Autonomous Robotics

Winter 2025

Abhishek Gupta

TAs: Carolina Higuera, Entong Su, Bernie Zhu

Class Outline

State Estimation

- Robotic System Design
- Filtering
- Localization
- SLAM

Control

- Feedback Control
- PID Control
- MPC
- LQR

Planning

- Search
- Heuristic Search
- Motion Planning
- Lazy Search

Learning

- Imitation Learning
- Policy Gradient
- Actor-Critic
- Model-Based RL

Logistics

- HW3 due Feb 20, extensions for hardware issues

- Post questions, discuss any issues you are having on Ed.
- Students with **no** access to 002, e-mail us with your student ID.
- Students that have not been added to the class, email abhgupta@cs.washington.edu with the subject-line "Waitlisted for CSE478"

Lecture Outline

Overview of the impact and scope of planning

↓

Formalizing the problem

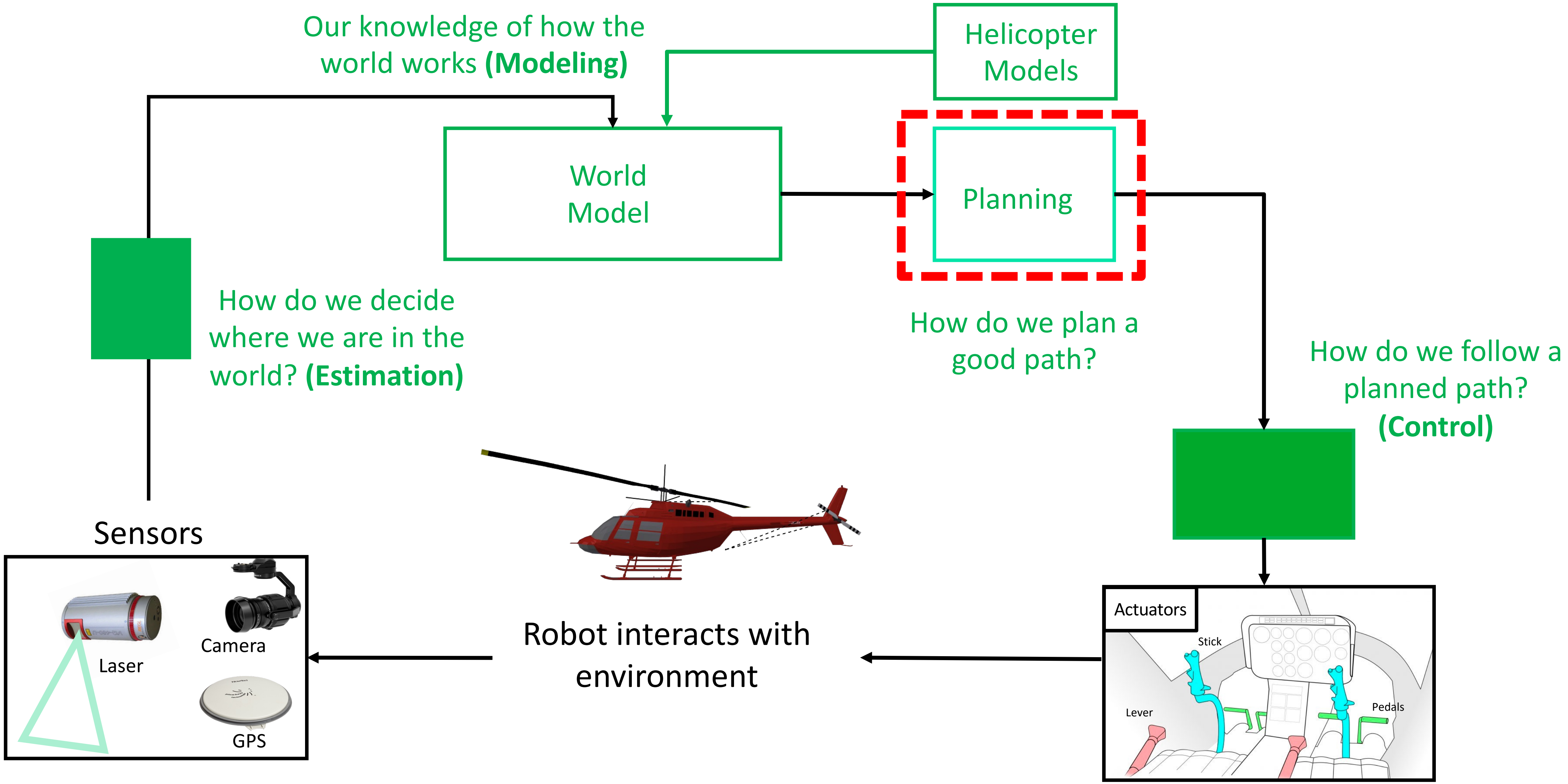
↓

Why is the problem hard?

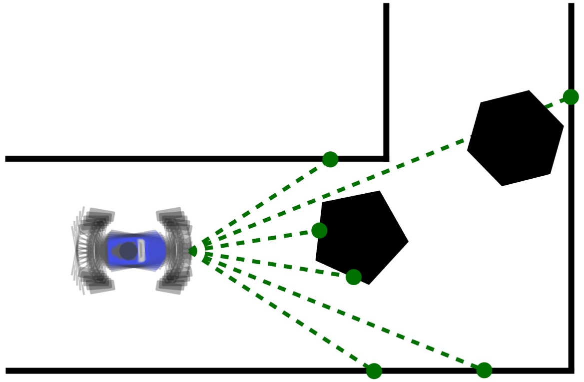
↓

A start at approaching the problem

Let’s zoom back out

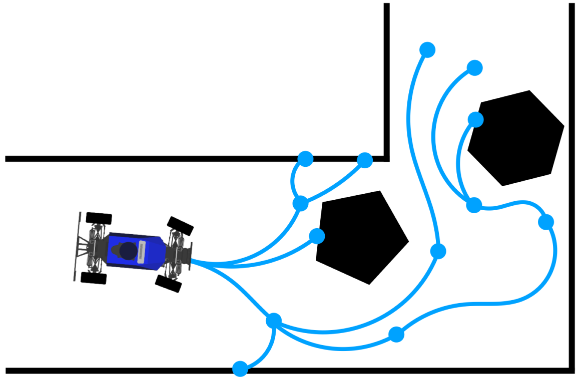


The Sense-Plan-Act Paradigm



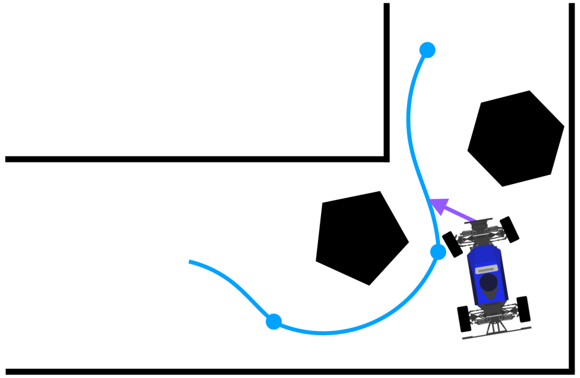
Estimate
robot state

Solved over first 3 weeks



Plan sequence of
motions

??



Control robot to
follow plan

Solved over last 2 weeks

Flying from Seattle to LA?

High-level sequence of actions:

get to the terminal, board an airplane, etc.

If we have a detailed plan to get to the terminal (and some idea of how to check in and board), should we also plan our route through LAX?

Rental car? Lodging? What future problems should we solve now?

How do I get out of my house?

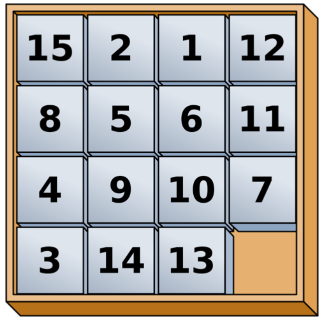
(EXAMPLE FROM LESLIE KAELBLING)

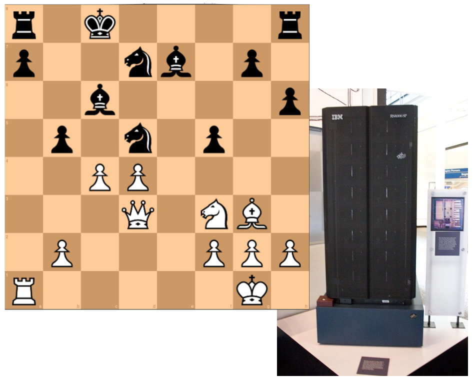
What Makes (Motion) Planning Difficult?

Classic AI planning problems: Rubik's cube, sliding-tile puzzle, chess

- Discrete state space, strictly-defined rules, humans have great intuition
- Developed many of the tools that are still used today!

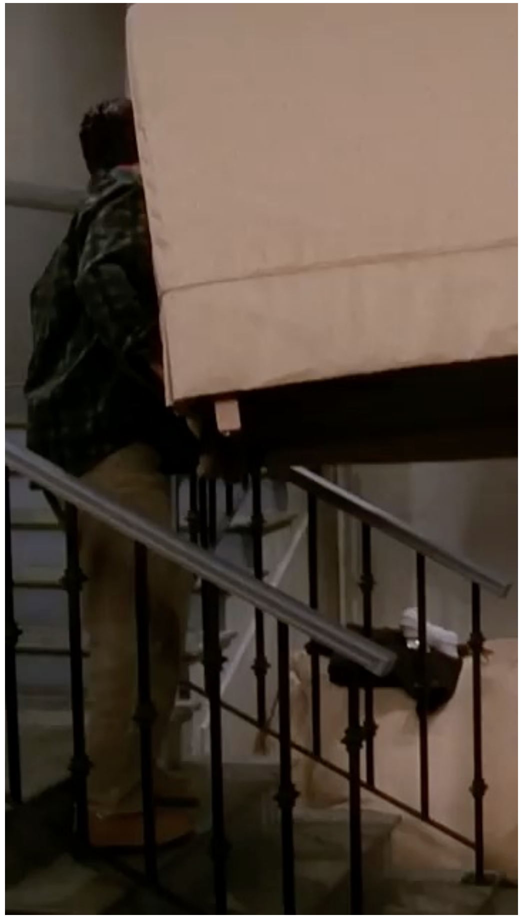
(Some) challenges in motion planning: continuous state space, expensive action simulation, robot model uncertainty, nonholonomic constraints





(WIKIPEDIA)

