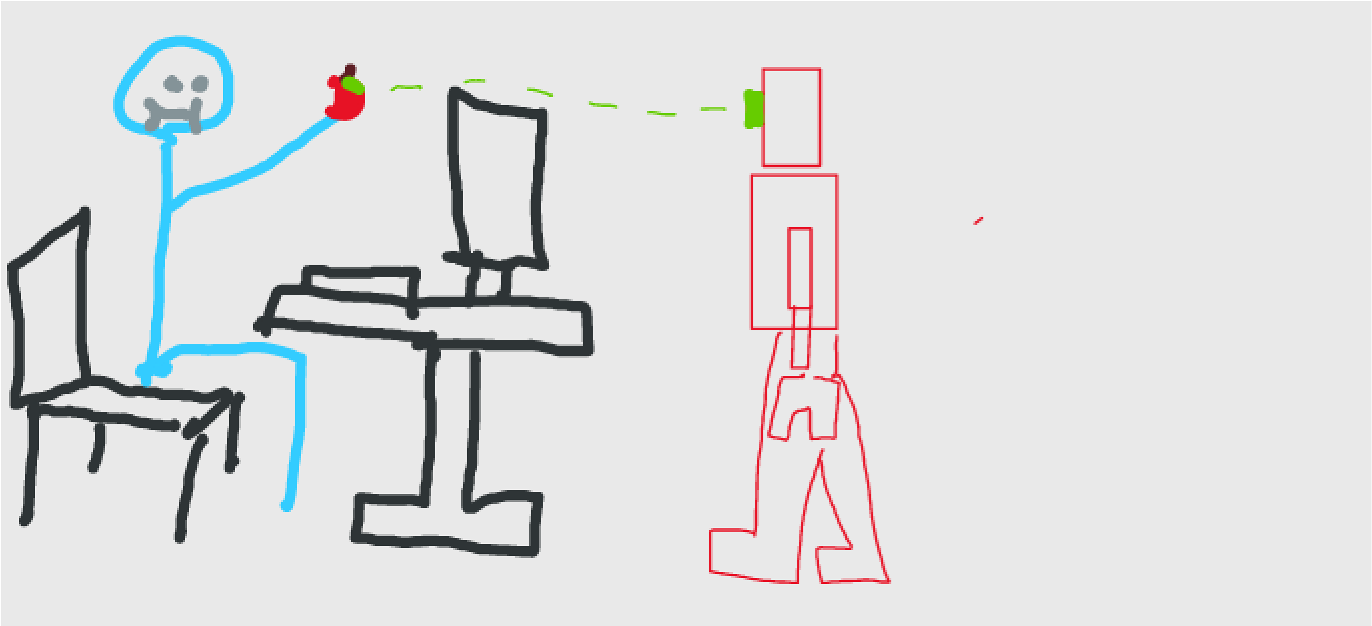
Example

- Communicate Intent



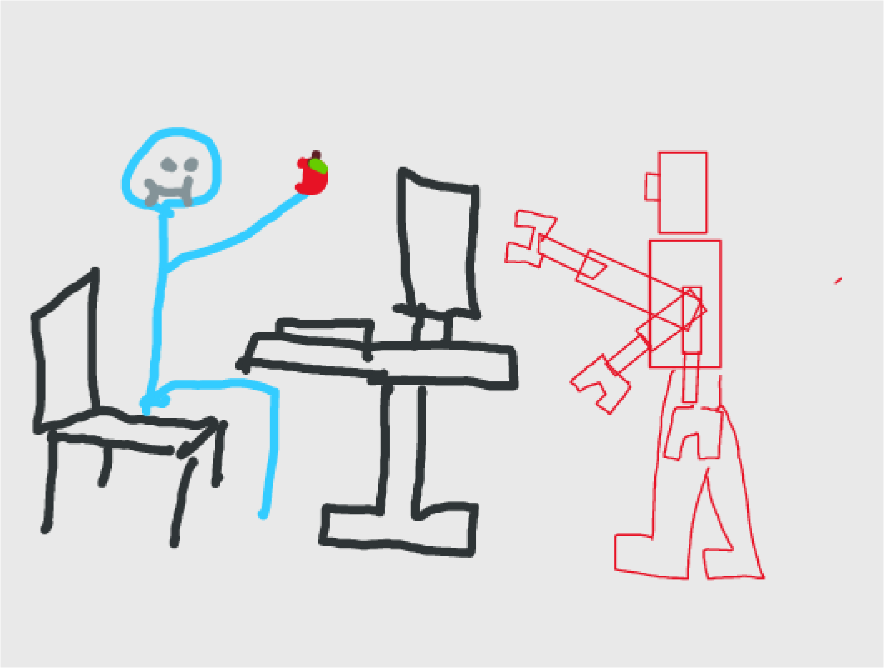
Example

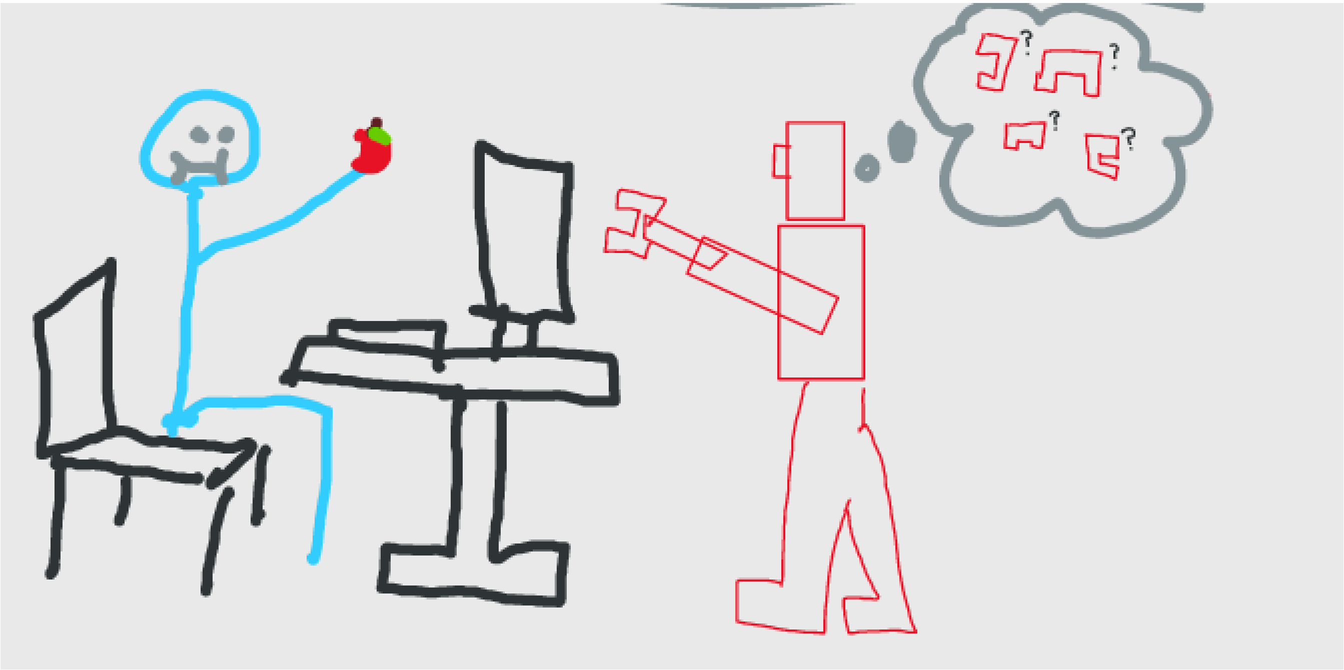
- Identifying and tracking object



Example

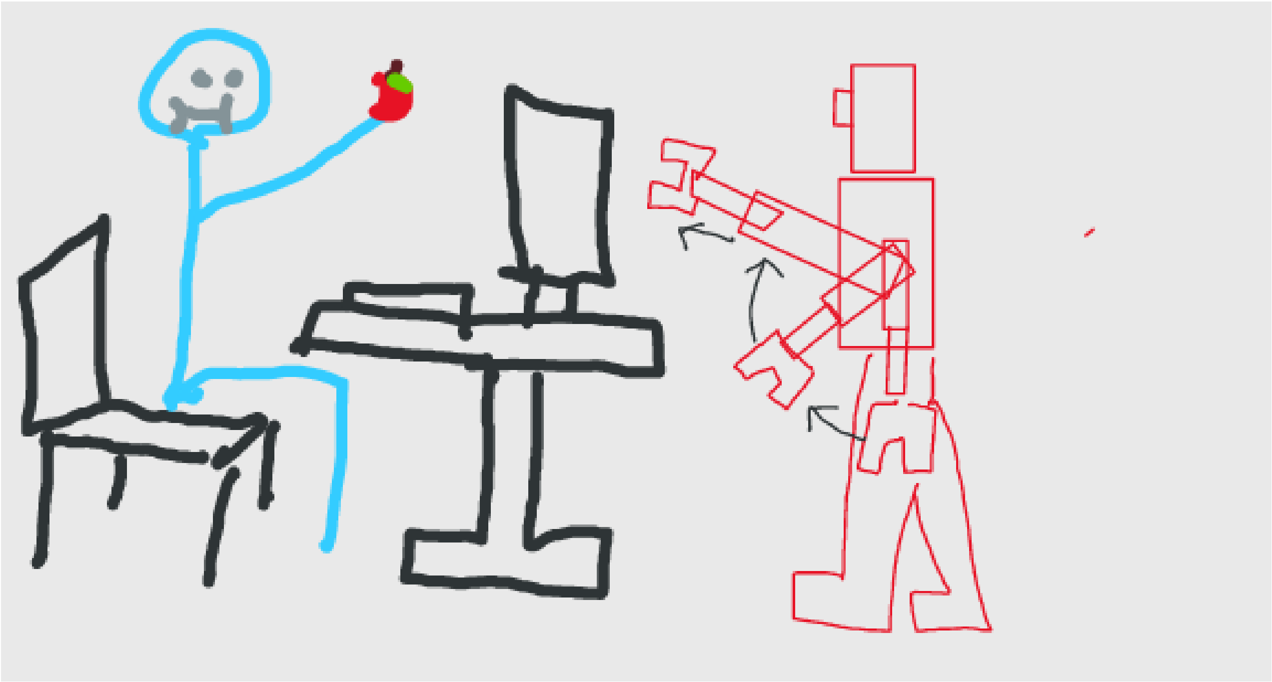
- Planning reaching and grasping



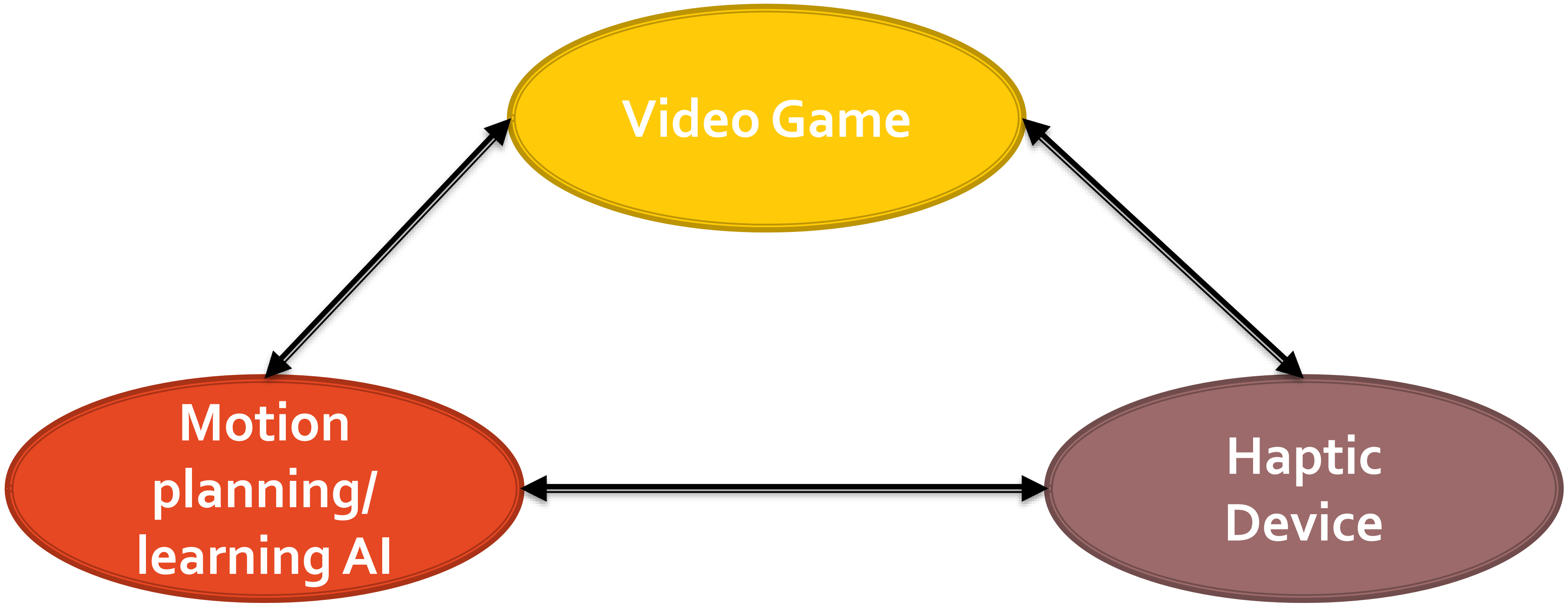


Example

- Trajectory control



Project 6 – Online motion planning in dynamic virtual environment



