Lecture 17
Motion Planning II

Prof. Katie DC

Modern Robotics Ch 10.2-10.5

April 14, 2020

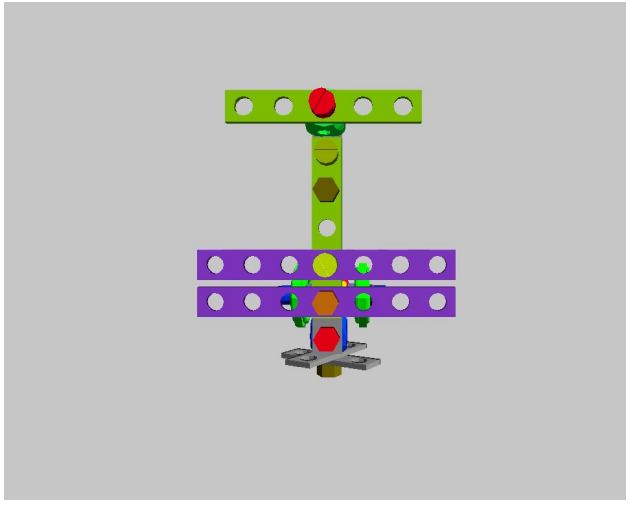
Administrivia

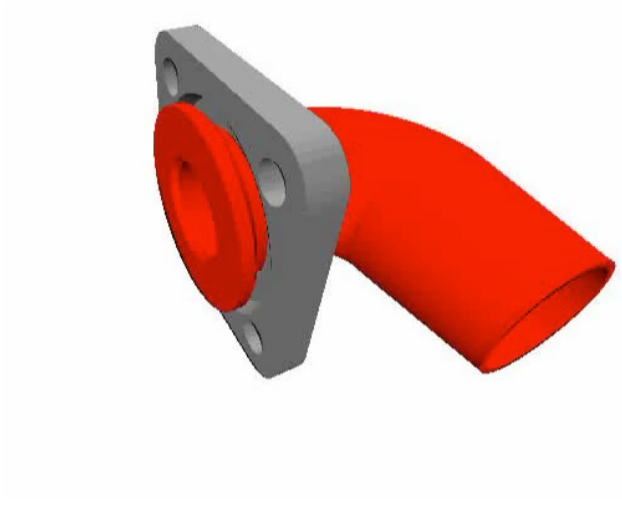
- Exam 2 regrades will be posted soon
- Exam 3 (comprehensive) in two weeks!
- Extra credit in-class activity on 5/5

Who is Nancy Amato?

- Head of the CS department and expert in motion planning
- Her paper on probabilistic planning is one of the most important papers in PRM, the first to not use uniform sampling in the configuration space
- She and her team wrote a seminal paper that shows how robot planning can be applied to protein motions (folding)

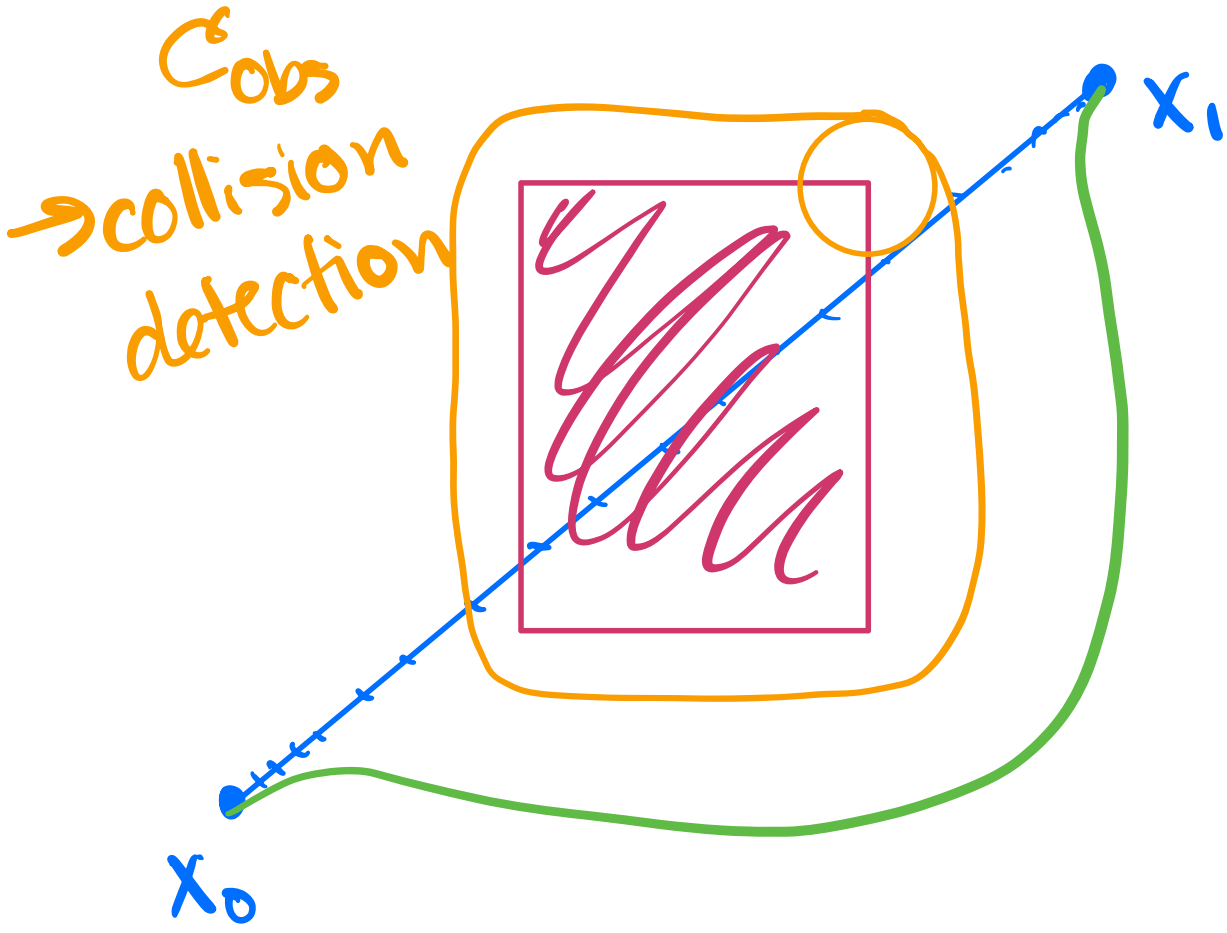
→ This line of work started a new research area in comp. biology