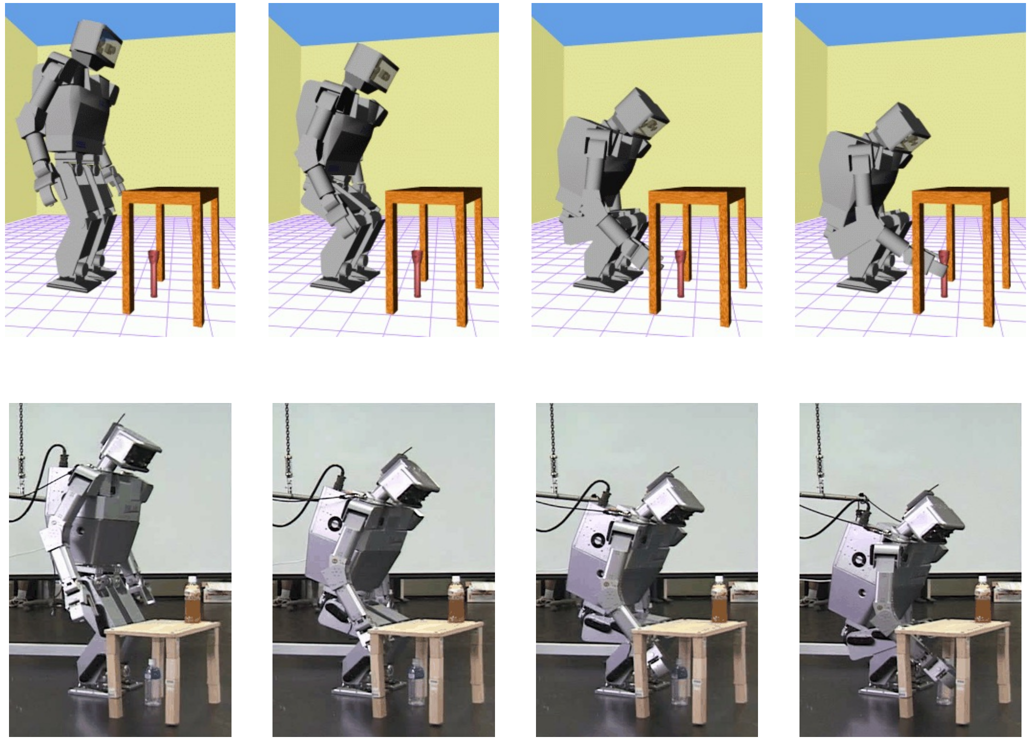
The Piano Mover's Problem



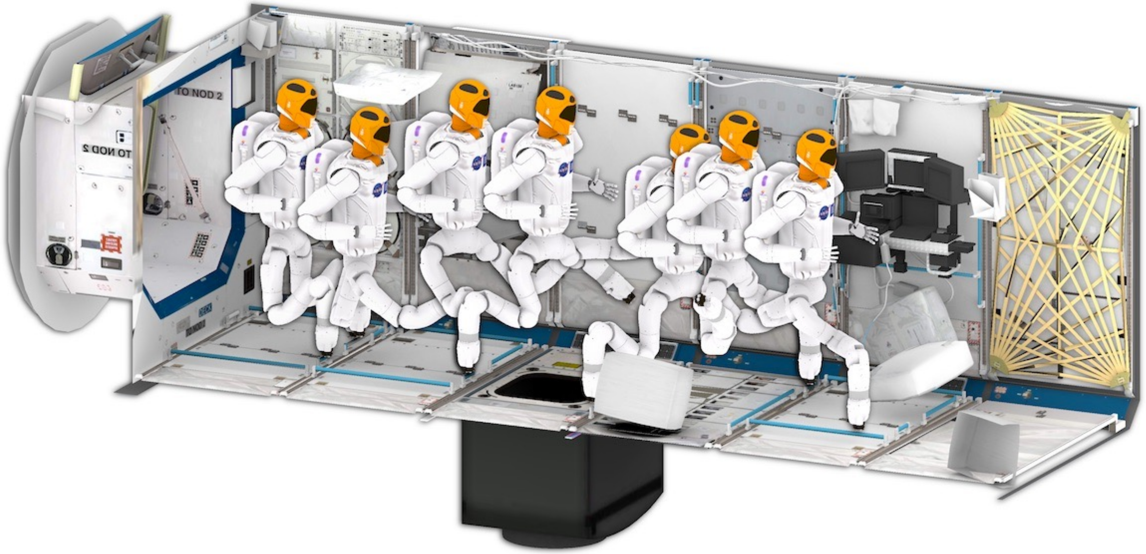
SCHWARTZ AND SHARIR, 1983

HTTPS://YOUTU.BE/UBAGTSNZABK

High-Dimensional Planning



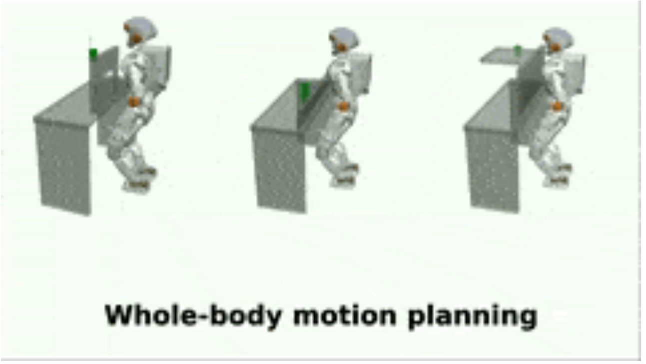
HONDA H7 HUMANOID ROBOT
KUFFNER ET AL., 2003

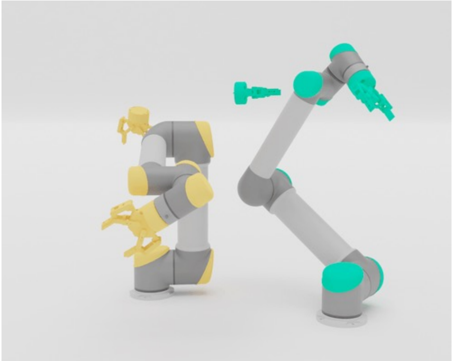


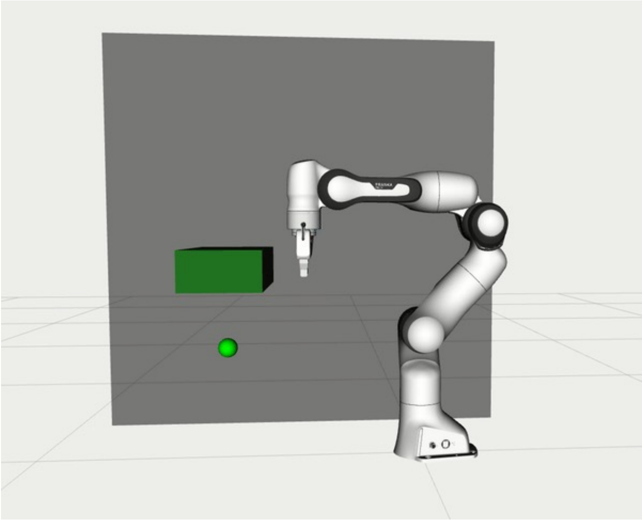
NASA R2 HUMANOID ROBOT
KINGSTON ET AL., 2019

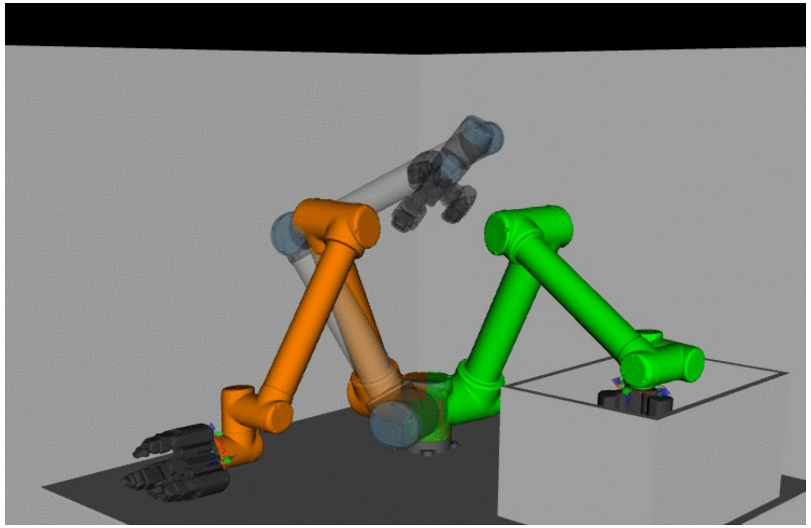
Motion/Path Planning

Examples (of what is usually referred to as motion planning):





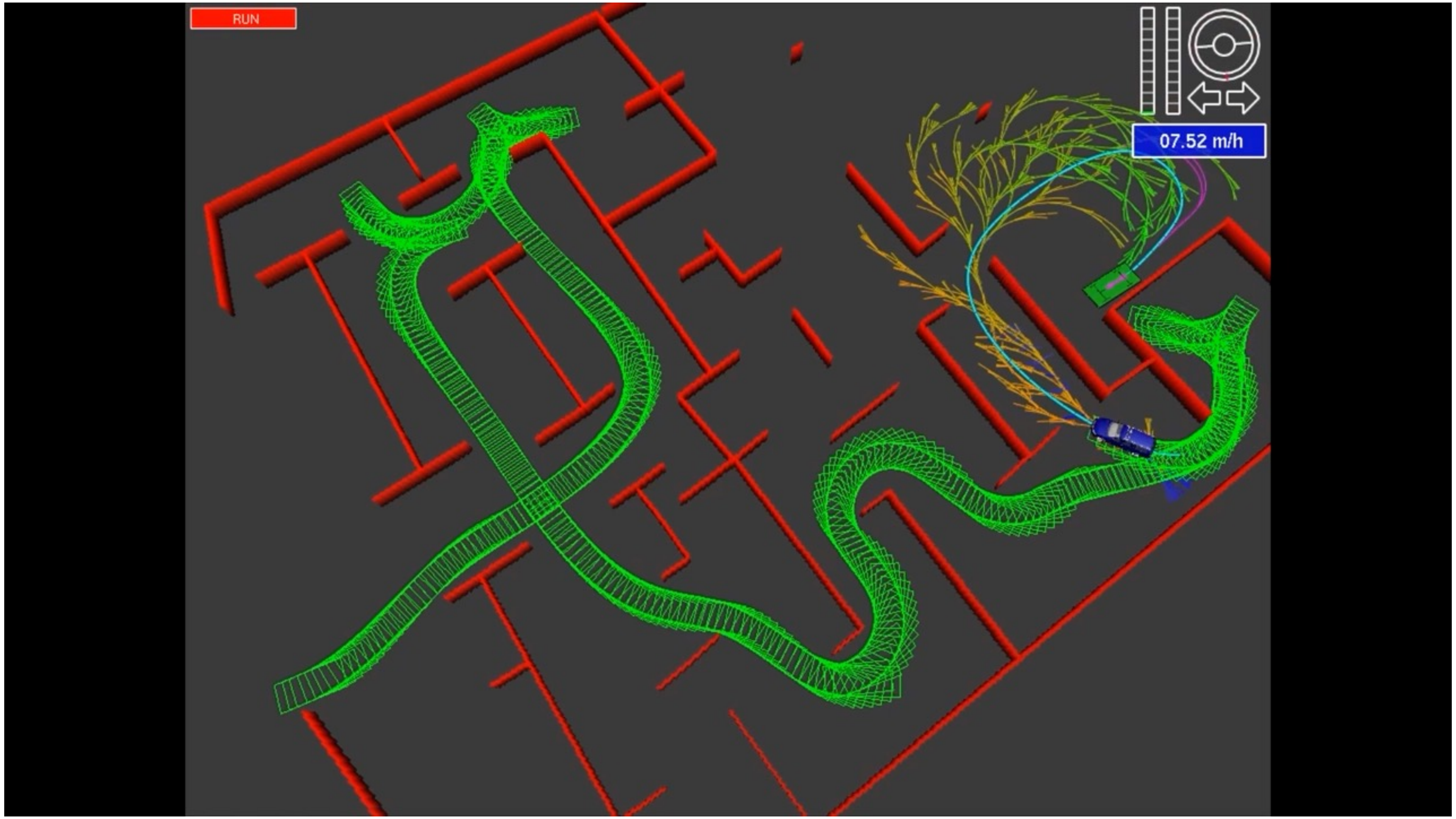




Real-time planning

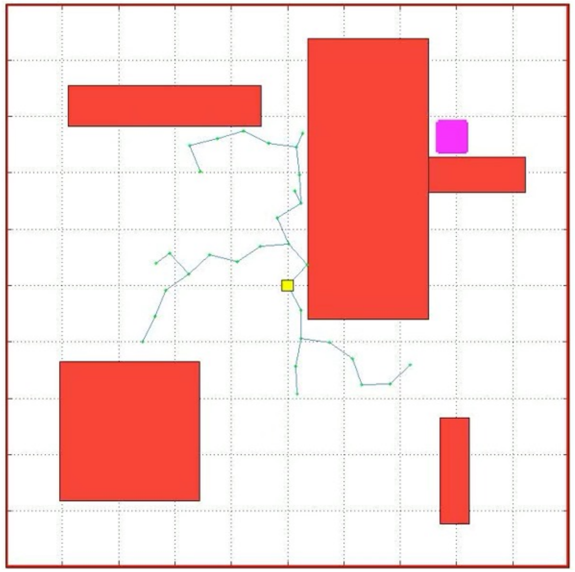
Willow garage, 2009

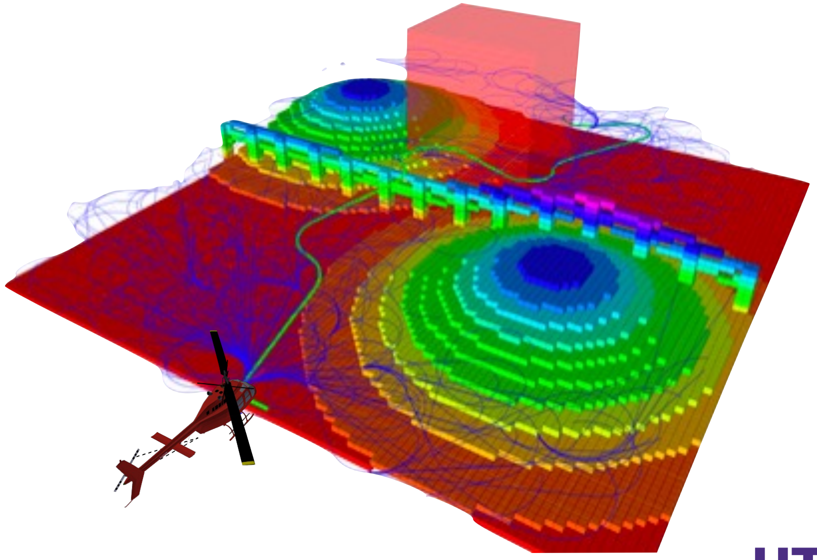
https://www.youtube.com/watch?v=qbQDJ1c_Nxk&feature=youtu.be

