AI Robotics

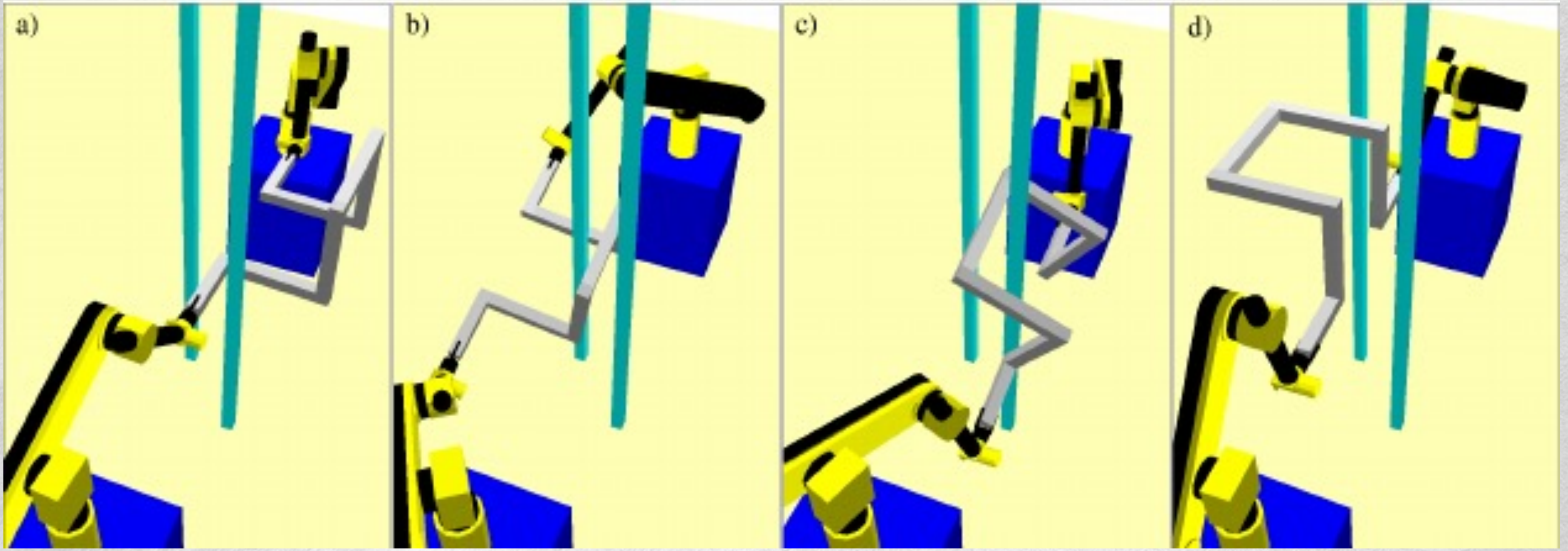
Motion Planning

Overview

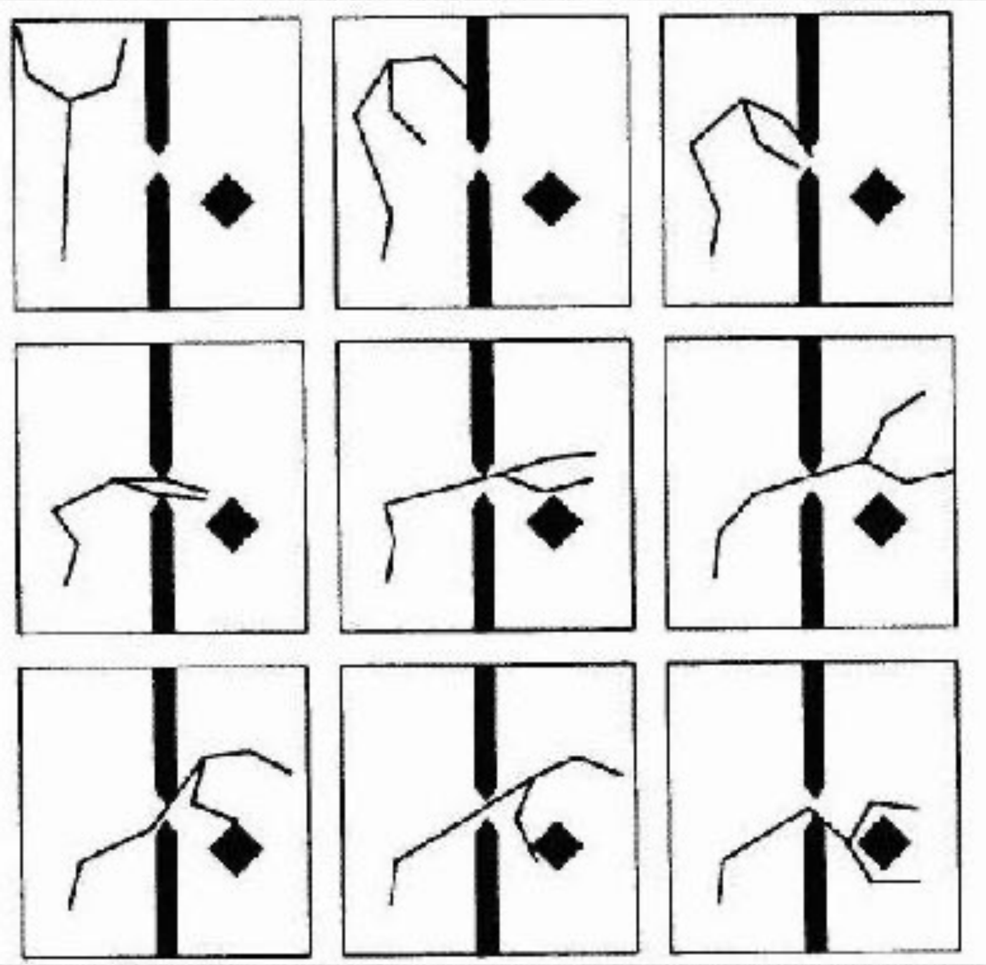
Planning

- Mission planning
  - Which tasks to perform and what order to perform them to achieve a specified goal
  - Tasks may have preconditions and effects
- Path-planning
  - Focus is on robot's (*x*, *y*, *theta*) pose
    - What is it now?