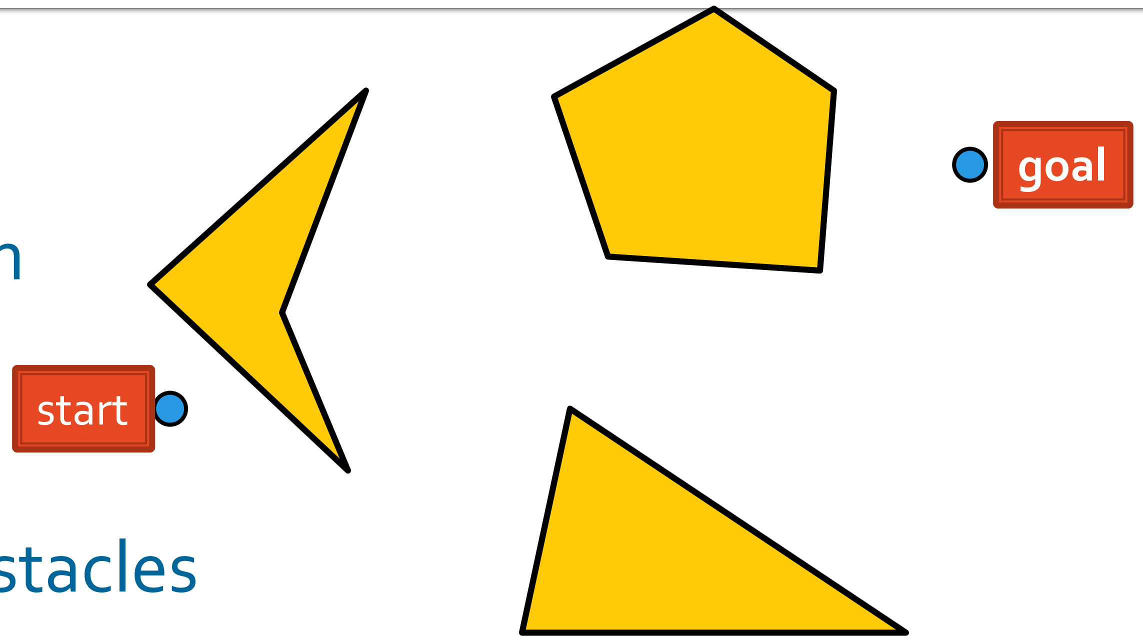
Path Planning for Point Robot

Overview

- Platform
- Course projects

Problem setup

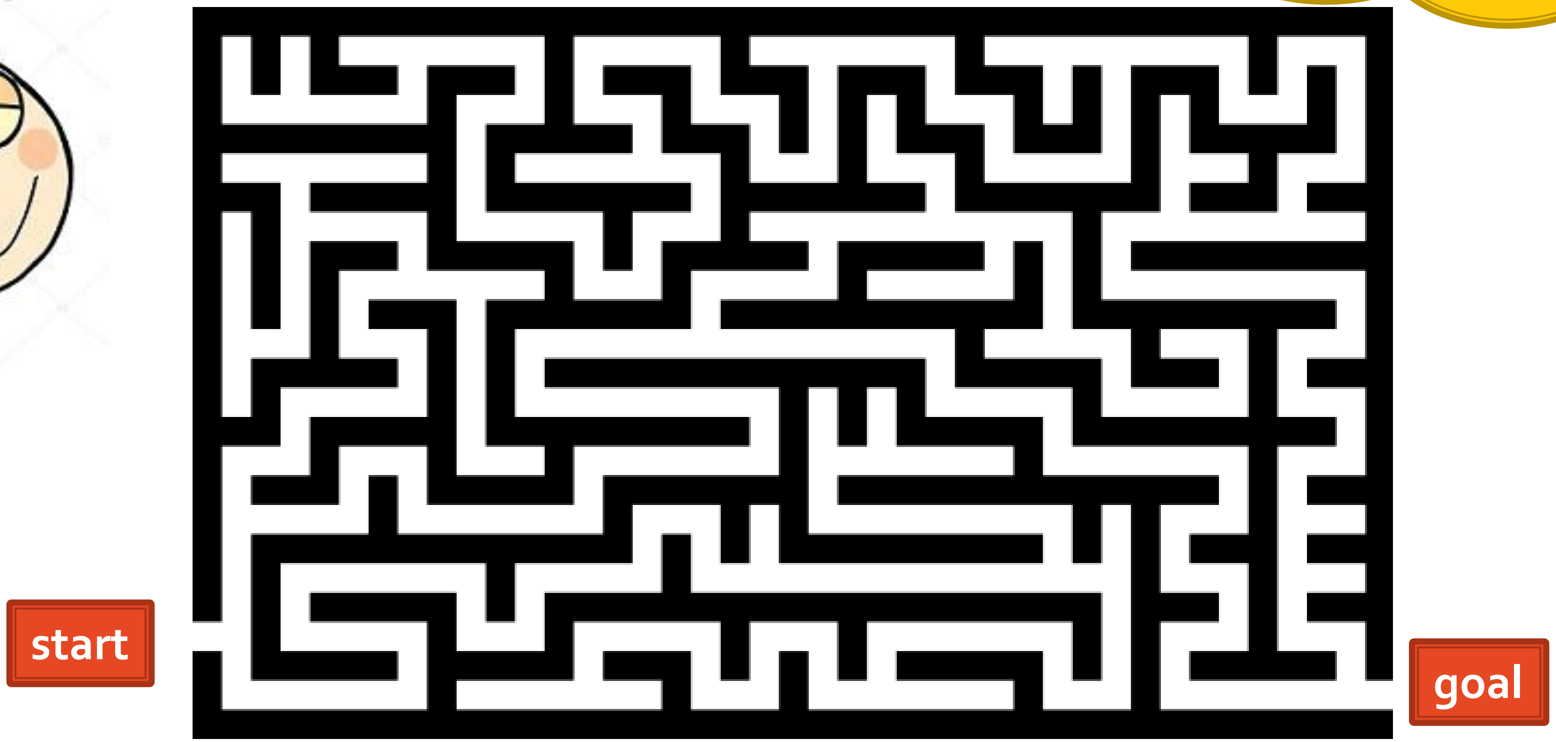
- Robot description
  - **Point robot**, of time-varying position
- Environment & robot Geometry
  - **2D** environment, with **polygonal** obstacles
- Objective
  - Find a collision-free path from start to goal



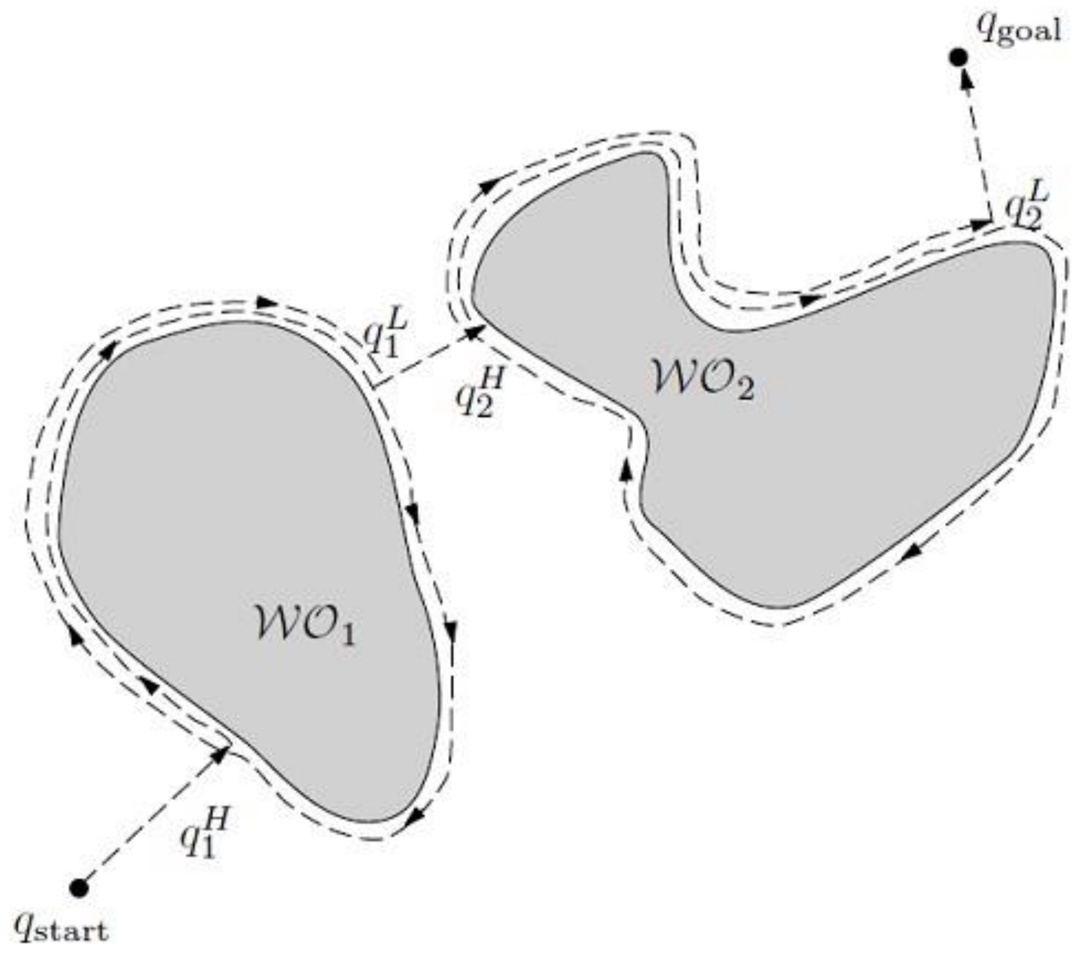
How does a bug navigate a 2D maze?