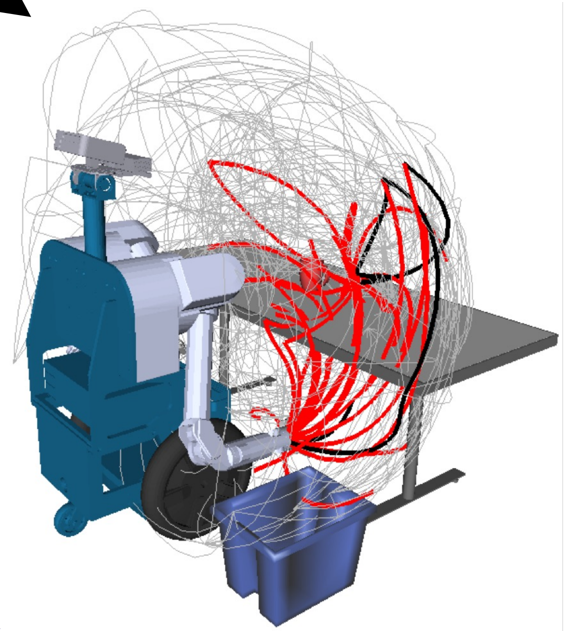


HTTPS://YOUTU.BE/QXZT-B7IUYW

One algorithm to rule them all







HTTPS://YOUTU.BE/FAFW8DOKVIK

Ok so let’s motion plan

1. Specify and formulate the problem

2. Understand the problem difficulty

3. Think about formalizations that let us tackle the problem

Lecture Outline

Overview of the impact and scope of planning

↓

Formalizing the problem

↓

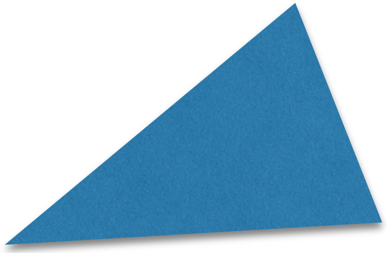
Why is the problem hard?

↓

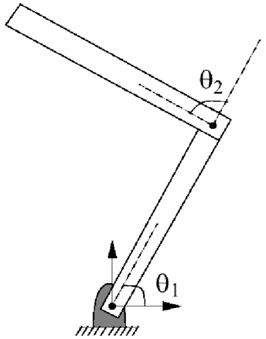
A start at approaching the problem

Specifying the problem

Representations that Generalize



(a) Translating Triangle



(b) 2-joint planar arm

(c) Racecar

(d) Manipulator

The Configuration Space

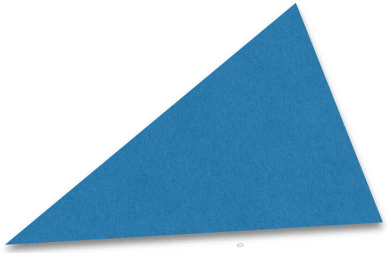
The configuration space or C-space is the manifold that contains the set of transformations achievable by the robot.

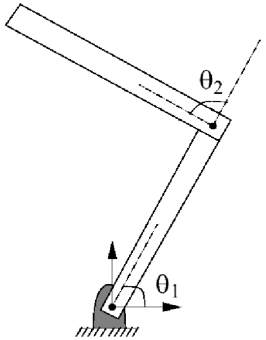
$$\mathcal{C}$$

Complete specification of the location of every point on robot geometry

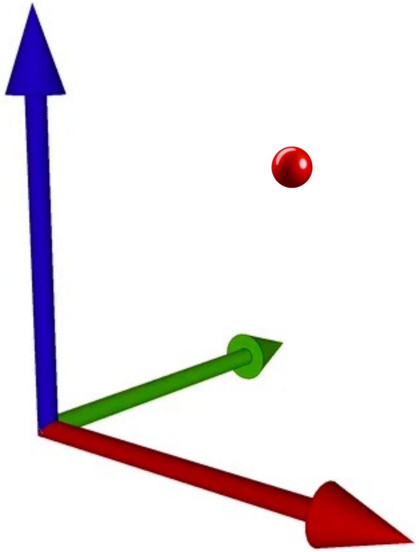
Key Insight

Represent the robot as a point in some high-dimensional space

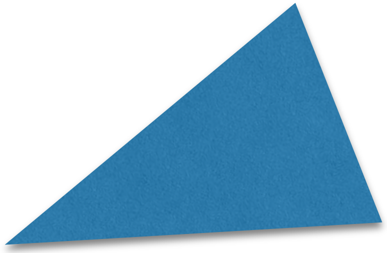




$$\mathcal{C}$$



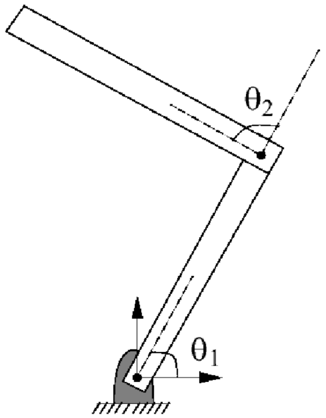
Example 1: Translating triangle

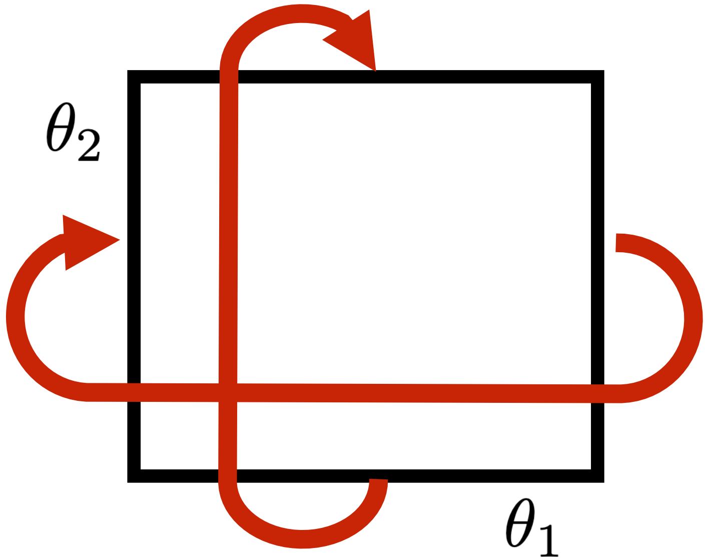


$$\mathbb{R} \times \mathbb{R} = \mathbb{R}^2$$

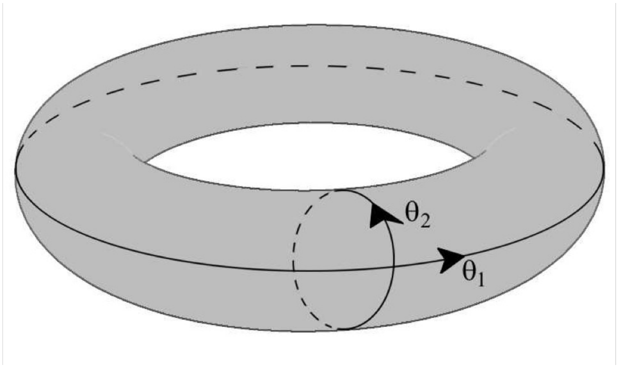
(cartesian product)

Example 2: 2-joint planar arm





$$\mathbb{S}^1 \times \mathbb{S}^1 = \mathbb{T}^2$$



Circle

$$\mathbb{S}^1 = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\}.$$

Example 3: Racecar

$$\mathbb{R}^2 \times \mathbb{S}^1$$

special Euclidean group $SE(2)$

Common C-spaces

Type of Robot	**$\mathcal{C}$-space Representation**
Mobile robot translating in the plane	$\mathbb{R}^2$
Mobile robot translating and rotating in the plane	$SE(2)$ or $\mathbb{R}^2 \times S^1$
Rigid body translating in the three-space	$\mathbb{R}^3$
A spacecraft	$SE(3)$ or $\mathbb{R}^3 \times SO(3)$
An n-joint revolute arm	T^n
A planar mobile robot with an attached n-joint arm	$SE(2) \times T^n$

(Kavraki and LaValle)

Obstacles

Obstacle specification

Robot operates in a 2D / 3D workspace $\mathcal{W} = \mathbb{R}^2 \text{ or } \mathbb{R}^3$

Subset of this space is obstacles

$$\mathcal{O} \subset \mathcal{W}$$

semi-algebraic models (polygons, polyhedra)

Geometric shape of the robot
(set of points occupied by robot at a config)

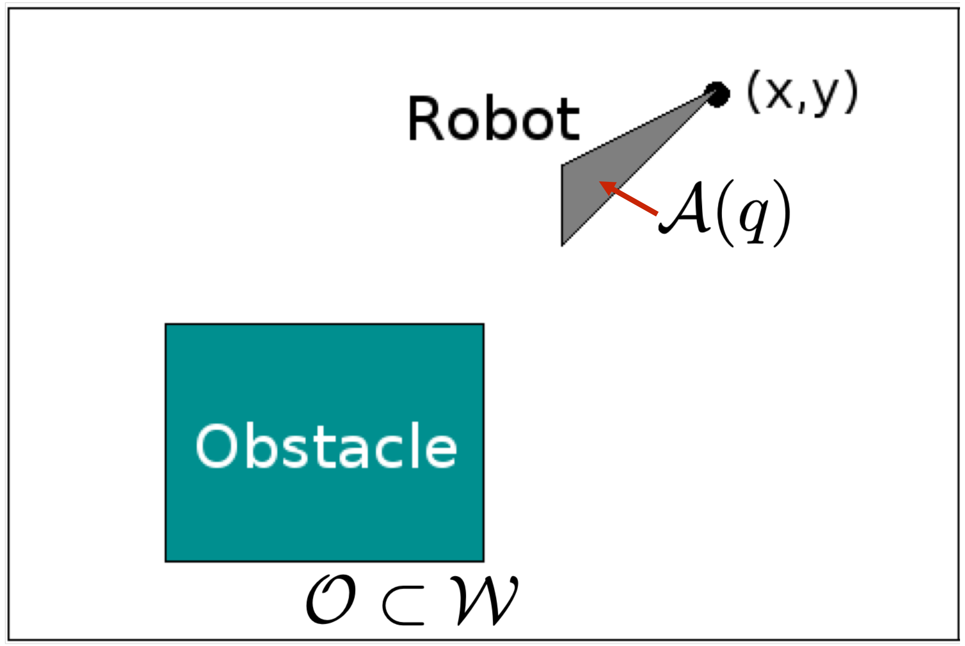
$$\mathcal{A}(q) \subset \mathcal{W}$$

C-space obstacle region

$$\mathcal{C}_{obs} = \{\boldsymbol{q} \in \mathcal{C} \mid \mathcal{A}(\boldsymbol{q}) \cap \mathcal{O} \neq \emptyset\}$$

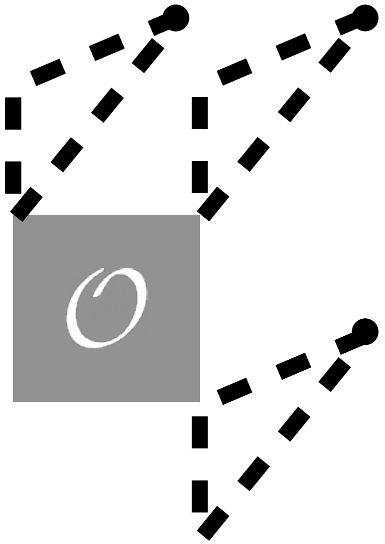
$$\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs}$$