Motion Planning Overview

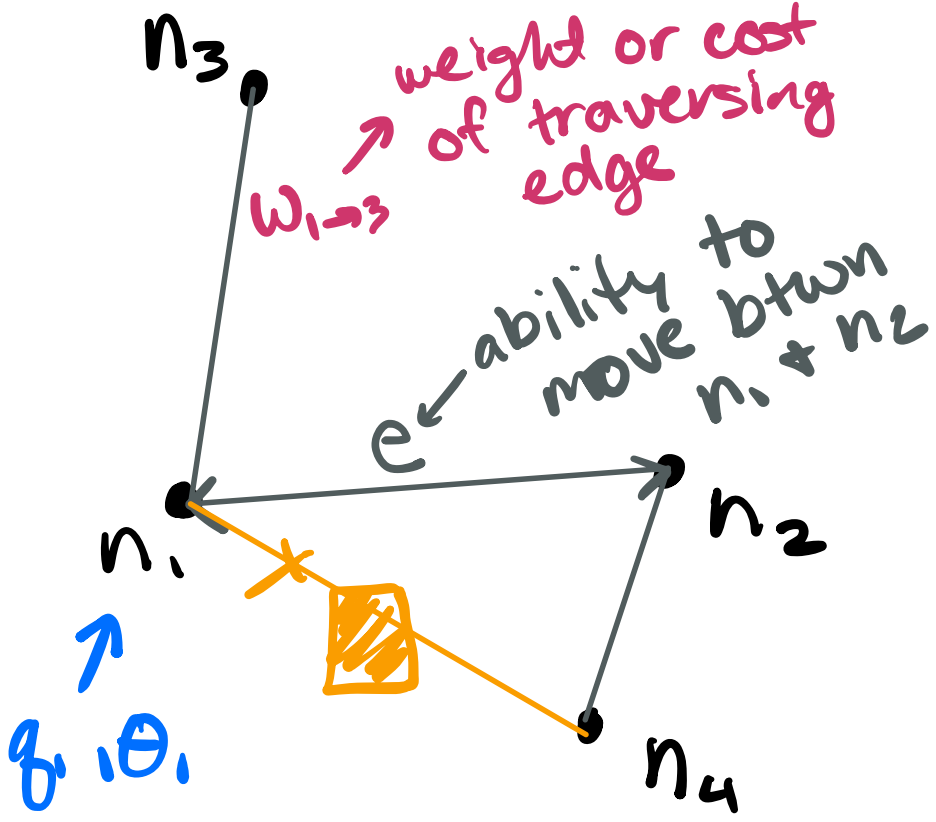
path + time spec
→ traj gen



motion planning
w/ obs avoidance

Graphs and Trees

- Motion planners often represent C-space as a **graph**
- A **graph** is a collection of **nodes** $\mathcal{N}$ and **edges** $\mathcal{E}$, where edge e connects two nodes