- Think as if you are a bug

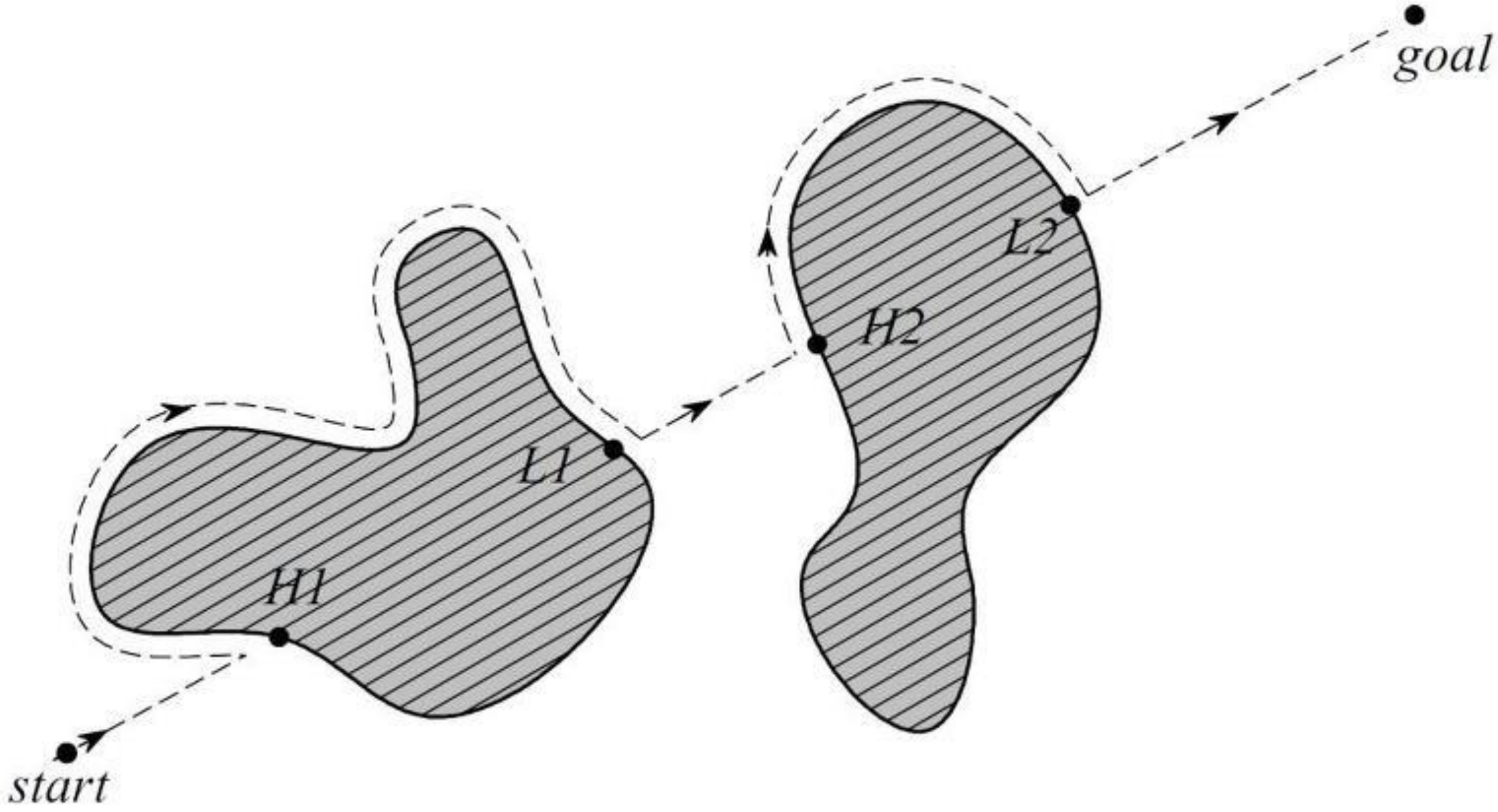
For now you have a ticket for speeding so you can not fly



Bug 1 Algorithm



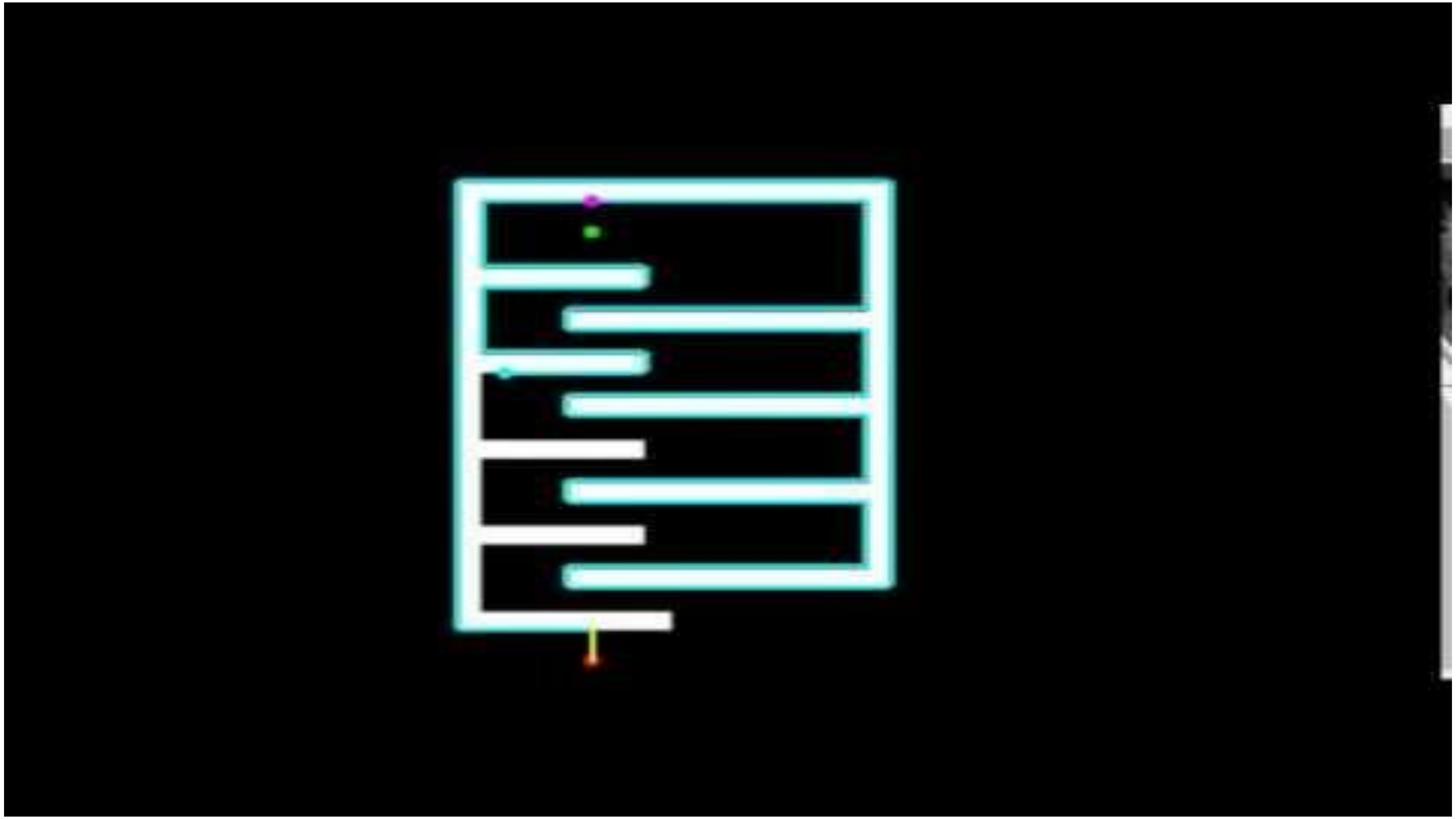
Bug 2 Algorithm



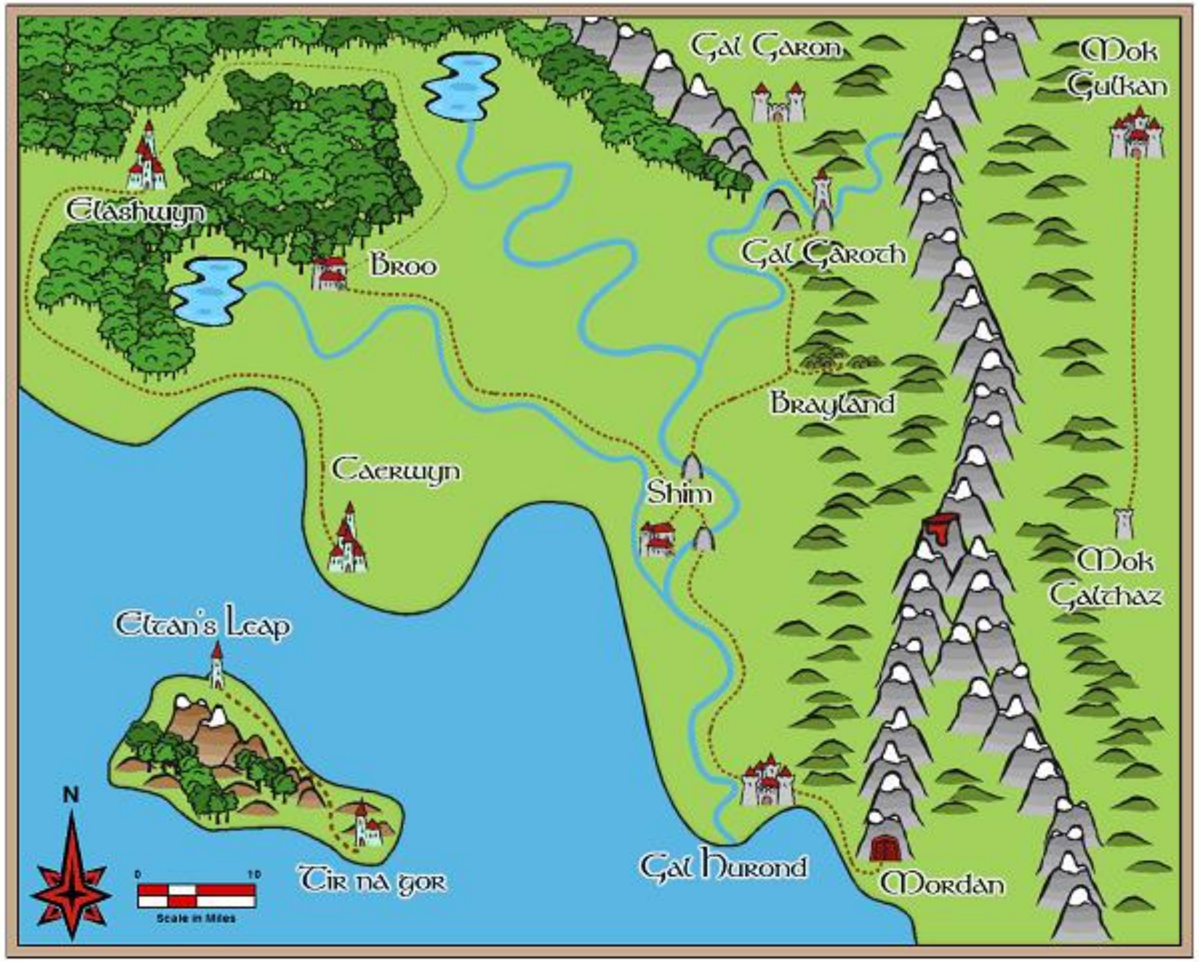
Bug 1 VS Bug 2

- Which is better?
- How to measure the algorithm efficiency?