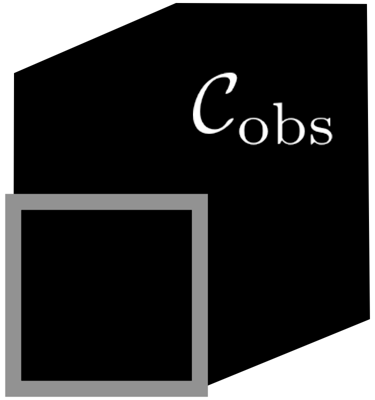
Example 1: Translating triangle



Can be efficiently computed using Minkowski sum

Example: Translating Triangle in Plane

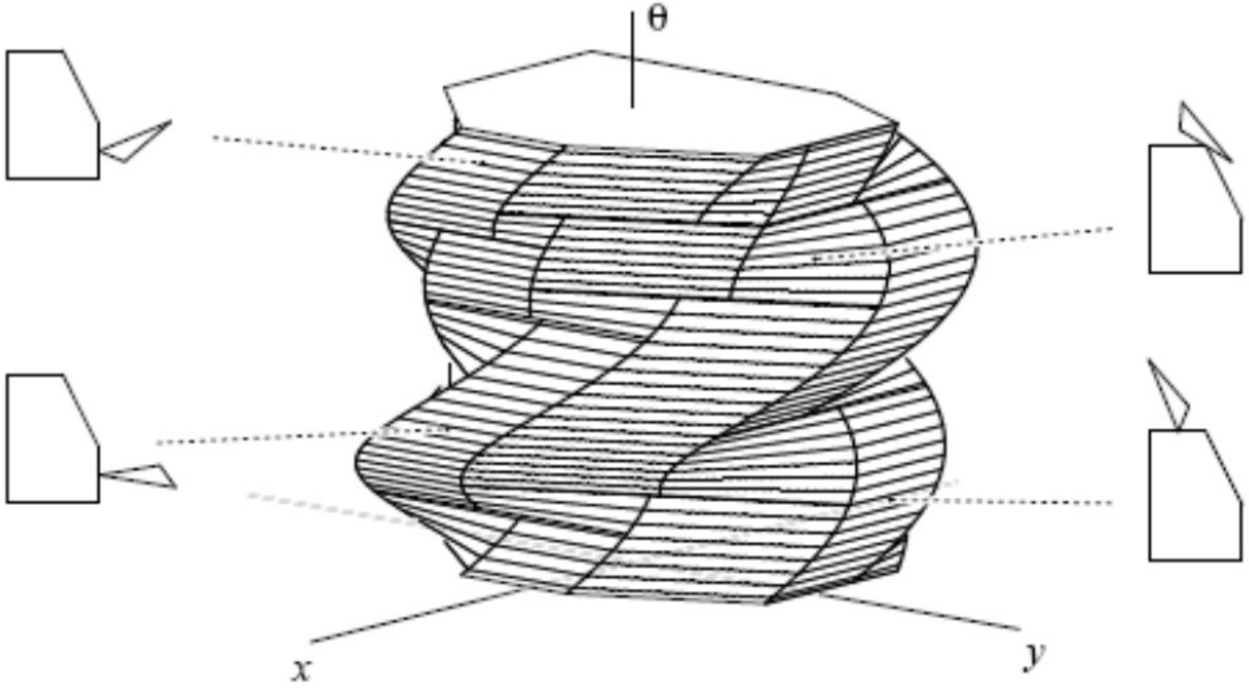




Can be computed for convex polygons (Minkowski sum)

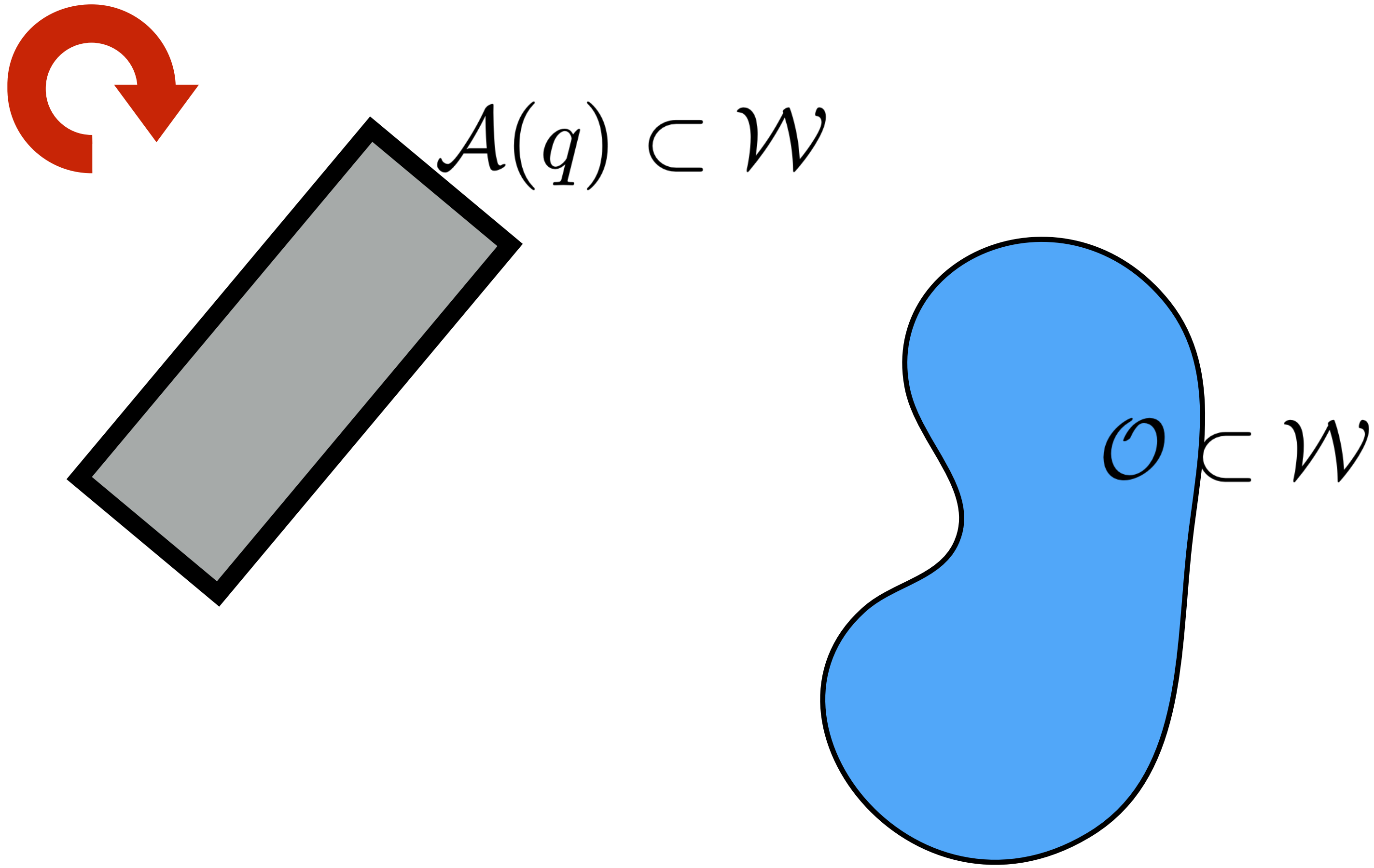
(EXAMPLE FROM LYDIA KAVRAKI AND STEVEN LAVALLE)

Example: Translating and Rotating Triangle

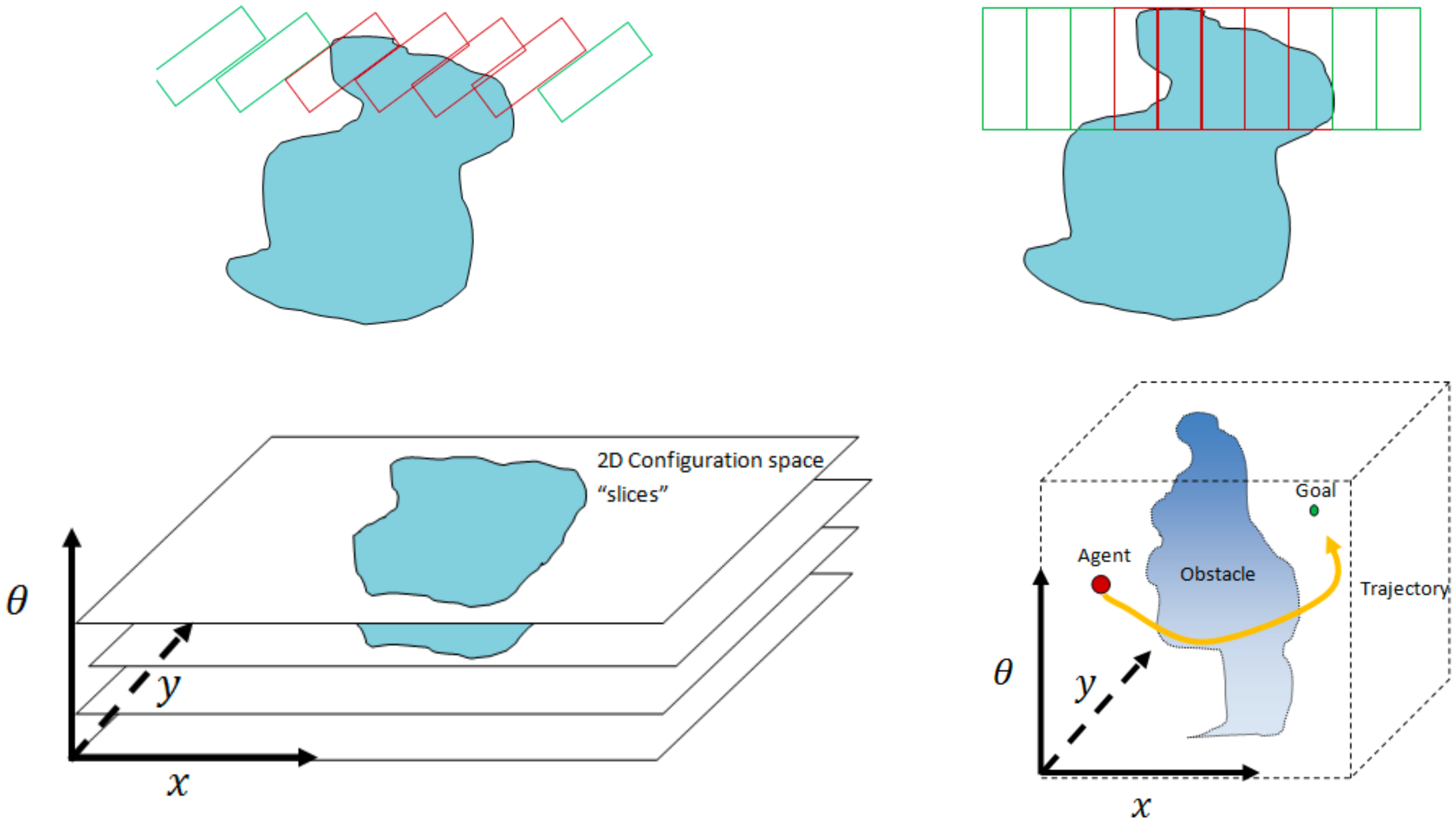


(EXAMPLE FROM HOWIE CHOSET)

Example 2: SE(2) robot



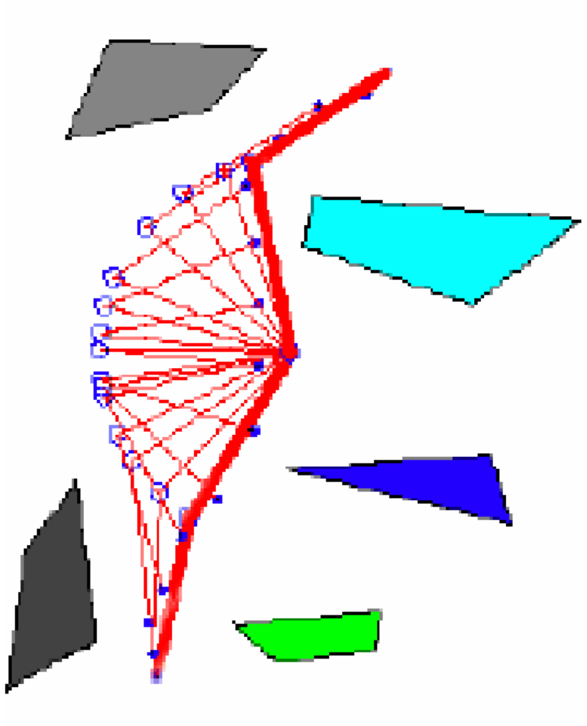
Example 2: SE(2) robot

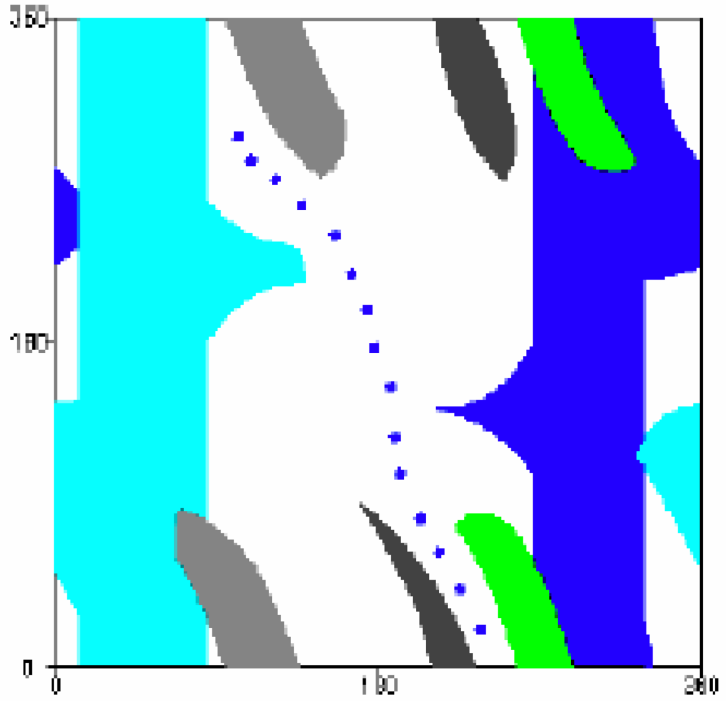


(Courtesy Matt Klingensmith)

https://www.gamasutra.com/blogs/MattKlingensmith/20130907/199787/Overview_of_Motion_Planning.php