often rep as a matrix

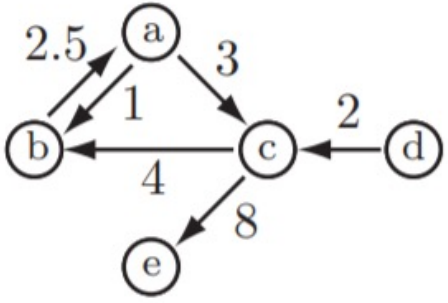
$A \in \mathbb{R}^{n \times n}$

$a_{ij} \leftarrow w_{i\to j}$

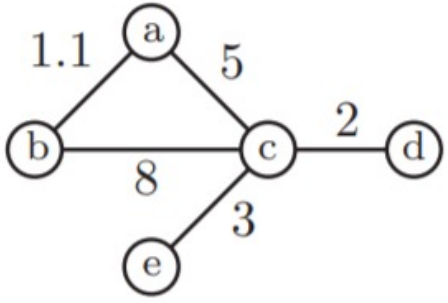
0 if no edge

Graphs and Trees

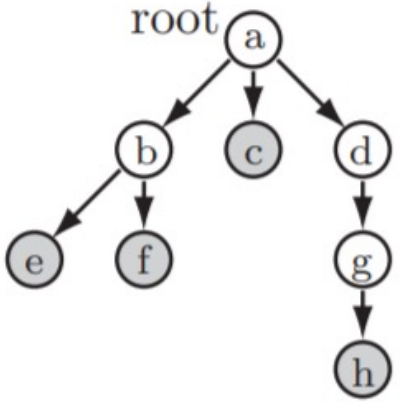
directed



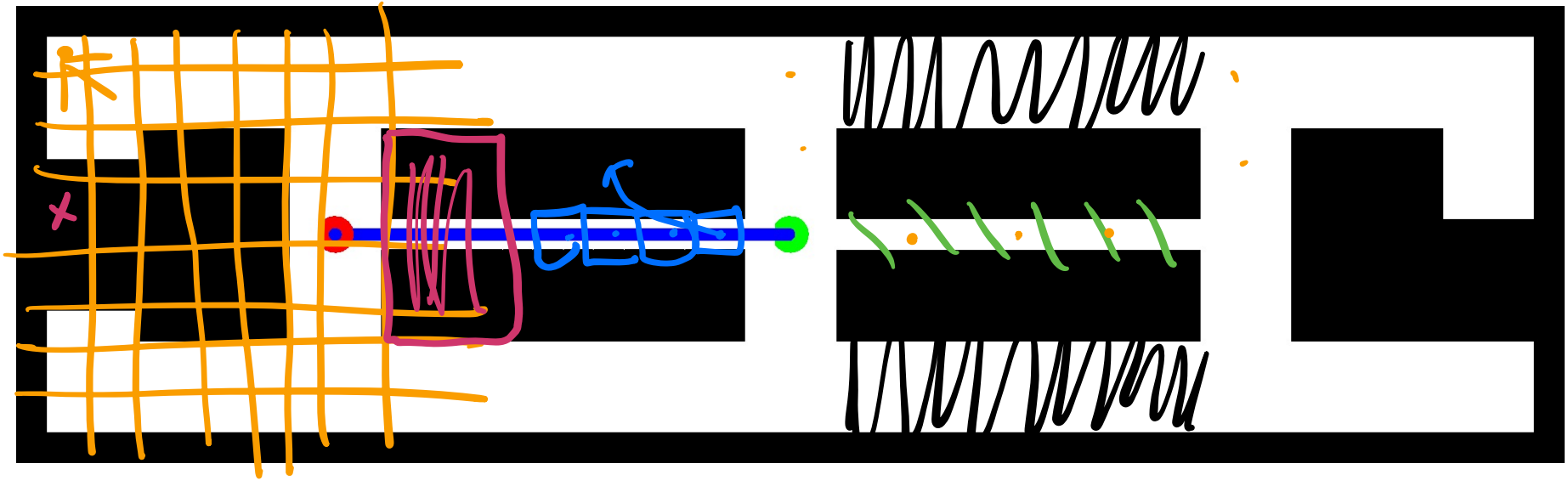
undirected



tree

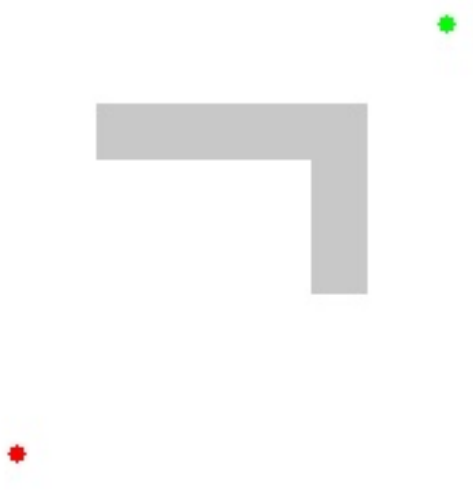


Grid-World Example

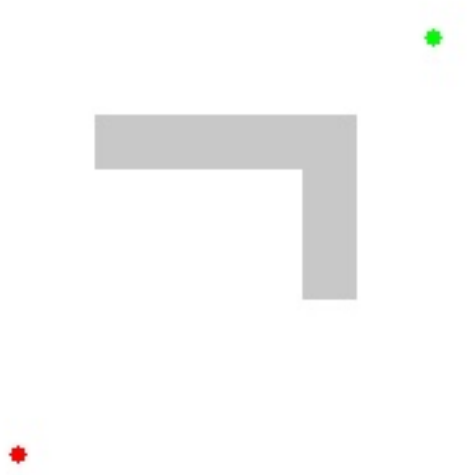


Graph Search Methods

A* search algorithm.

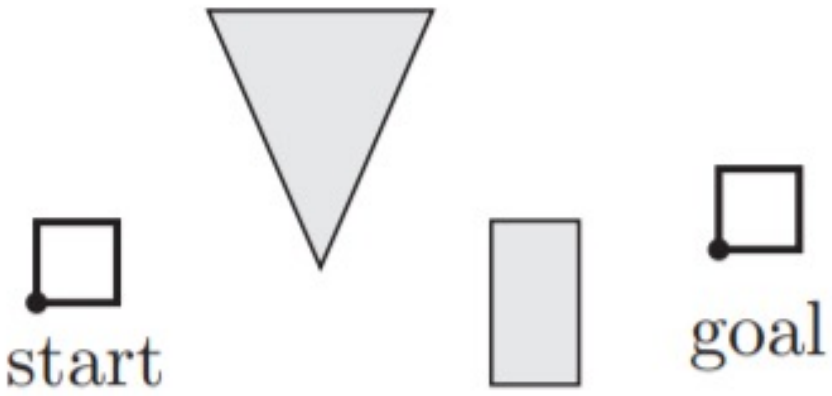


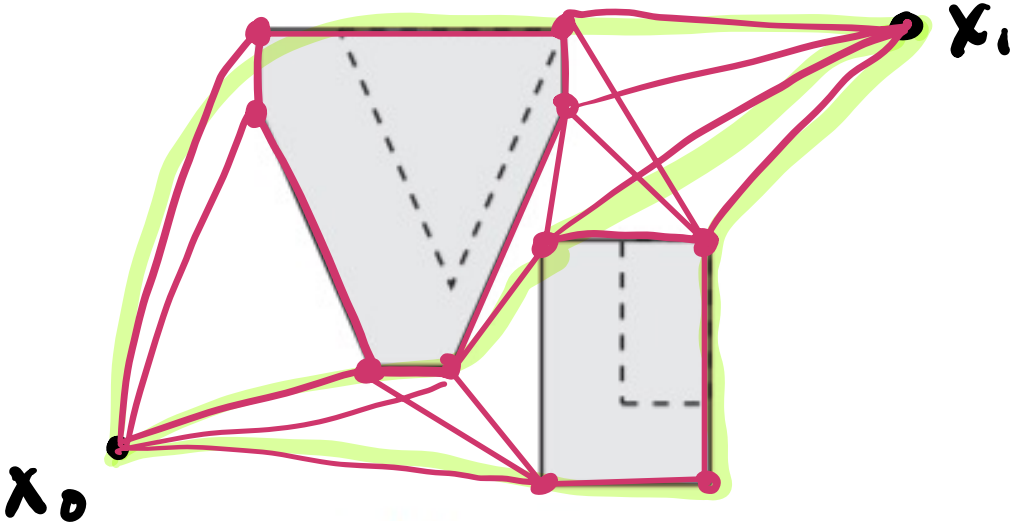
Dijkstra's algorithm.