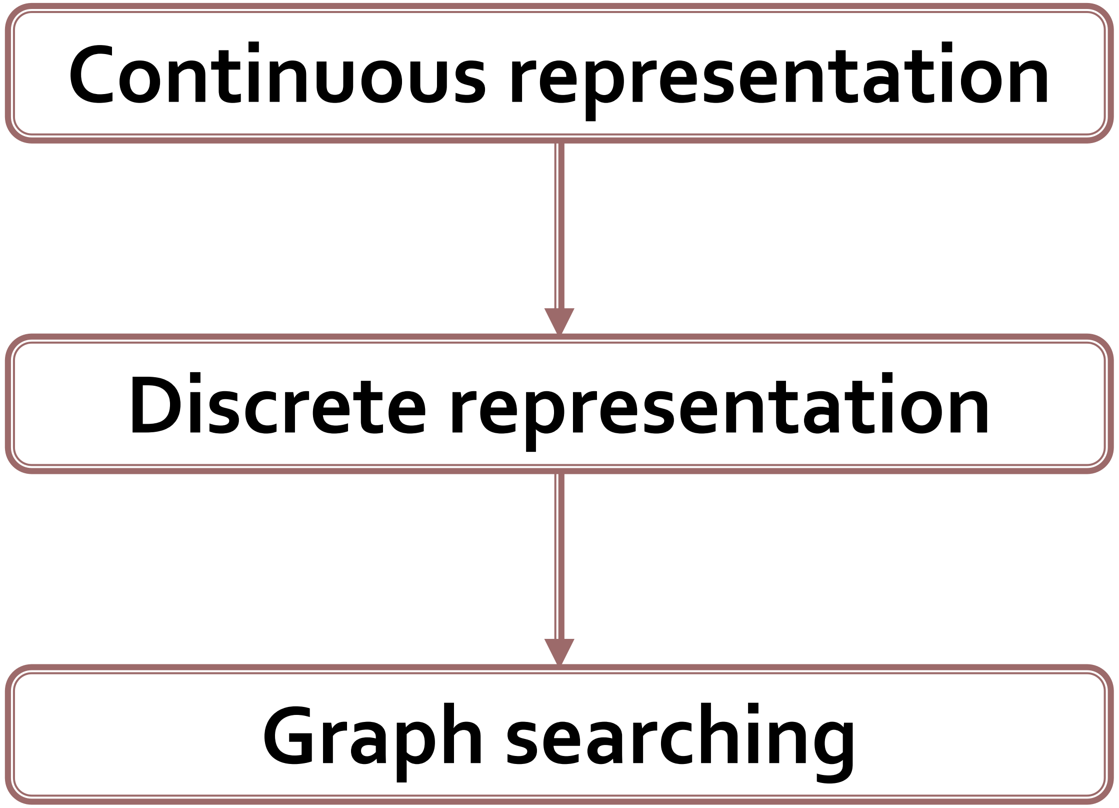
Comparison of Bug 1 and 2 algorithms



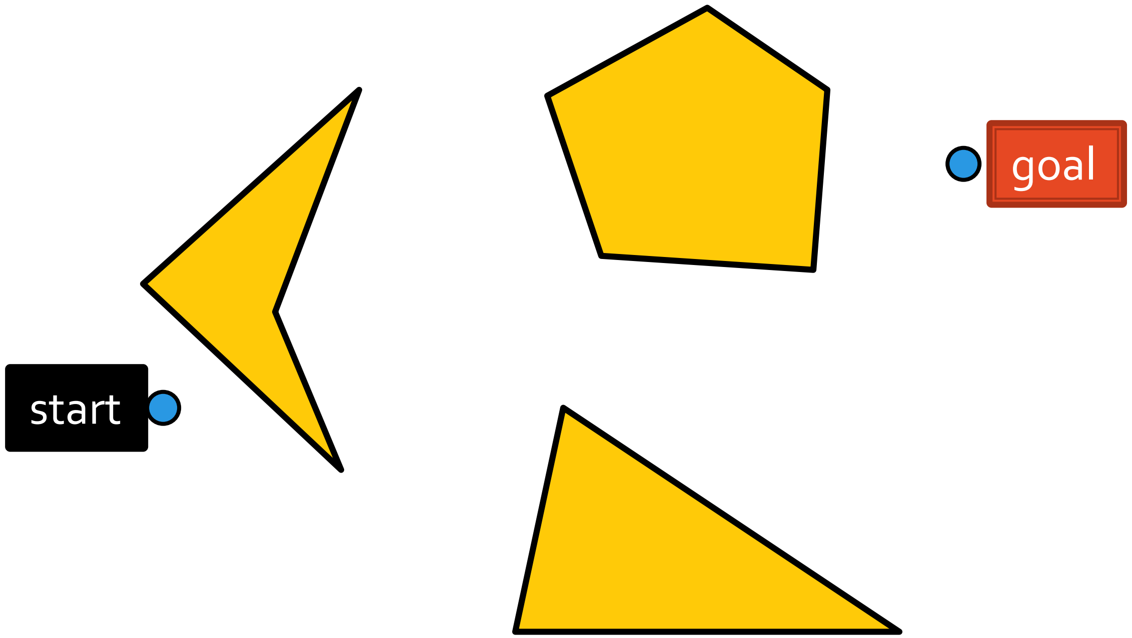
Navigation with more global information



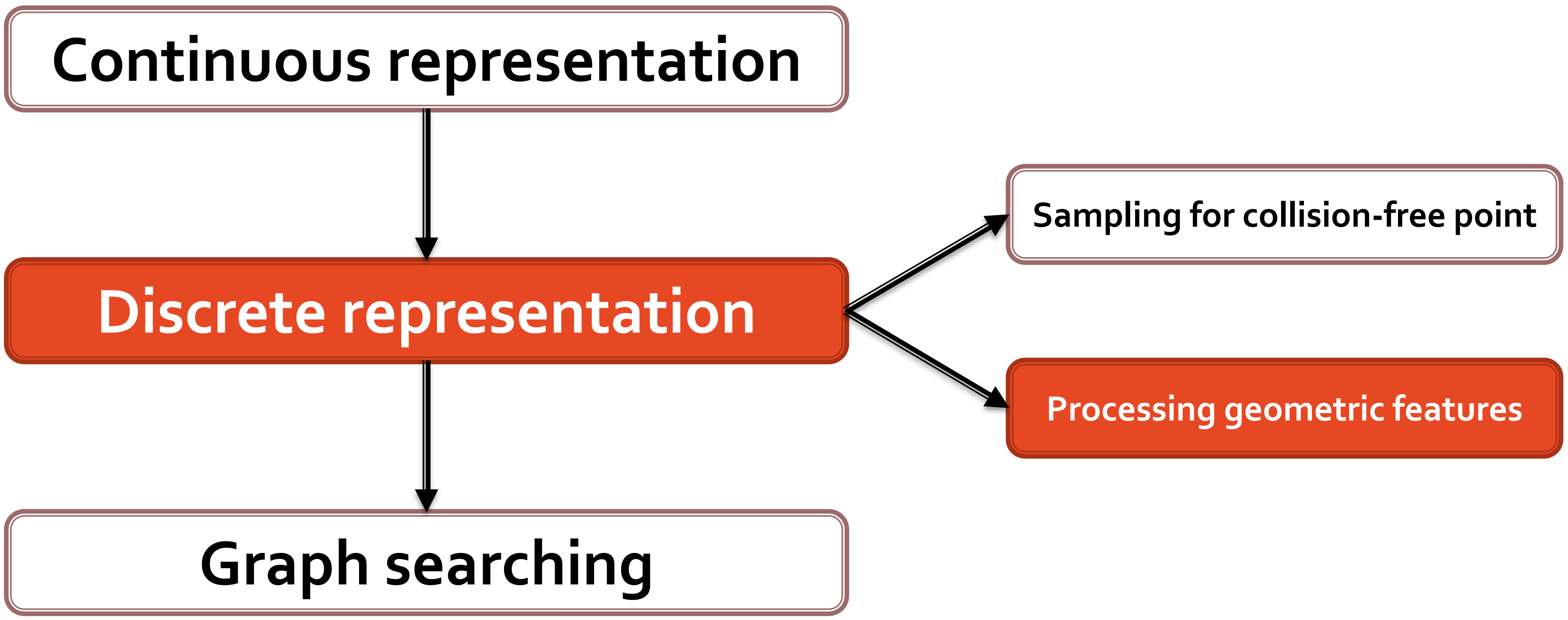
Framework of a 2D navigation problem



Continuous representation

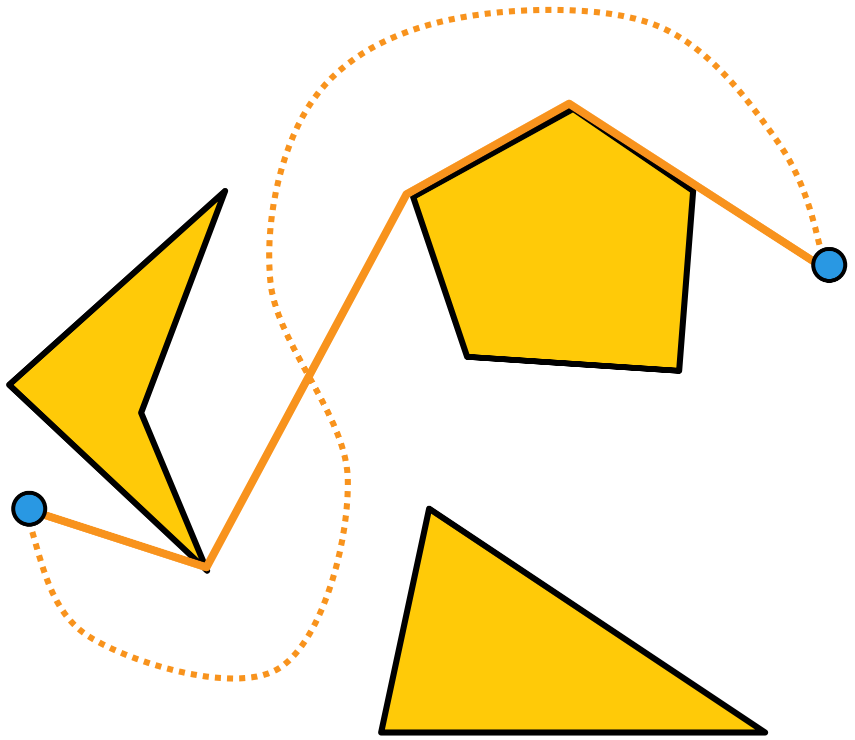


Framework


Continuous representation
Discrete representation
Sampling for collision-free point
Processing geometric features
Graph searching

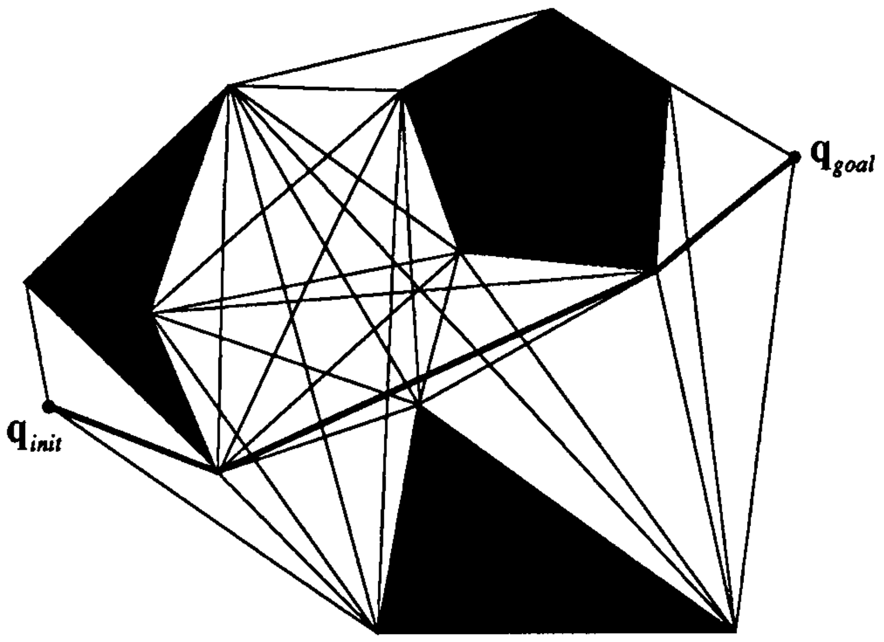
Visibility map

- If a collision-free path exists
  - There must be a piecewise linear path that bends only at the obstacles vertices