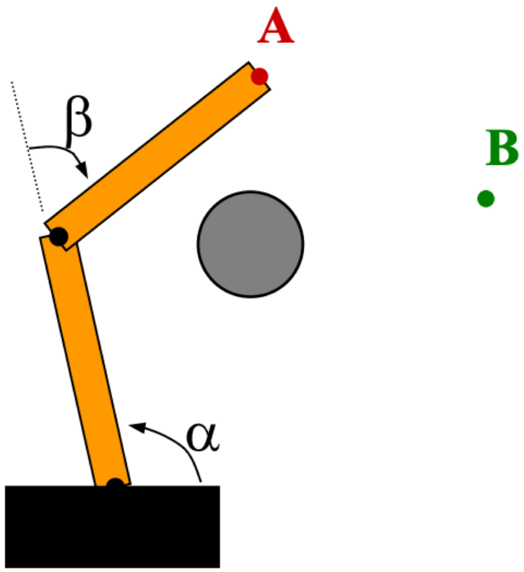
Example 3: 2-link planar arm

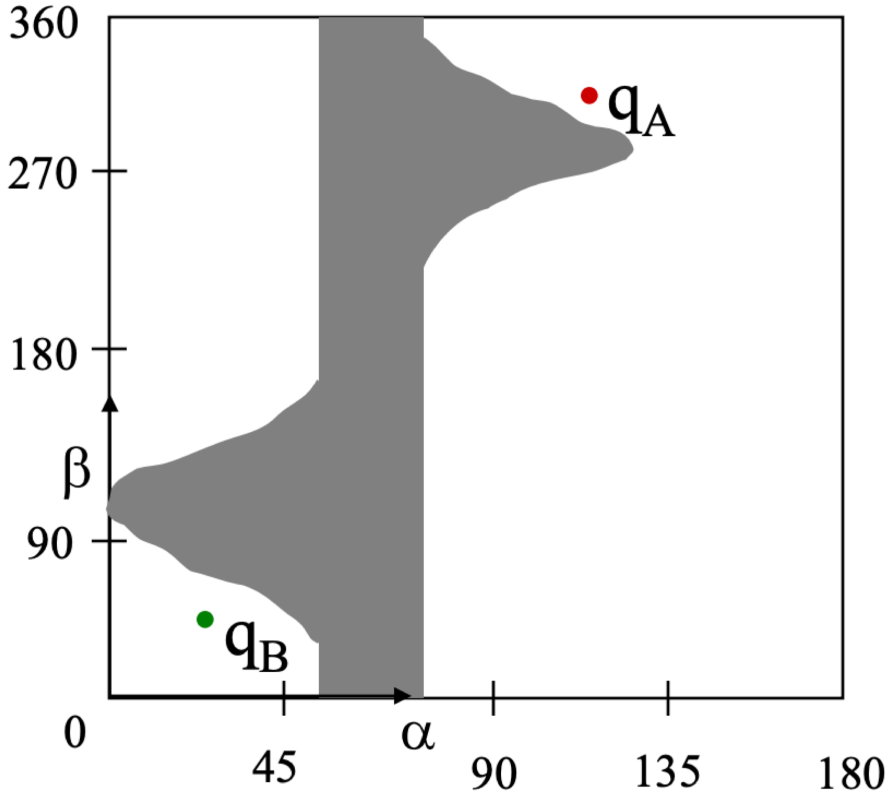




Courtesy Tapomayukh Bhattacharya

Example 3: 2-link planar arm

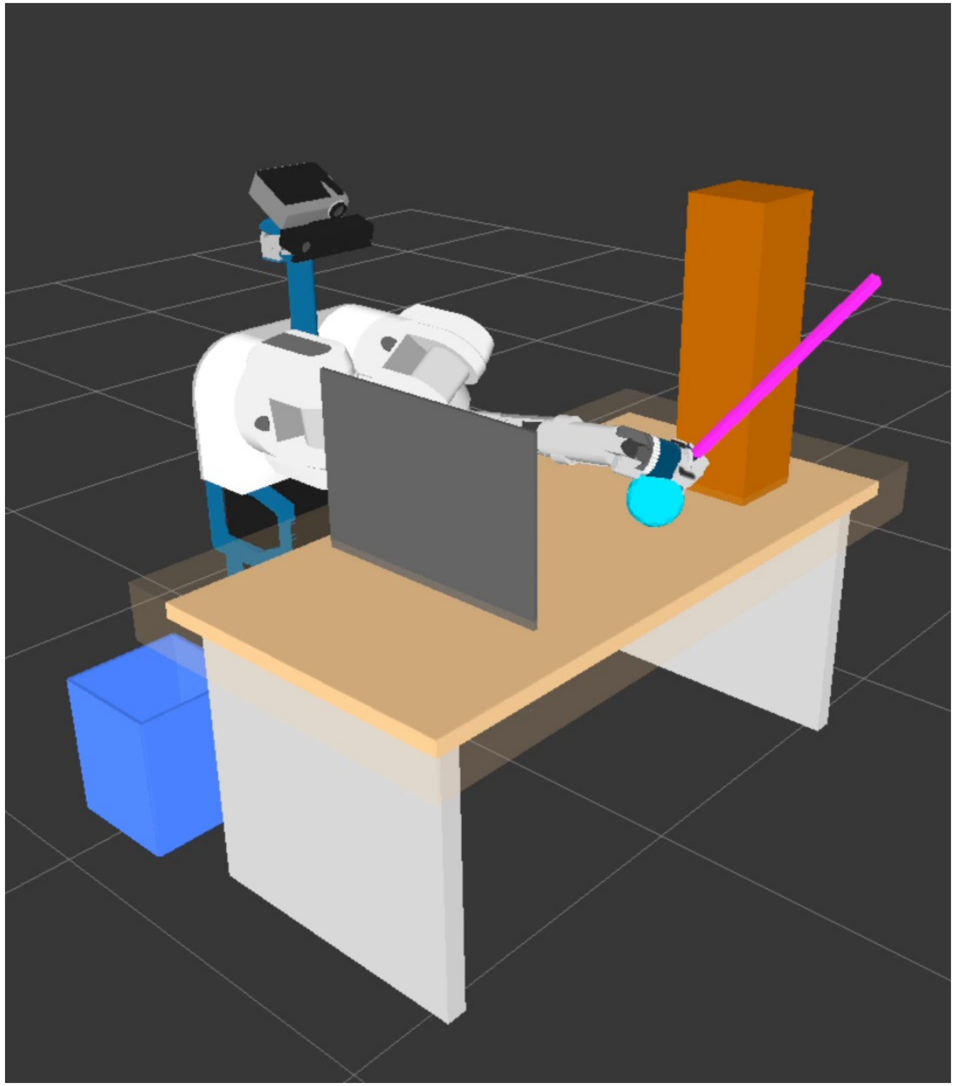




Courtesy Tapomayukh Bhattacharya

Geometric Path Planning Problem

Geometric Path Planning Problem



Also known as
Piano Mover's Problem (Reif 79)

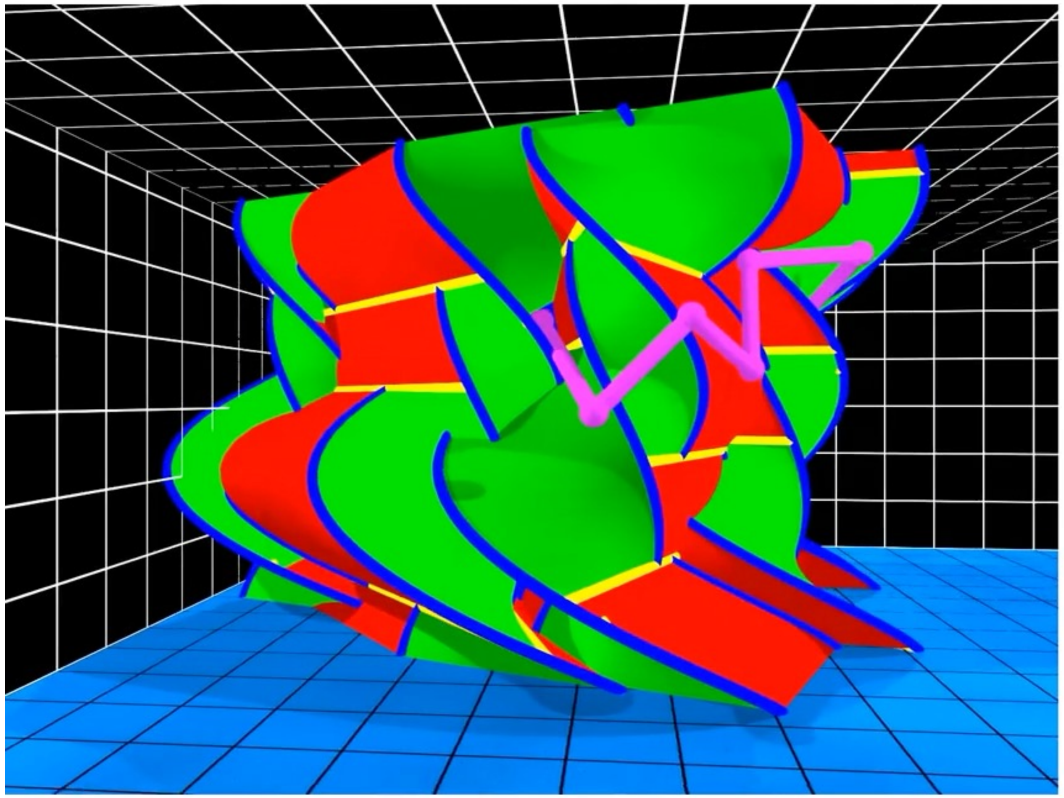
Given:

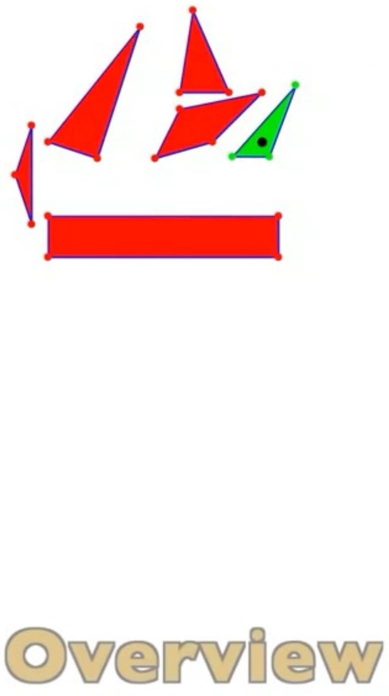
1. A *workspace* $\mathcal{W}$, where either $\mathcal{W} = \mathbb{R}^2$ or $\mathcal{W} = \mathbb{R}^3$.
2. An *obstacle region* $\mathcal{O} \subset \mathcal{W}$.
3. A *robot* defined in $\mathcal{W}$. Either a rigid body $\mathcal{A}$ or a collection of m links: $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m$.
4. The *configuration space* $\mathcal{C}$ (C_{obs} and C_{free} are then defined).
5. An *initial configuration* $\boldsymbol{q_I} \in \mathcal{C}_{free}$.
6. A *goal configuration* $\boldsymbol{q_G} \in \mathcal{C}_{free}$. The initial and goal configuration are often called a *query* $(\boldsymbol{q_I}, \boldsymbol{q_G})$.