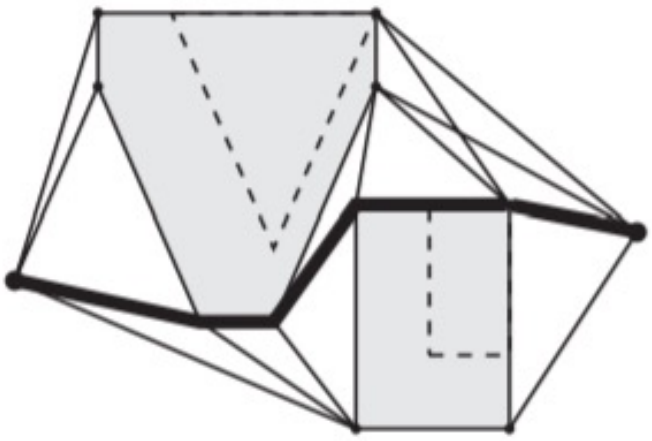
Credit: Subh83 on Wikipedia

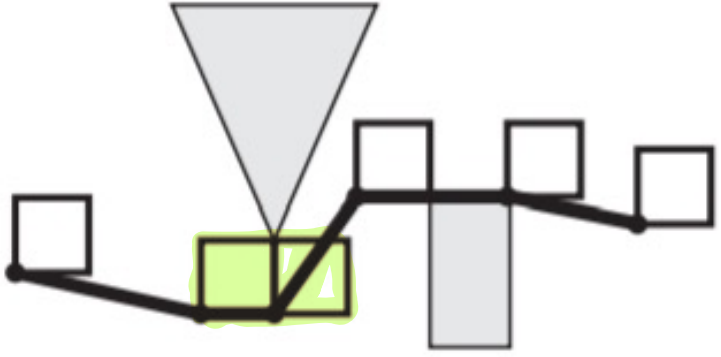
A simple roadmap: visibility graph





A simple roadmap: visibility graph





Roadmap

- We can represent a high-dimensional $\mathcal{C}_{\text{free}}$ by a one-dimentional **roadmap** R with the following properties
  - Reachability: From every point $q \in \mathcal{C}_{\text{free}}$, a free path to a point $q' \in R$ can be found trivially (e.g., a straight-line path)
  - Connectivity: For each connected component of $\mathcal{C}_{\text{free}}$, there is one connected component of R.
- A roadmap is an undirected weighted graph

Question: How to find the points in a roadmap in the general case?

Roadmap

- We can represent a high-dimensional $\mathcal{C}_{\text{free}}$ by a one-dimentional **roadmap** R with the following properties
  - Reachability: From every point $q \in \mathcal{C}_{\text{free}}$, a free path to a point $q' \in R$ can be found trivially (e.g., a straight-line path)
  - Connectivity: For each connected component of $\mathcal{C}_{\text{free}}$, there is one connected component of R.
- A roadmap is an undirected weighted graph

Question: How to find the points in a roadmap in the general case?

Sampling-based method: throw a bunch of random points (samples) on the space, and hope they will make the "key points"

Sampling methods

- A random or deterministic function to choose a sample from the C-space or state space
- A function to evaluate whether a sample is in $\mathcal{C}_{\text{free}}$
- A simple local planner to try to connect to the new sample
- These functions are used to build up a graph or tree representing feasible motions of the robot

Sampling methods

- Compared to grid search methods, sampling methods can quickly find *satisficing* solutions in high-dimensional spaces
  - Fewer nodes will be used
  - Node number of grid points increases exponentially with the dimension of the space
- Most sampling methods are probabilistically complete
  - The probability of finding a solution, when one exists, approaches 100% as the number of samples goes to infinity

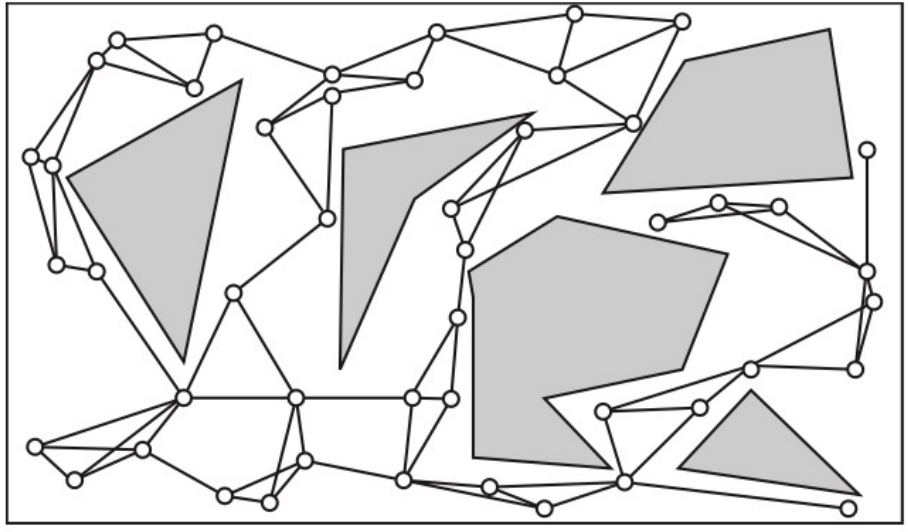
Sampling Based Planners: Probabilistic Roadmaps (PRMs) Kavraki and Latombe, 1994

- **Idea:** build (offline) a graph (i.e., the roadmap) representing the "connectivity" of the environment
- Offline phase:
  - Sample N points from C_{free}
  - Try to connect these points using a fast "local planner"
  - If connection is successful, add an edge between the points.
- At run time:
  - Adding the start node and end node to the graph, by connecting them to the closest nodes in the roadmap
  - Find a path on the roadmap, typically A* search