Visibility Graph

- Nodes
  - q_{init}, q_{goal}, obstacle vertices
- Edges
  - Obstacle edges
  - No intersection with obstacles



Naïve Algorithm for Computing Visibility Graph

Input: q_{init}, q_{goal}, polygonal obstacles
Output: visibility graph G

```
1: for every pair of nodes u,v
2:   if segment(u,v) is an obstacle edge then
3:      insert edge(u,v) into G;
4:   else
5:      for every obstacle edge e
6:         if segment(u,v) intersects e
7:            go to (1);
8:      insert edge(u,v) into G.
```

Running time?

```
for every pair of nodes u,v
 if segment(u,v) is an obstacle edge then
   insert edge(u,v) into G;
 else
   for every obstacle edge e
     if segment(u,v) intersects e
       go to (1);
   insert edge(u,v) into G.
```

$O(n^2)$

$O(n)$

$O(n)$

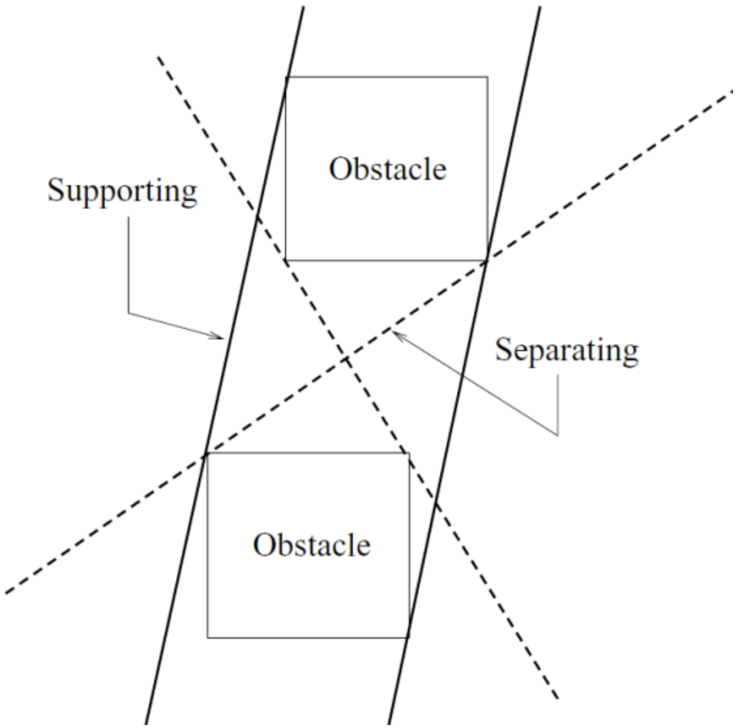
- Running time?

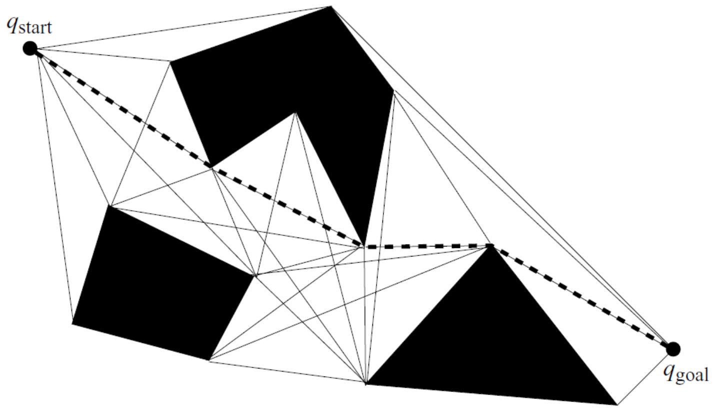
  O(n^3)

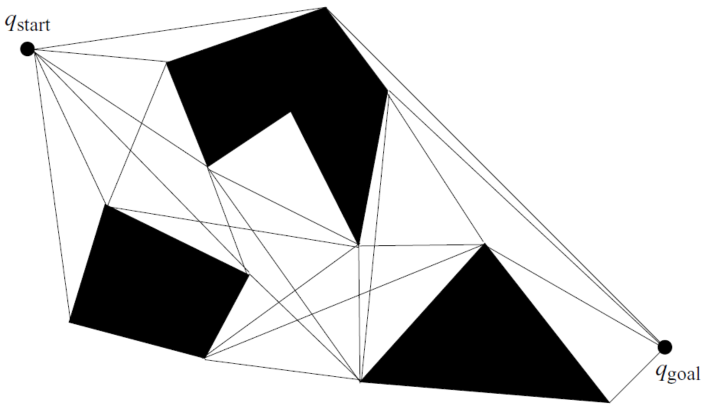
- More efficient algorithm?
  - Sweep-line algorithm – **O(n^2 log n)** (see Principles 5.1.2)

Reduced Visibility Graph

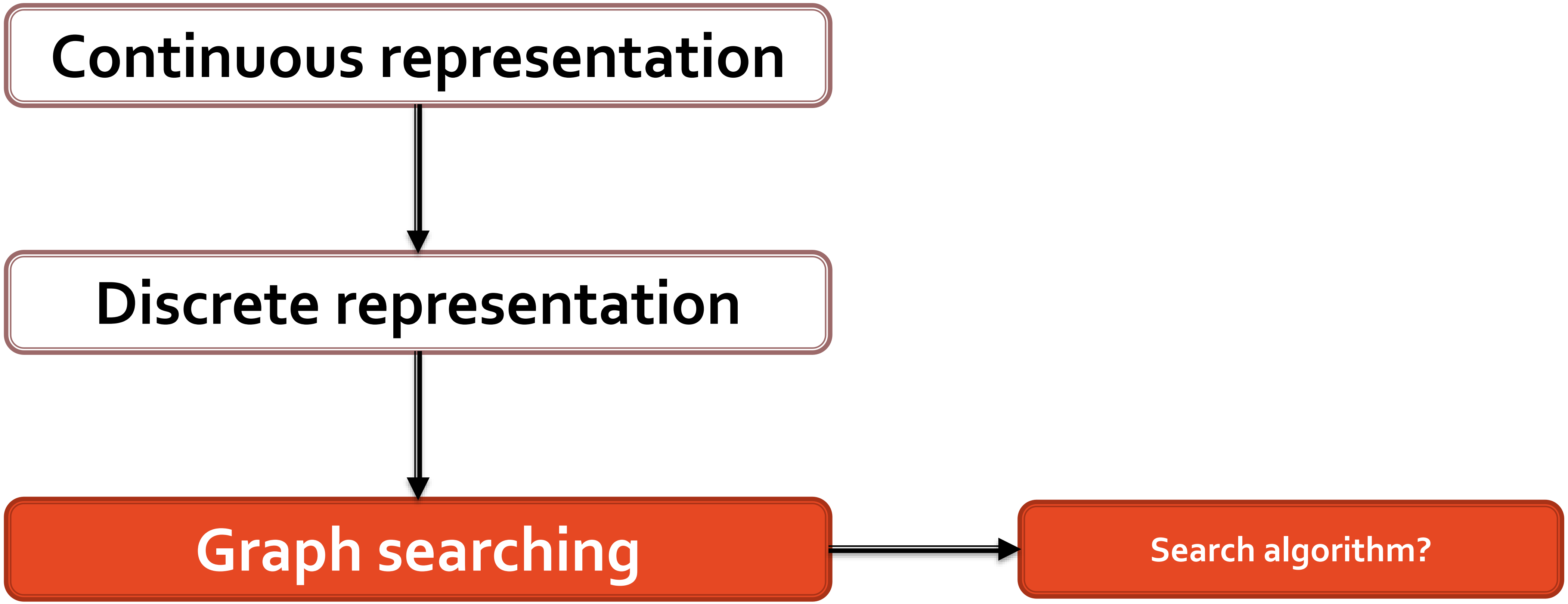
- Construct visibility graph from
  - Supporting and separating lines



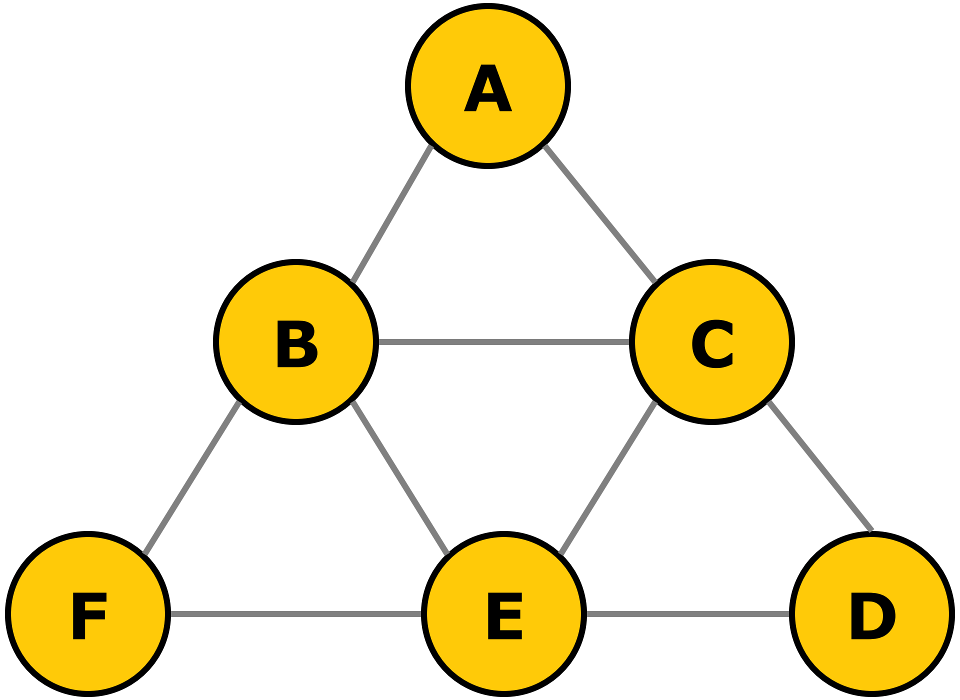




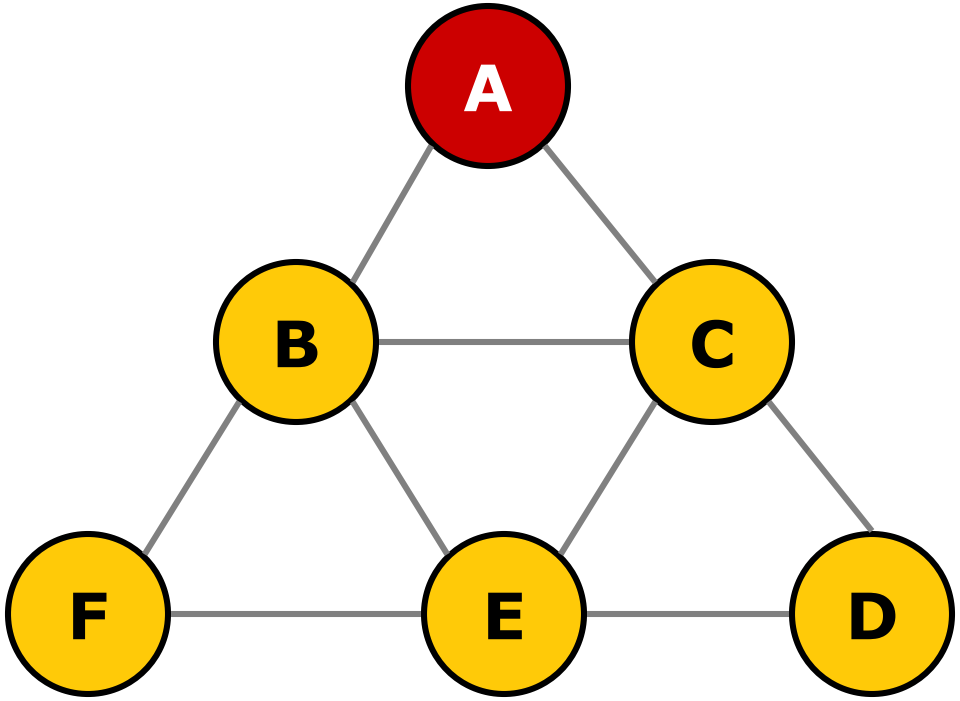
Framework


Continuous representation
Discrete representation
Graph searching
Search algorithm?