Compute a (continuous) path, $\tau : [0,1] \to \mathcal{C}_{free}$, *such that* $\tau(0) = \boldsymbol{q_I}$ *and* $\tau(1) = \boldsymbol{q_G}$.

Also may want to minimize cost $c(\tau)$

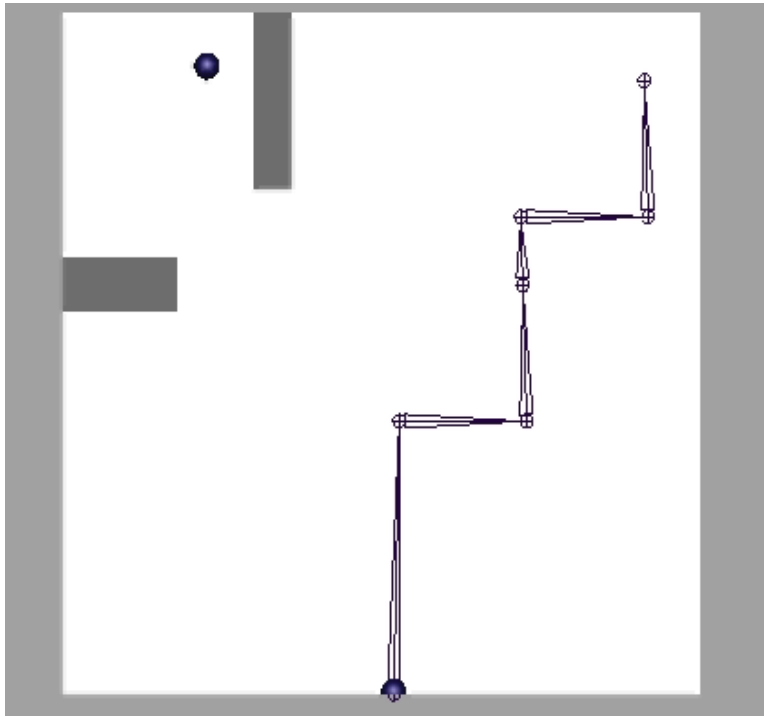
Planning in Configuration Space





Motion/Path Planning

Examples (of what is usually referred to as motion planning):



Planned motion for a 6DOF robot arm

Lecture Outline

Overview of the impact and scope of planning

↓

Formalizing the problem

↓

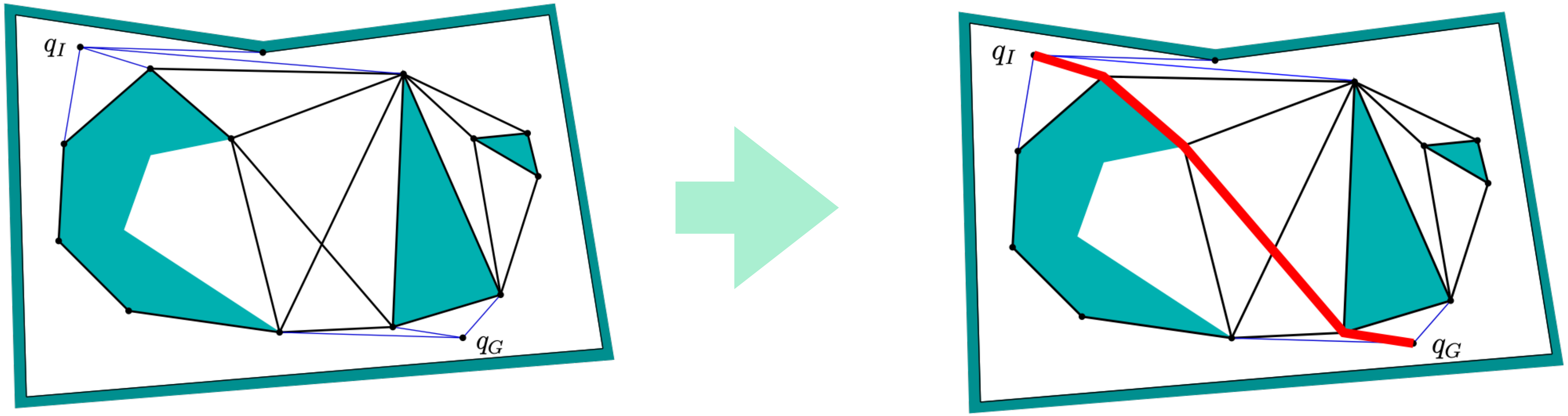
Why is the problem hard?

↓

A start at approaching the problem

Understanding Problem Difficulty

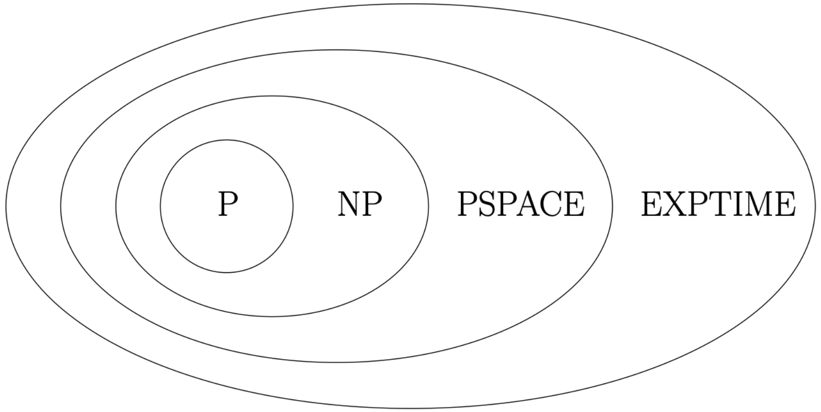
Can we solve this for some problems?

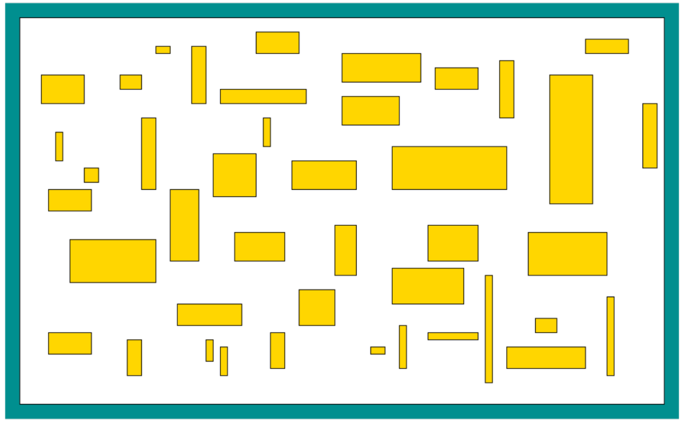


Yes! E.g. 2D polygon robots / obstacles can be solved with visibility graphs

Hardness of general motion planning

Piano Mover's problem is PSPACE-hard (Reif et al.)





Even planning for translating rectangles is PSPACE-hard!

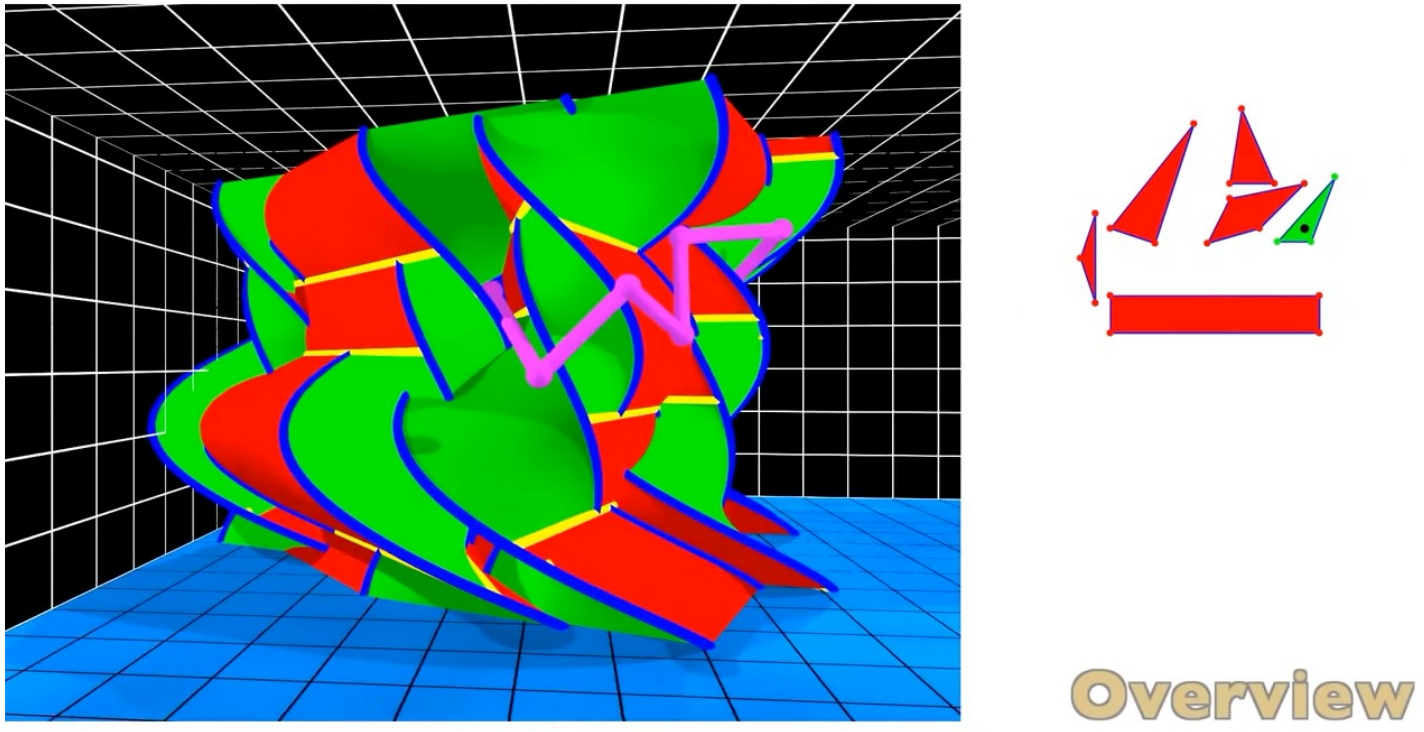
(Hopcroft et al. 84)

Certain 3D robot planning under uncertain is NEXPTIME-hard!

(Canny et al. 87)

Intuition: Why is motion planning non-trivial?