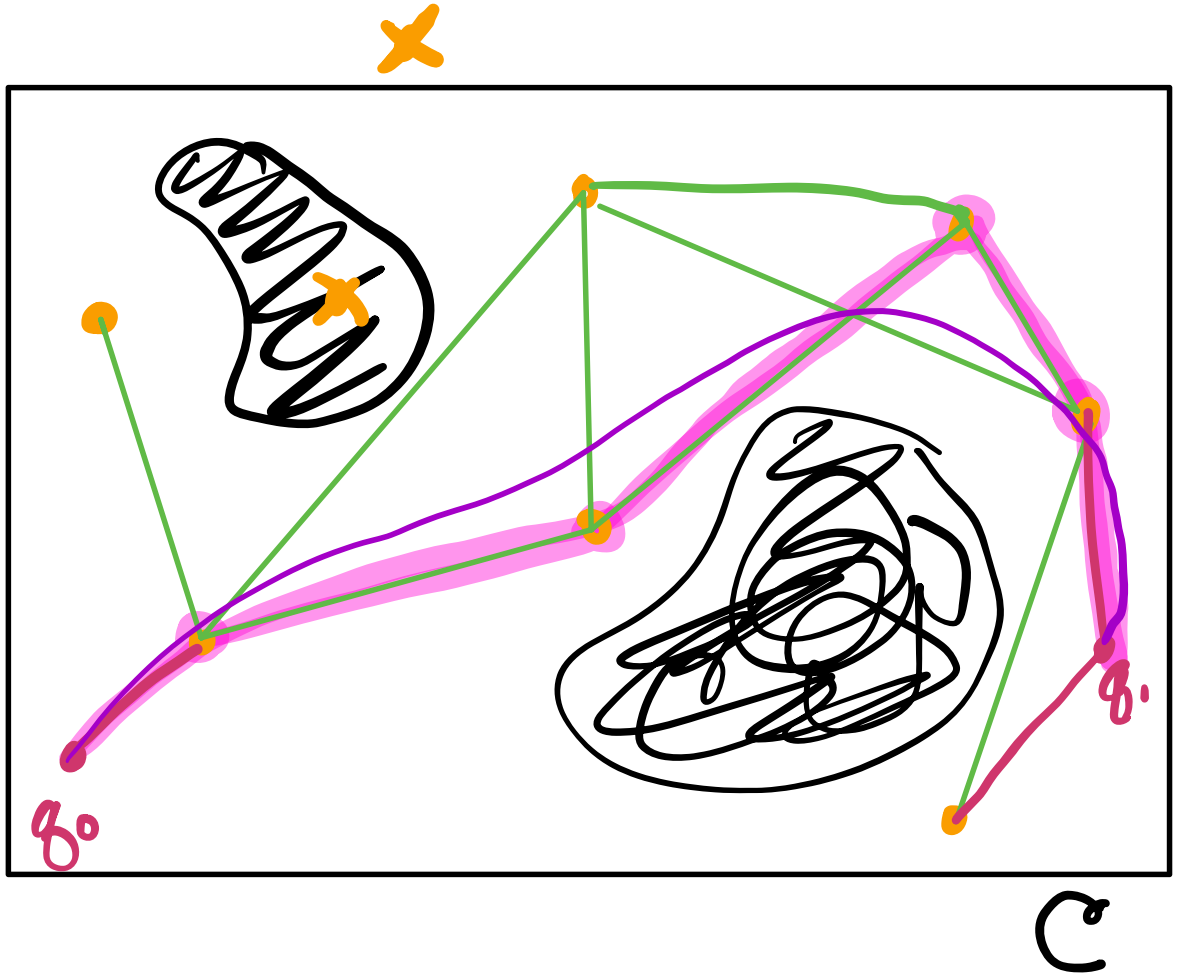
Sampling Based Planners: Probabilistic Roadmaps (PRMs) Kavraki and Latombe, 1994

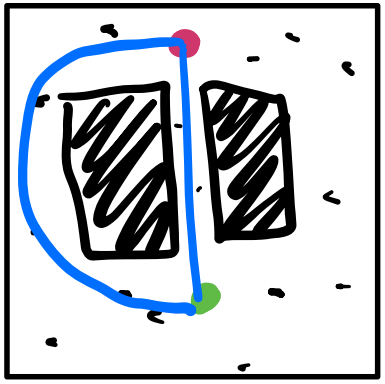
- **Idea:** build (offline) a graph (i.e., the roadmap) representing the "connectivity" of the environment

First planner ever to demonstrate the ability to solve general planning problems in > 4-5 dimensions!