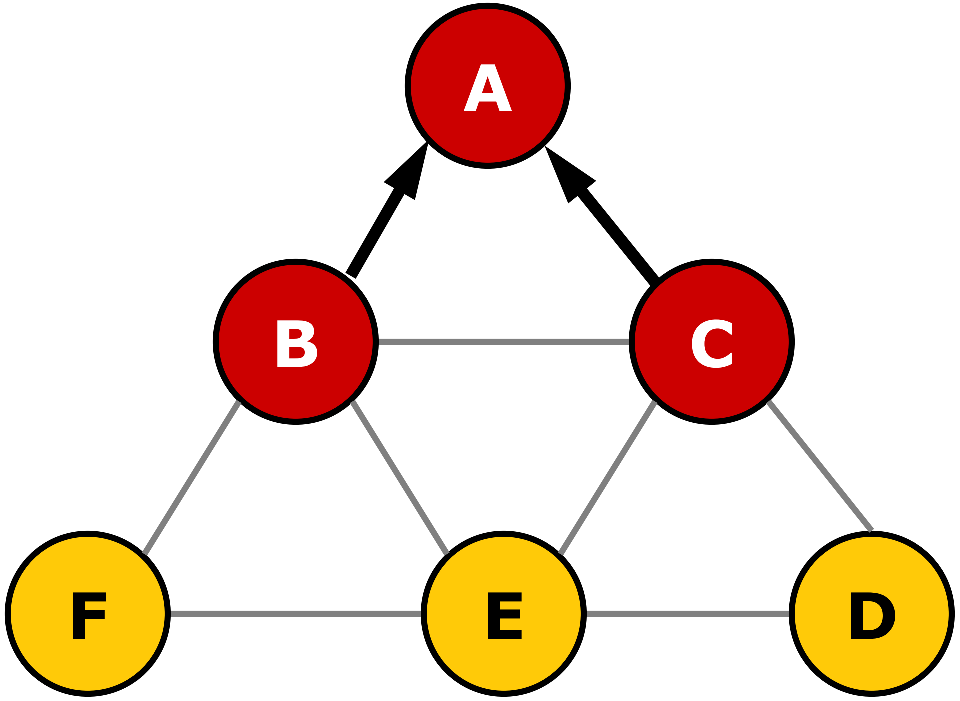
Breadth-first search



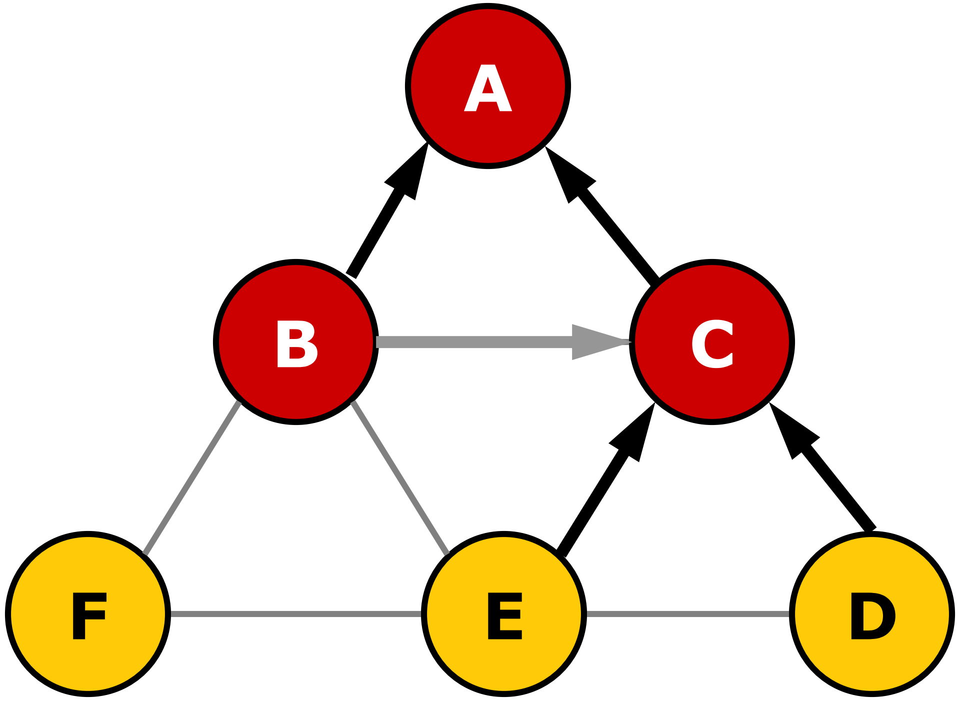
Breadth-first search



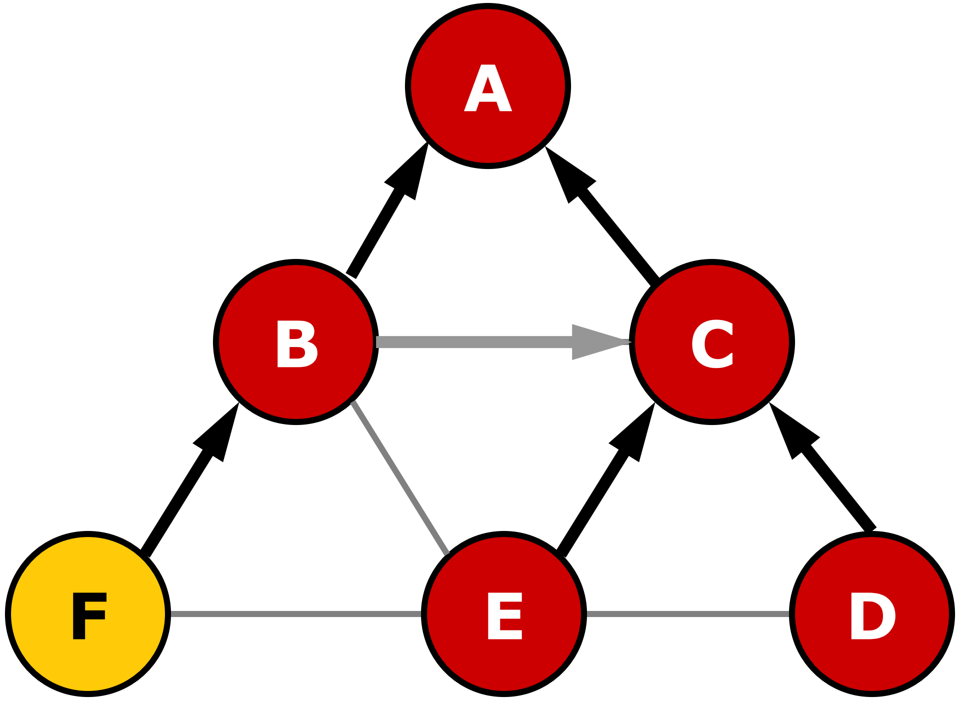
Breadth-first search



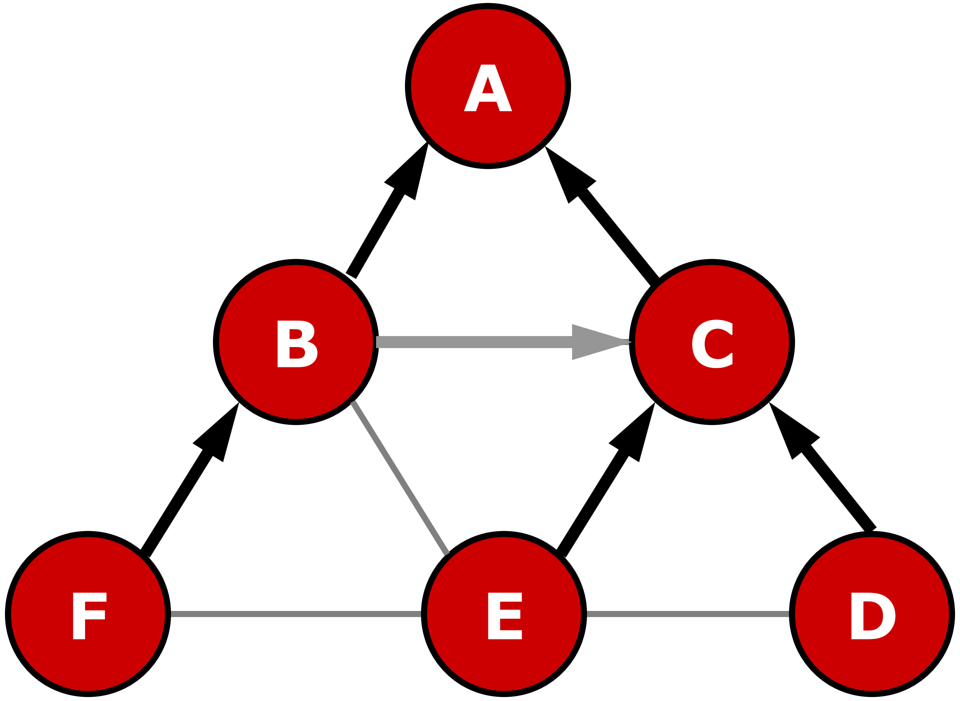
Breadth-first search



Breadth-first search



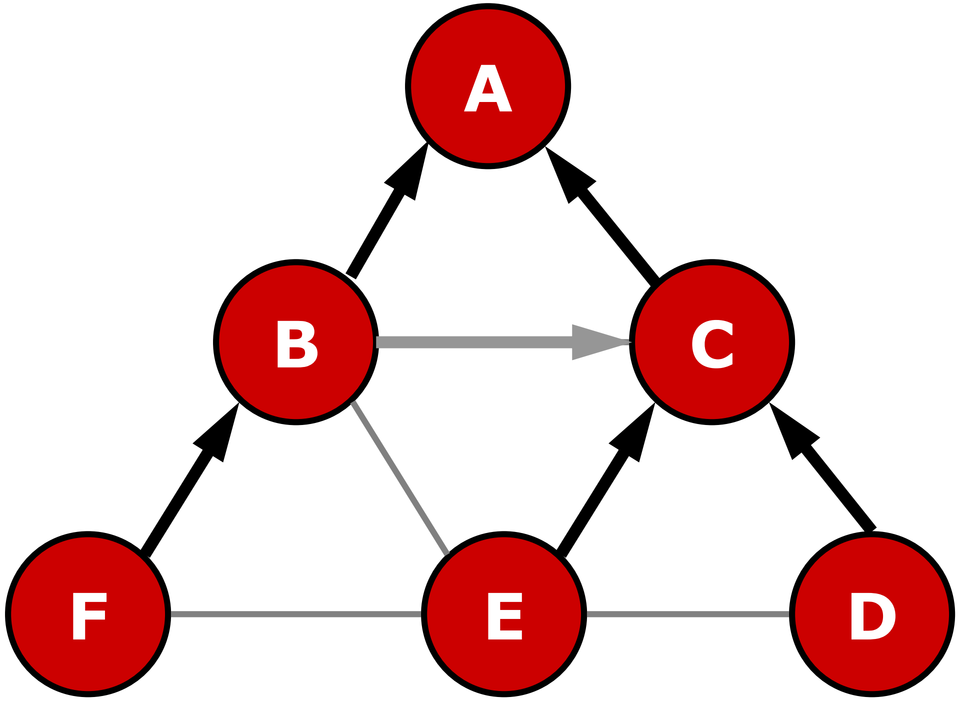
Breadth-first search



Breadth-first search

Input: q_{init}, q_{goal}, visibility graph G
Output: a path between q_{init} and q_{goal}

```
Q = new queue;
Q.enqueue(q_init);
mark q_init as visited;
while Q is not empty
  curr = Q.dequeue();
  if curr == q_goal then
    return curr;
  for each w adjacent to curr
     if w is not visited
       w.parent = curr;
       Q.enqueue(w)
       mark w as visited
```



Other graph search algorithms

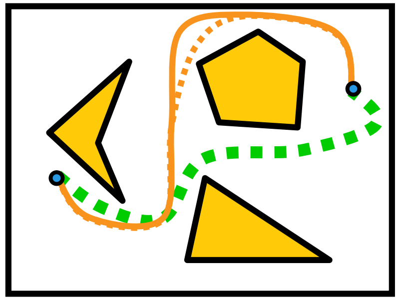
- Depth-first
  - Explore newly-discovered nodes first
  - Guaranteed to generate shortest path in the graph?