- Searching/Optimization through a complex non-convex space
- Combination of discrete/continuous optimization



Scales poorly with dimensionality of space and number of obstacles

Geometric Path Planning

WORKSPACE, OBSTACLE REGIONS

$\mathcal{W}, \mathcal{O} \subset \mathcal{W}$

CONFIGURATION SPACE

$\mathcal{C}, \mathcal{C}_{\text{obs}}, \mathcal{C}_{\text{free}}$

START + GOAL CONFIGURATIONS

q_s, q_g

PLANNING

$\xi : [0, 1] \rightarrow \mathcal{C}_{\text{free}}$

$\xi(0) = q_s, \xi(1) = q_g$

COLLISION-FREE PATH

(WHICH MAY MINIMIZE COST)

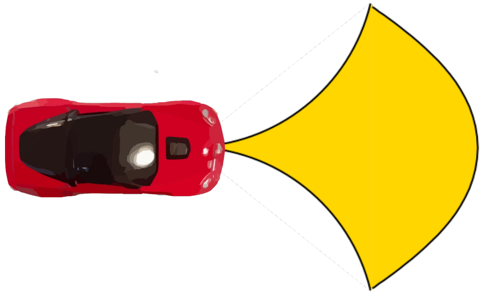
Differential constraints

In geometric path planning, we were only dealing with C-space

$$q \in \mathcal{C}$$

We now introduce differential constraints

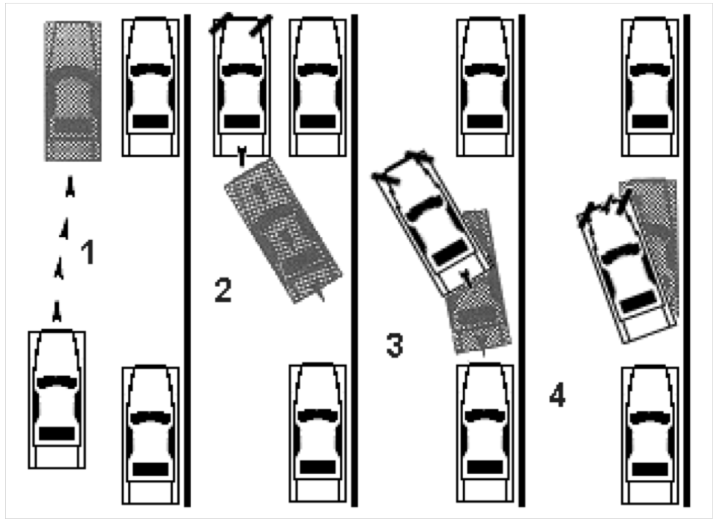
$$\begin{bmatrix} \dot{q} \\ \ddot{q} \end{bmatrix} = f(\begin{bmatrix} q \\ q \end{bmatrix}, u)$$

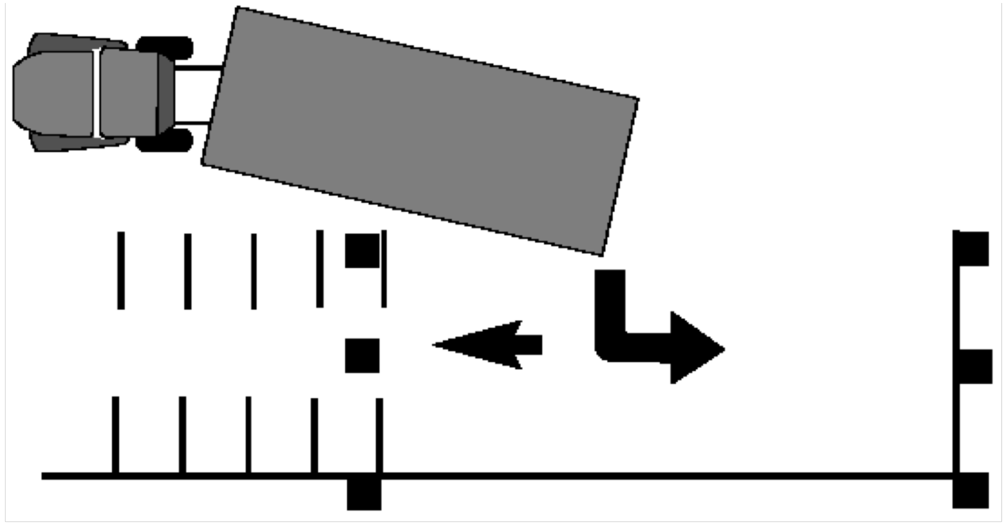


Let the state space *x* be the following augmented C-space

$$x = (q, \dot{q}) \qquad \dot{x} = f(x, u)$$

Differential constraints make things even harder





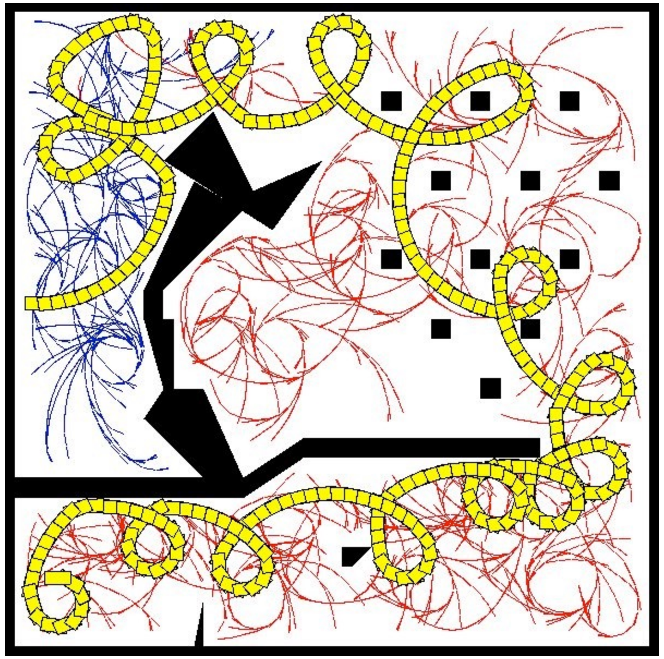
These are examples of non-holonomic system

non-holonomic differential constraints are not completely integrable

i.e. the system is trapped in some sub-manifold of the config space

Differential constraints make things even harder

Emergency landing where
UAV can only turn left



"Left-turning-car"

These are examples of non-holonomic system

non-holonomic differential constraints are not completely integrable

i.e. the system is trapped in some sub-manifold of the config space

Motion planning under differential constraints

1. Given world, obstacles, C-space, robot geometry (same)

2. Introduce state space *X*. Compute free and obstacle state space.

3. Given an action space U

4. Given a state transition equations $\dot{x} = f(x, u)$

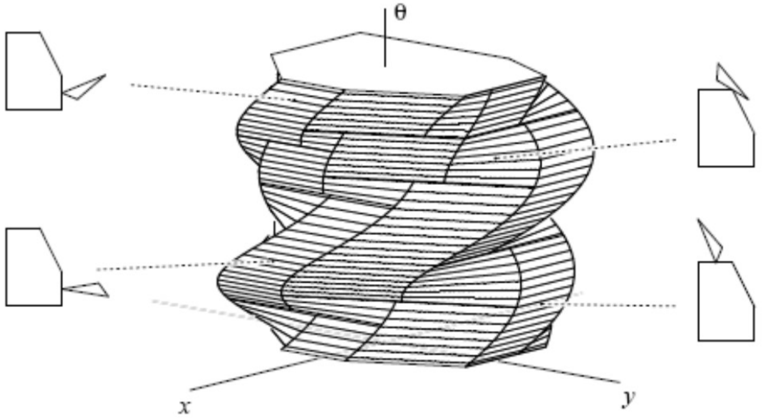
5. Given initial and final state, cost function $J(x(t), u(t)) = \int c(x(t), u(t))dt$

6. Compute action trajectory that satisfies boundary conditions, stays in free state space and minimizes cost.

Challenges in Motion Planning

Computing configuration-space obstacles	**HARD!**
Planning in continuous high-dimensional space	**HARD!**
Underactuated dynamics/constrained system does not allow direct teleportation	**HARD!**

Goal: tractable approximations with provable guarantees!



(EXAMPLE FROM HOWIE CHOSET)

Lecture Outline

Overview of the impact and scope of planning

↓

Formalizing the problem

↓

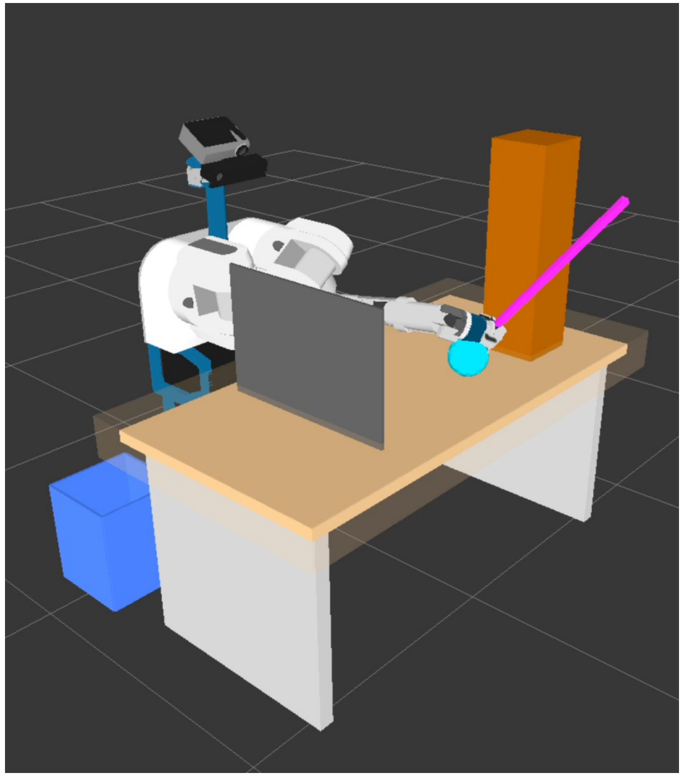
Why is the problem hard?

↓

A start at approaching the problem

Formulating the problem in a way that we can approach

How might we tackle this problem?

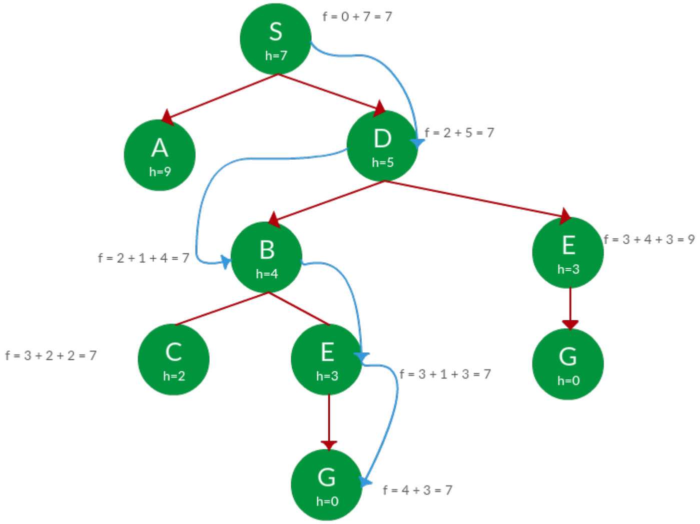


Given:

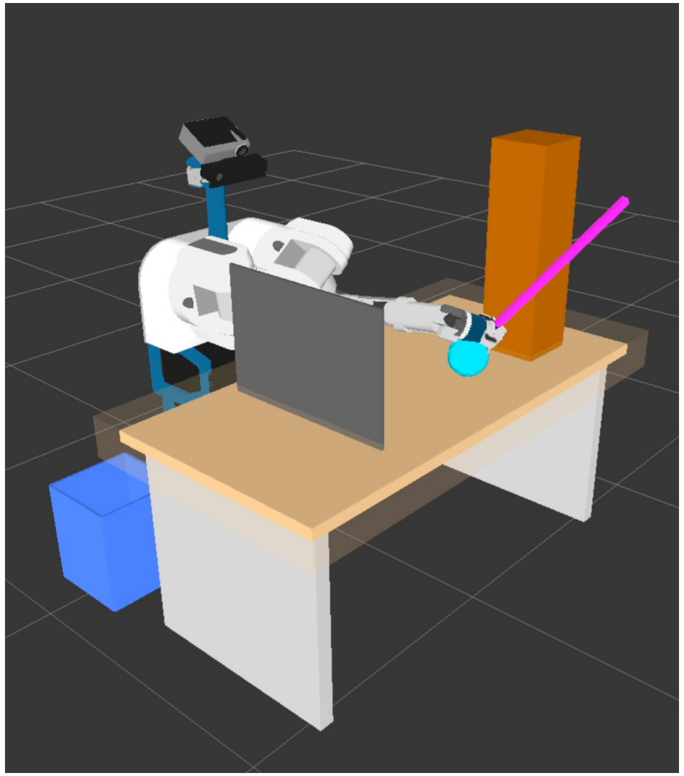
1. A *workspace* $\mathcal{W}$, where either $\mathcal{W} = \mathbb{R}^2$ or $\mathcal{W} = \mathbb{R}^3$.
2. An *obstacle region* $\mathcal{O} \subset \mathcal{W}$.
3. A *robot* defined in $\mathcal{W}$. Either a rigid body $\mathcal{A}$ or a collection of m links: $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m$.
4. The *configuration space* $\mathcal{C}$ (C_{obs} and C_{free} are then defined).
5. An *initial configuration* $\boldsymbol{q_I} \in \mathcal{C}_{free}$.
6. A *goal configuration* $\boldsymbol{q_G} \in \mathcal{C}_{free}$. The initial and goal configuration are often called a *query* $(\boldsymbol{q_I}, \boldsymbol{q_G})$.

Compute a (continuous) path, $\tau : [0, 1] \rightarrow \mathcal{C}_{free}$, *such that* $\tau(0) = \boldsymbol{q_I}$ *and* $\tau(1) = \boldsymbol{q_G}$.

Lets use ideas from search!



How might we tackle this problem?