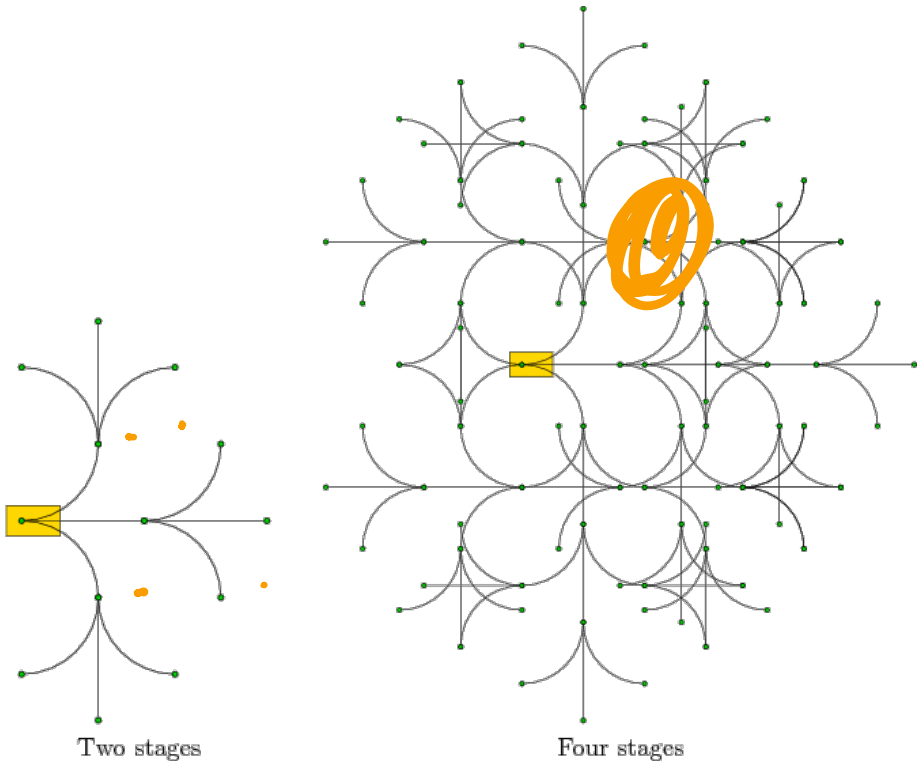


Sampling Based Planners: Probabilistic Roadmaps





Reachability Tree for Dubin's Car



Credit: Steven LaValle, Planning Algorithms

Rapidly Exploring Random Trees (RRT)

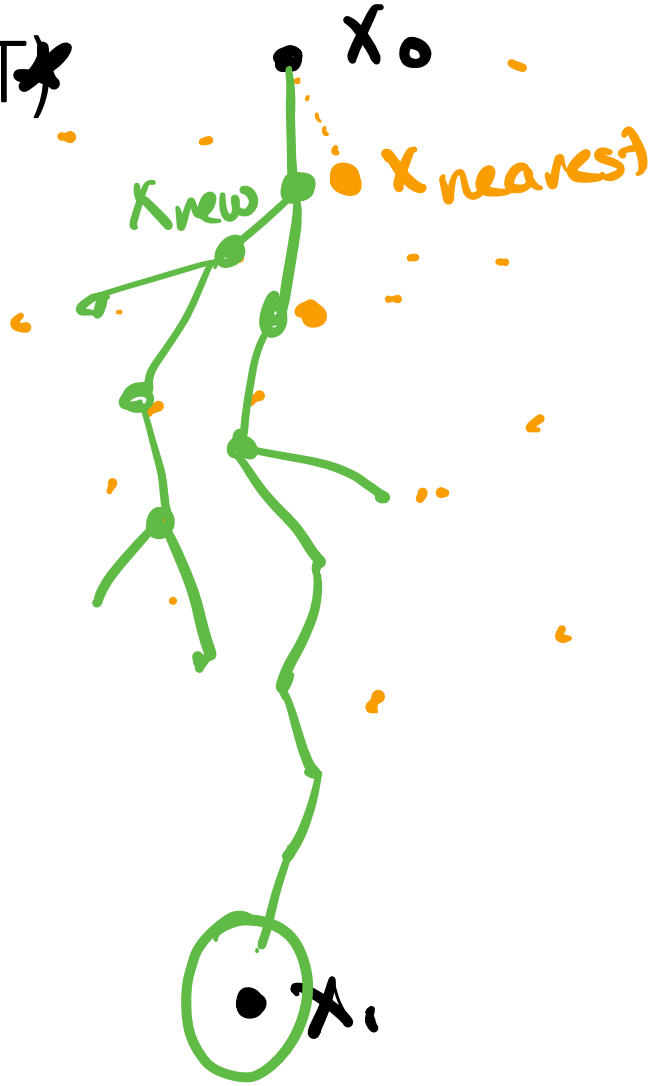
Algorithm 10.3 RRT algorithm.