No

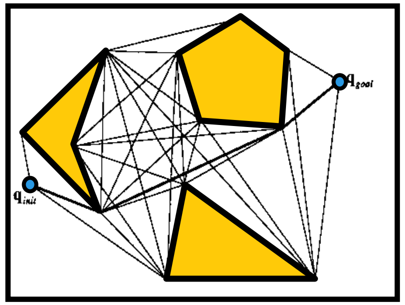
- Dijkstra's Search
  - Find shortest paths to the goal node in the graph from the start
- A*
  - Heuristically-guided search
  - Guaranteed to find shortest path

Recap

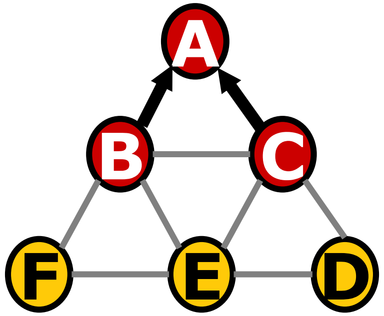
Continuous representation



Discrete representation



Graph searching



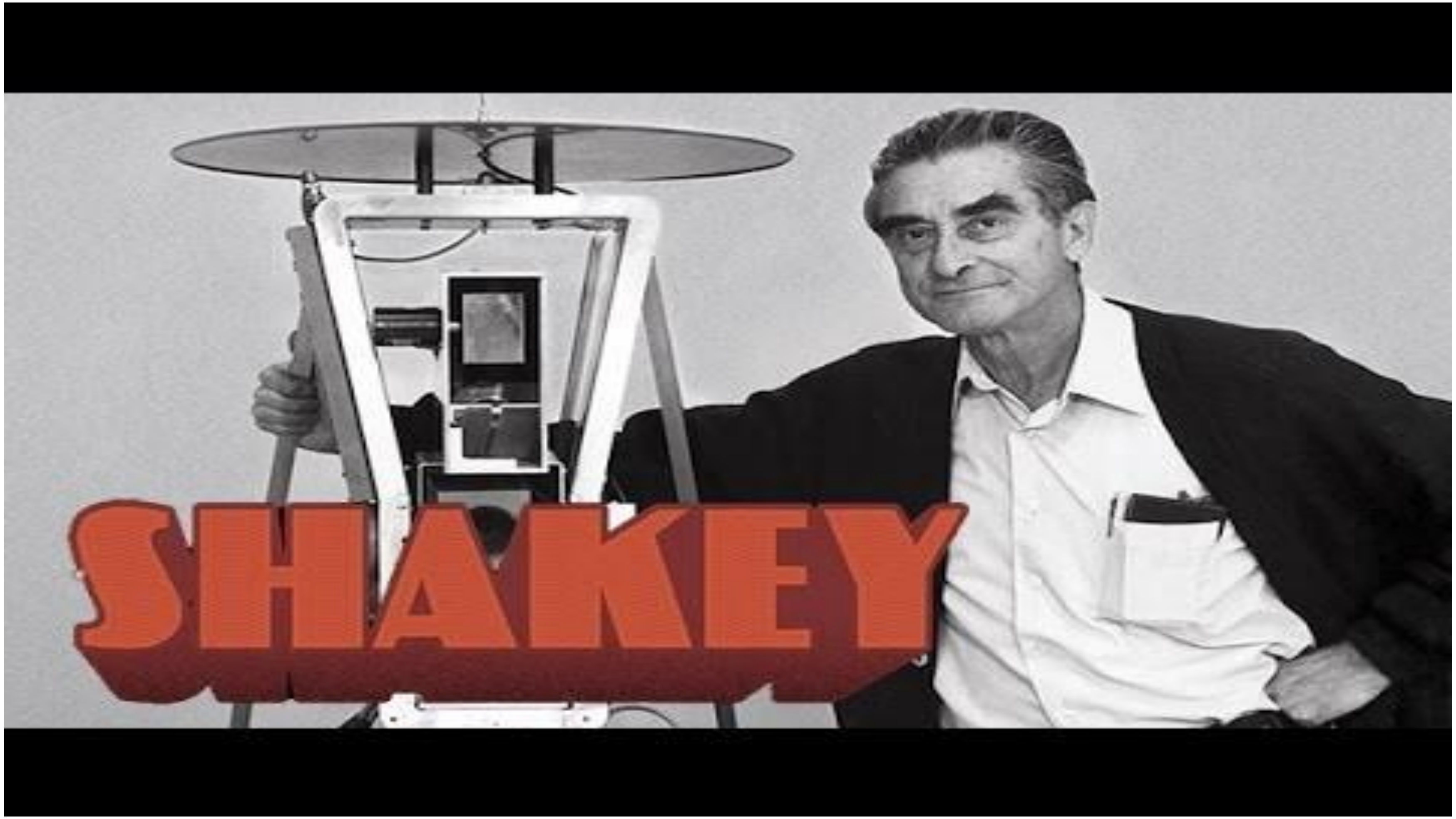
Recap

- Running time
  - Compute the visibility graph – Naïve method – O(n^3)
    - An optimal $O(n^2)$ time algorithm exists.

- Space?
  - Store graph as adjacency list or adjacency matrix

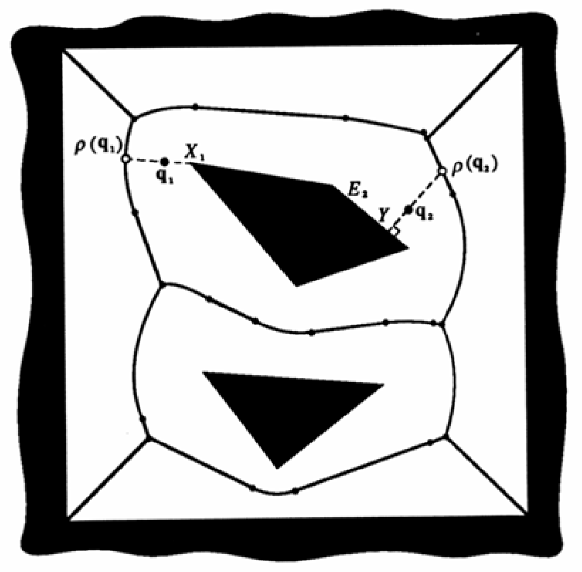
O(n^2)

Application to Shakey the Robot (1969) Navigation using visibility map

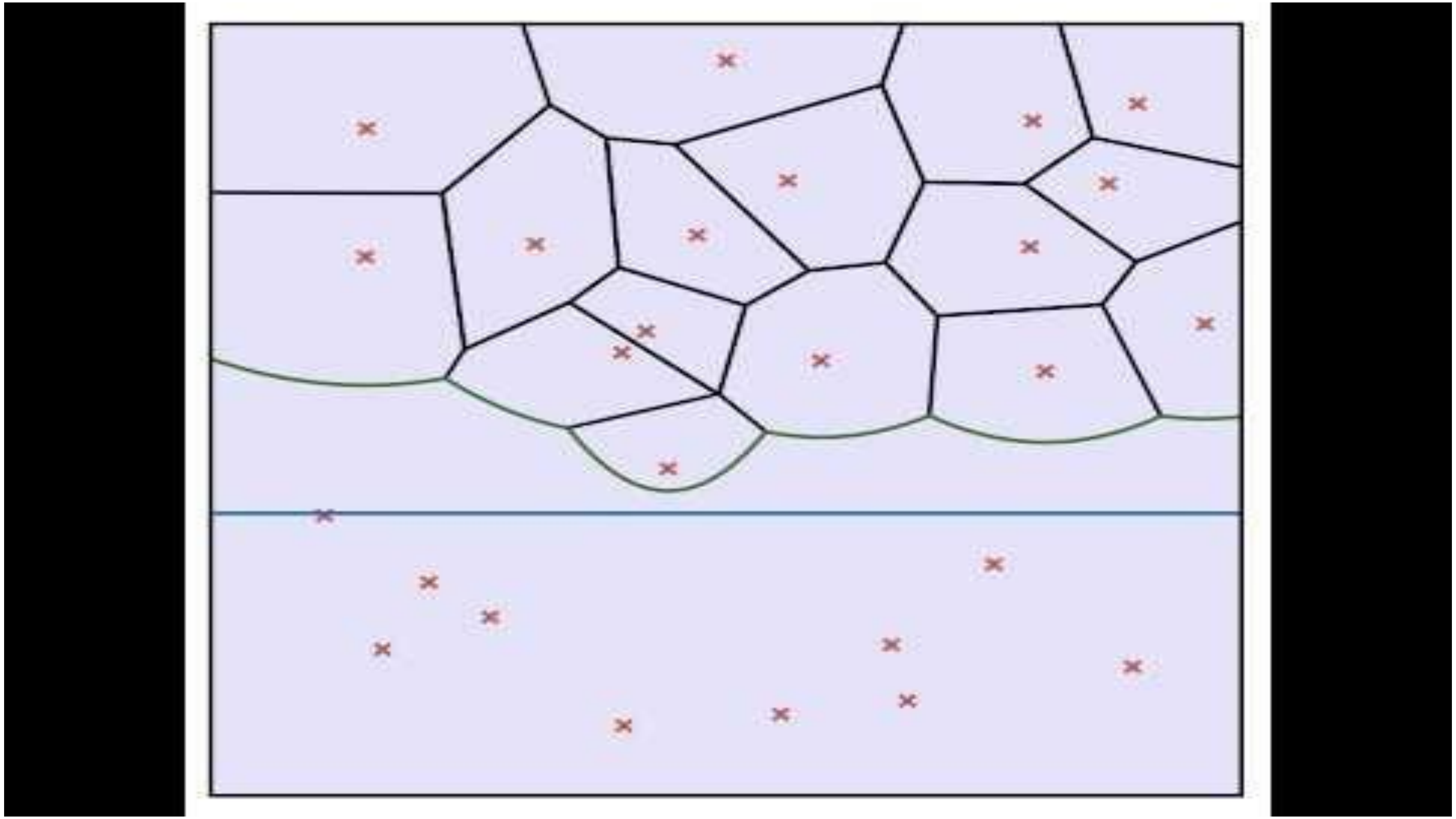


Other ways for building roadmaps?

- **Voronoi graph**
  - Introduced by **computational geometry** researchers.
  - Generate paths that **maximizes clearance**.