Given:

1. A *workspace* $\mathcal{W}$, where either $\mathcal{W} = \mathbb{R}^2$ or $\mathcal{W} = \mathbb{R}^3$.
2. An *obstacle region* $\mathcal{O} \subset \mathcal{W}$.
3. A *robot* defined in $\mathcal{W}$. Either a rigid body $\mathcal{A}$ or a collection of m links: $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m$.
4. The *configuration space* $\mathcal{C}$ (C_{obs} and C_{free} are then defined).
5. An *initial configuration* $\boldsymbol{q_I} \in \mathcal{C}_{free}$.
6. A *goal configuration* $\boldsymbol{q_G} \in \mathcal{C}_{free}$. The initial and goal configuration are often called a *query* $(\boldsymbol{q_I}, \boldsymbol{q_G})$.

Compute a (continuous) path, $\tau : [0,1] \rightarrow \mathcal{C}_{free}$, *such that* $\tau(0) = \boldsymbol{q_I}$ *and* $\tau(1) = \boldsymbol{q_G}$.

Continuous space

Hard to characterize obstacles

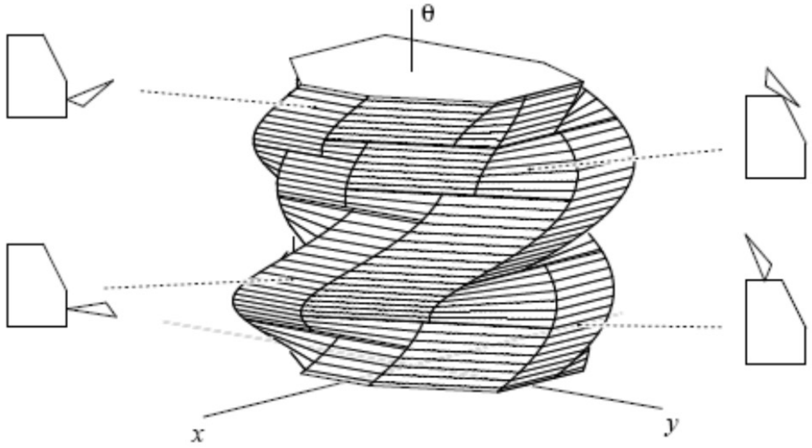
Sampling-Based Motion Planning

Computing configuration-space obstacles is hard

- Use a collision checker instead!

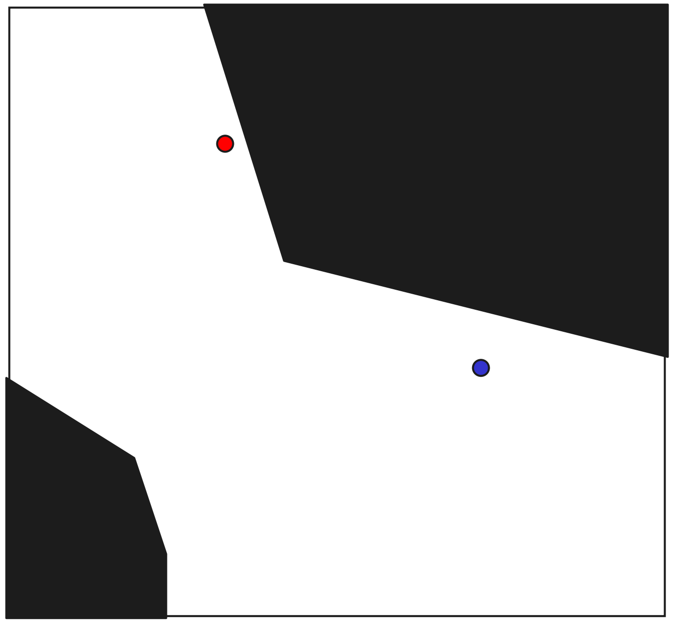
Planning in continuous high-dimensional space is hard

- Construct a discrete graph approximation of the continuous space!

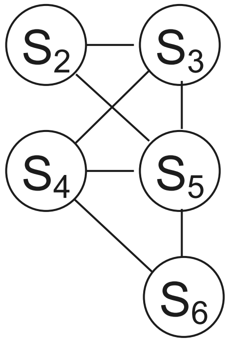


(EXAMPLE FROM HOWIE CHOSET)

Planning as Search



Convert into a search problem



planning map

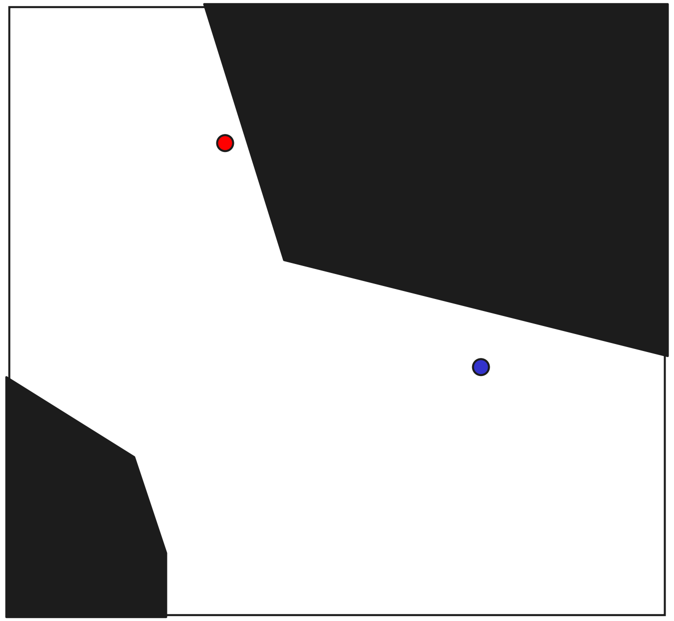
search the graph
for a least-cost path
from s_{start} to s_{goal}

Can use efficient techniques for **<u>discrete</u>** graph search

Explicit graph search

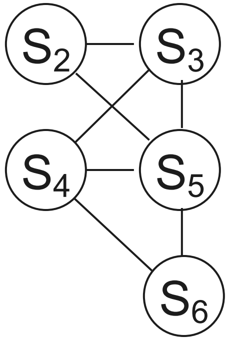
Implicit sampling-based search

Recasting Planning as Search



Convert into a search problem

How?



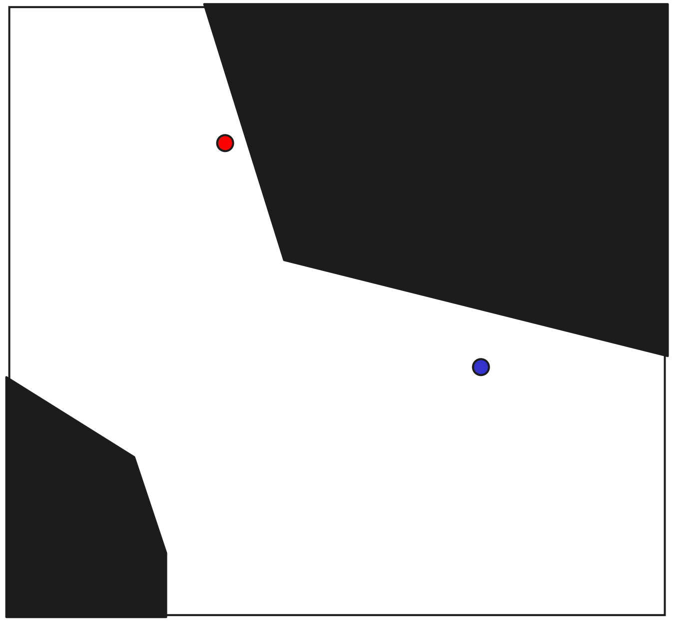
planning map

search the graph for a least-cost path from s_{start} to s_{goal}

Can use efficient techniques for <u>**discrete**</u> graph search

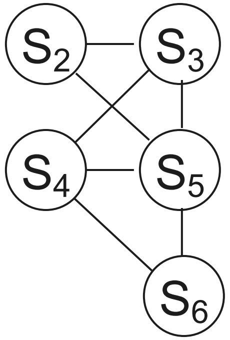
Which ones?

Recasting Planning as Search



Convert into a search problem

How? = Sampling



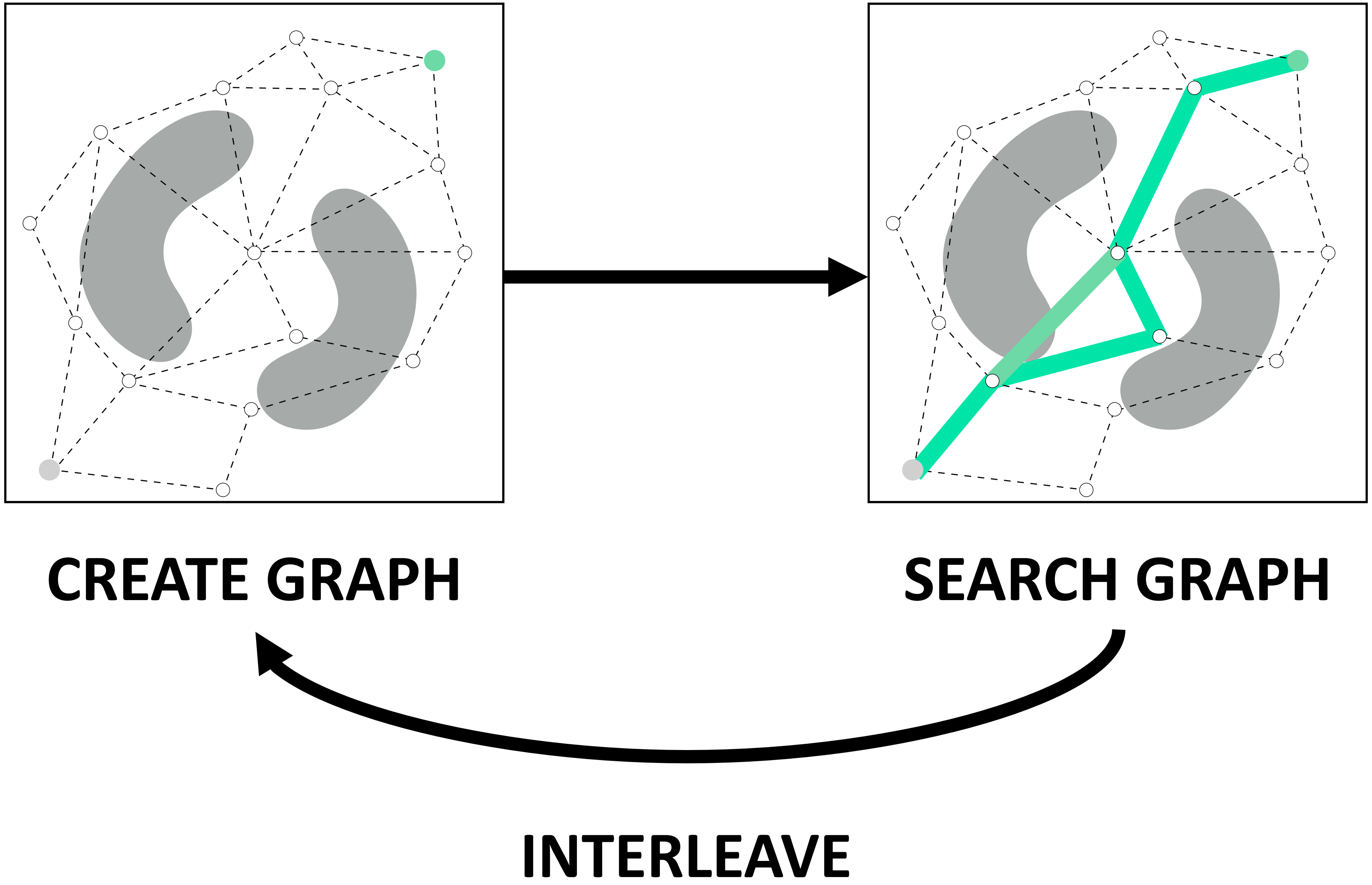
planning map

search the graph
for a least-cost path
from s_{start} to s_{goal}

Can use efficient techniques for **discrete** graph search

Which ones? = Best-first explicit search or Implicit sampling-based graph search

Sampling-Based Motion Planning



Lecture Outline

Overview of the impact and scope of planning

↓

Formalizing the problem

↓

Why is the problem hard?

↓

A start at approaching the problem

Class Outline

State Estimation

- Robotic System Design
- Filtering
- Localization
- SLAM

Control

- Feedback Control
- PID Control
- MPC
- LQR

Planning

- Search
- Heuristic Search
- Motion Planning
- Lazy Search

Learning

- Imitation Learning
- Policy Gradient
- Actor-Critic
- Model-Based RL