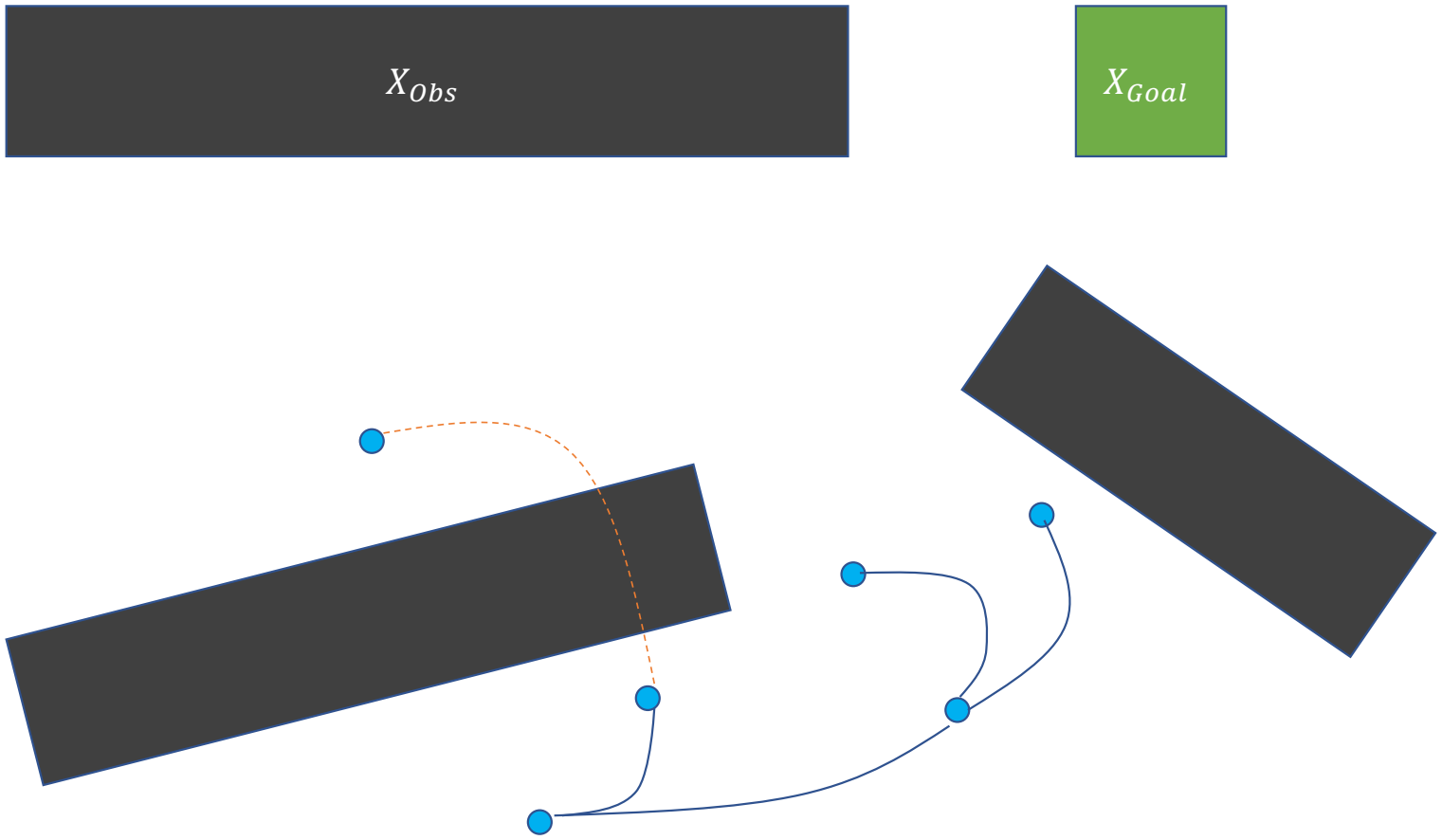
initialize search tree T with x_{start}
while T is less than the maximum tree size **do**
  $x_{\text{samp}} \leftarrow$ sample from $\mathcal{X}$
  $x_{\text{nearest}} \leftarrow$ nearest node in T to x_{samp}
  employ a local planner to find a motion from x_{nearest} to x_{new} in the direction of x_{samp}
  if the motion is collision-free **then**
    add x_{new} to T with an edge from x_{nearest} to x_{new}
    if x_{new} is in $\mathcal{X}_{\text{goal}}$ **then**
      return SUCCESS and the motion to x_{new}
    end if
  end if
end while
return FAILURE

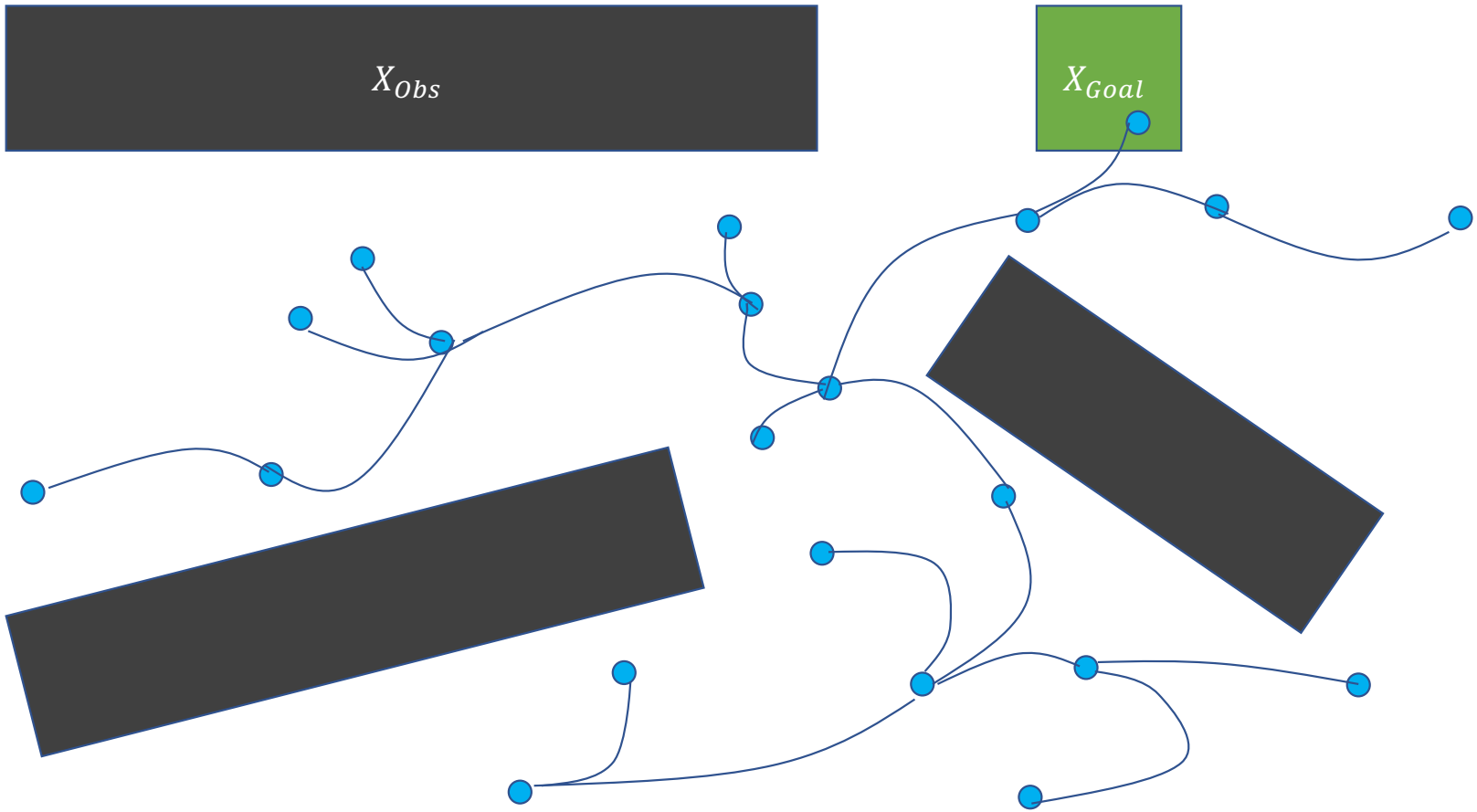
Rapidly Exploring Random Trees (RRT)

Algorithm 10.3 RRT algorithm.