Voronoi map

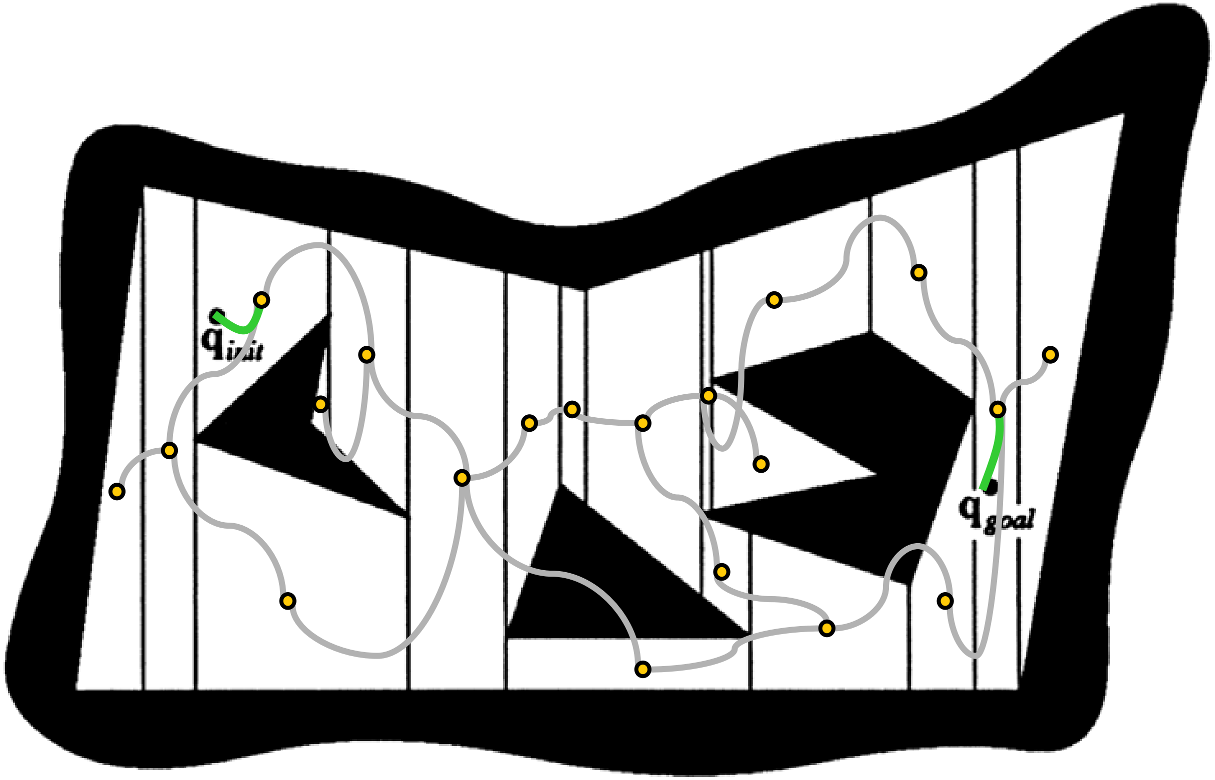


Cell decomposition

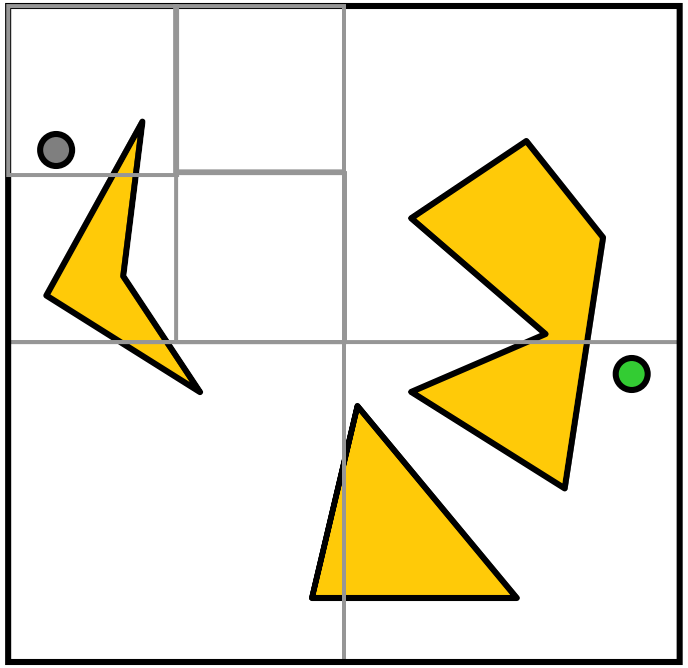
- Exact methods
  - 2D - Trapezoids, triangles, etc.
  - Adjacency map

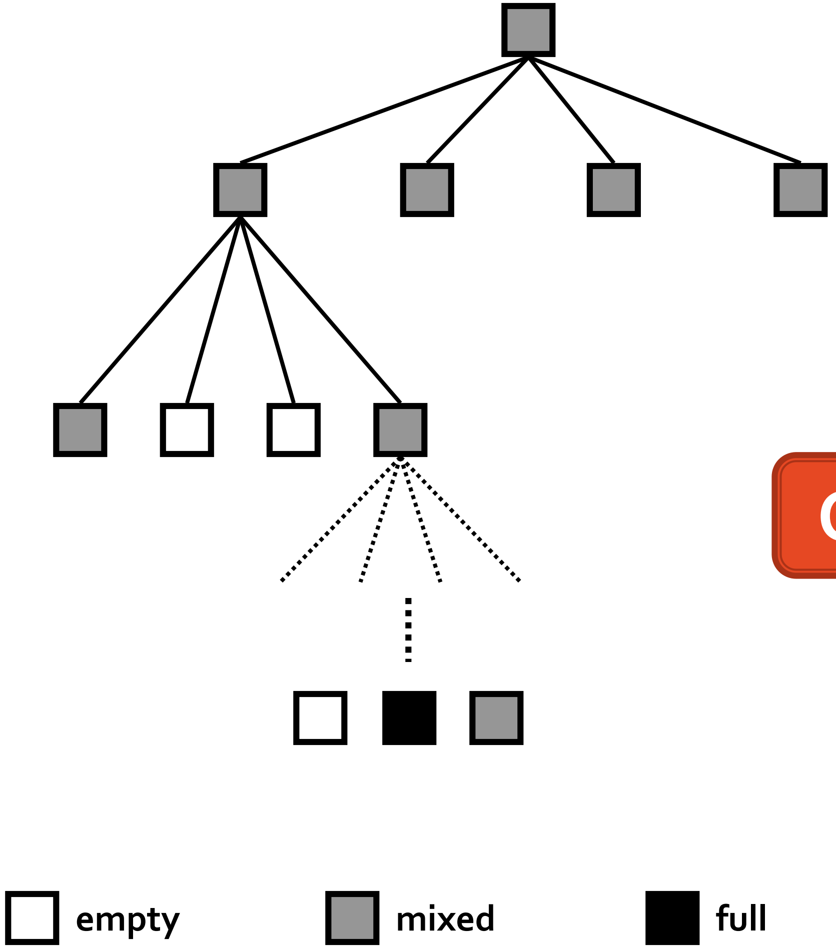
- Approximate methods
  - Decompose space into cells usually have **simple, regular** shapes
  - Facilitate **hierarchical** space decomposition

Cell decomposition – Exact method



Cell decomposition – Approximate method





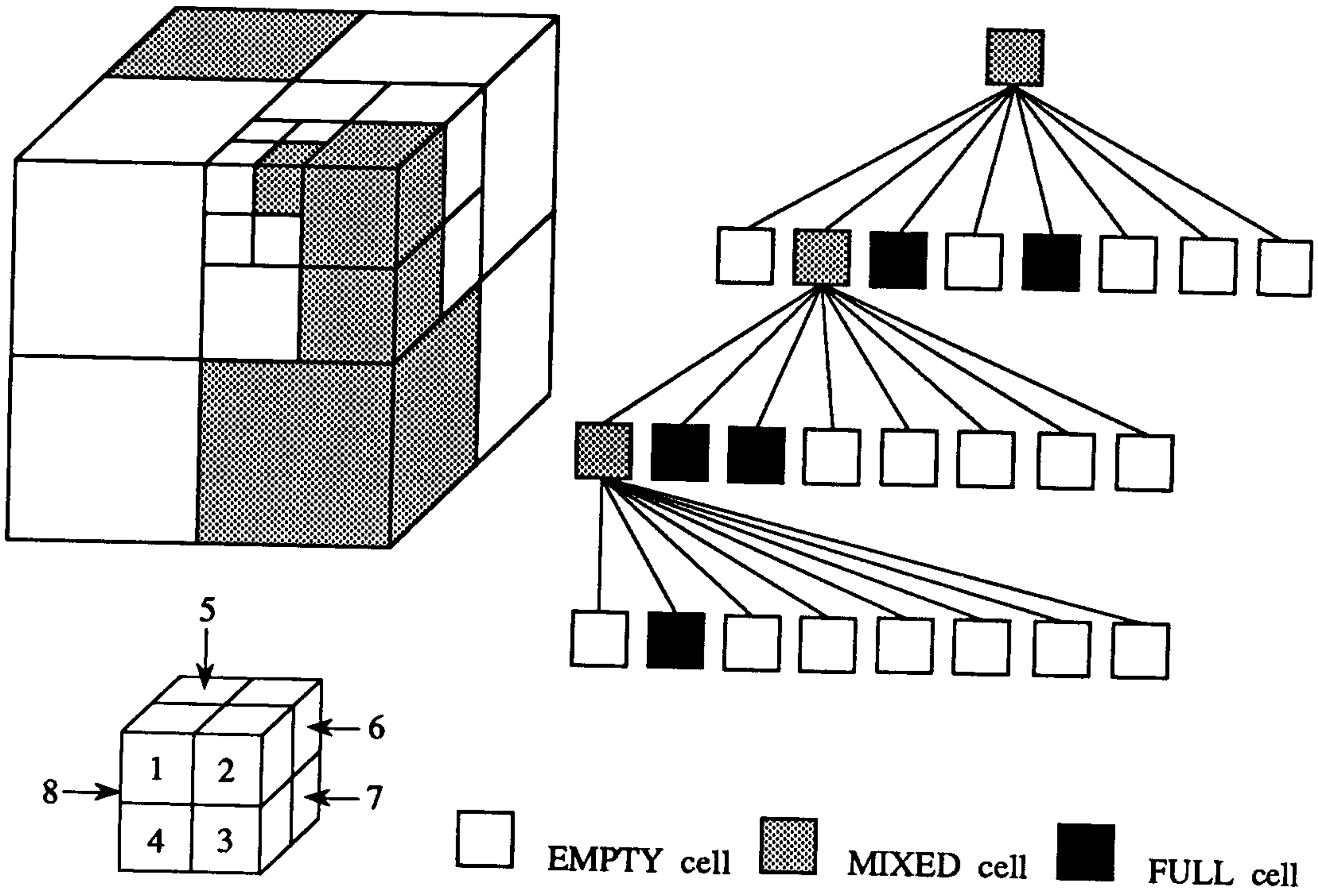
Quad-tree

Hierarchical Decomposition

- Strategy
  - (1) **Decompose** the free space into cells
  - (2) **Search** for a sequence of **mixed or empty cells** that connect the initial and goal positions
  - (3) **Further decompose** the **mixed**
  - (4) Repeat (2) and (3) until a sequence of empty cells is found

Other trees?

Octree



Assignment 2 – Due on Wed (Jan 24)

- Reading
  - Principles 5.1.2: Sweep-line algorithm
  - Individual literature review
  - Select 4 best work and 1 for presentation

Optional Assignment

- Optional
  - Accept volunteer to give a 5 min- talk on next Wed (Jan 24)
  - Topic – a cool application of motion planning
  - Send your presentation slides (notes must be included) by Monday (Jan 22) at noon
  - TA will email the selected presenter by 10 am of Tuesday (Jan 23)
  - Extra credit – use to replace one of your low-score quiz in the future

Next Friday (Jan 26) – TRINA workshop

- Objectives
  - Introduction to TRINA system (Hardware + software)
  - Help you setup workstation
- Project group will be announcement next Wednesday (Jan 24)
- Requirement
  - Each group need to bring at least one workstation with
  - Ubuntu 16 and ROS kinetic installed
  - Prefer to be dual-boot. VM can be SUPER slow for what we need

End