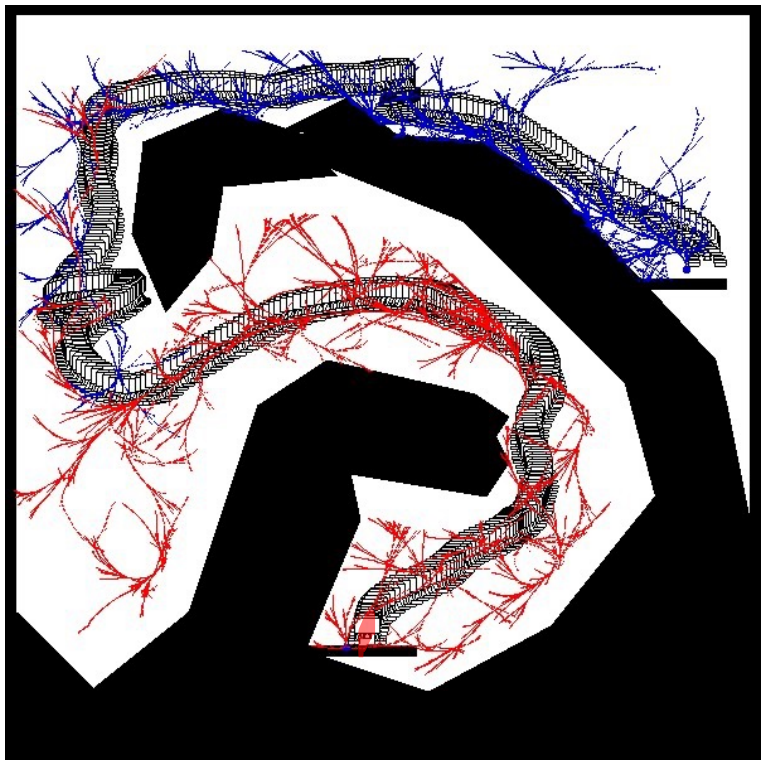
1: initialize search tree T with x_{start}
2: **while** T is less than the maximum tree size **do**
3: $x_{\text{samp}} \leftarrow$ sample from $\mathcal{X}$
4: $x_{\text{nearest}} \leftarrow$ nearest node in T to x_{samp}
5: employ a local planner to find a motion from x_{nearest} to x_{new} in the direction of x_{samp}
6: **if** the motion is collision-free **then**
7: add x_{new} to T with an edge from x_{nearest} to x_{new}
8: **if** x_{new} is in $\mathcal{X}_{\text{goal}}$ **then**
9: **return** SUCCESS and the motion to x_{new}
10: **end if**
11: **end if**
12: **end while**
13: **return** FAILURE

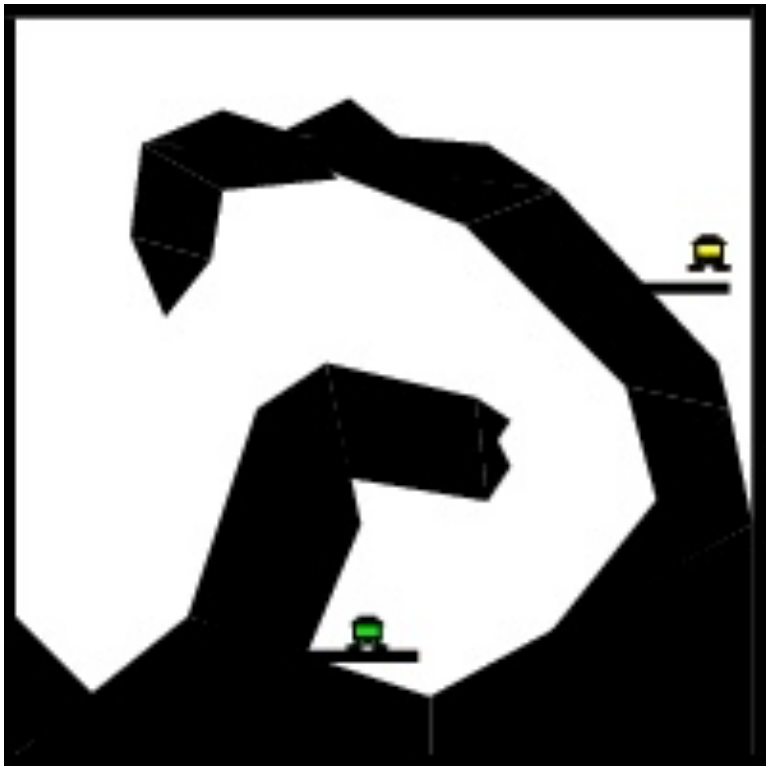



X_{Obs}
X_{Goal}


X_{Obs}
X_{Goal}

RRT: Lunar Lander





Check out Steven Lavalle's RRT Gallery: http://msl.cs.uiuc.edu/rrt/gallery.html

A few things that might be useful to know

- What are some key properties of planners?
- Think about what applications some properties or types of planners might be needed.
- If given a very simple graph, can you find the shortest path?
- Be somewhat familiar with the pros and cons of the planners we just discussed.

Summary

- Given an initial state and a desired final state, **motion planning** provides us with tools to find a time horizon and a sequence of actions to find a trajectory that reaches the goal without collisions
- A **roadmap** path planner uses a graph representation of free space, which can then provide a trajectory using search algorithms
- Introduced the **visibility graph**, which is a simple complete planner
- The basic **RRT** algorithm is a sampling-based method that grows a single search tree from start to find a motion to goal
  - Uses a local planner to find a motion from the nearest node to the sampled node