    - What do we want it to be?
    - What sequence of poses will get me from where it is now to where it should be?
- Motion-planning
  - Focus also on robot's pose (called a configuration), but in general this can be a high-dimensional pose, consisting of many joint angles (think a robotic arm)

Motion Planning

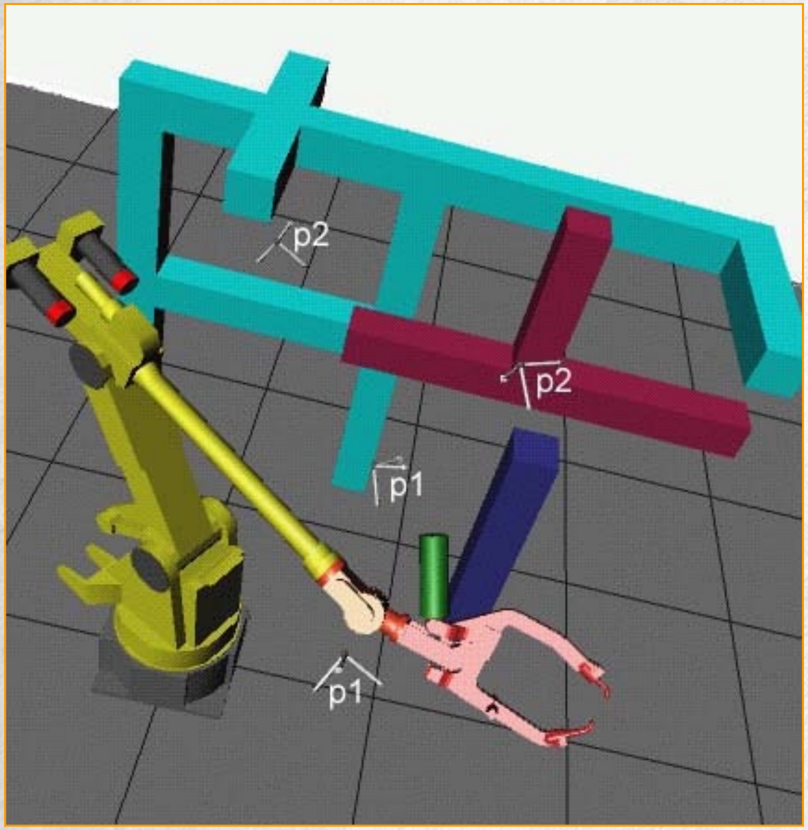


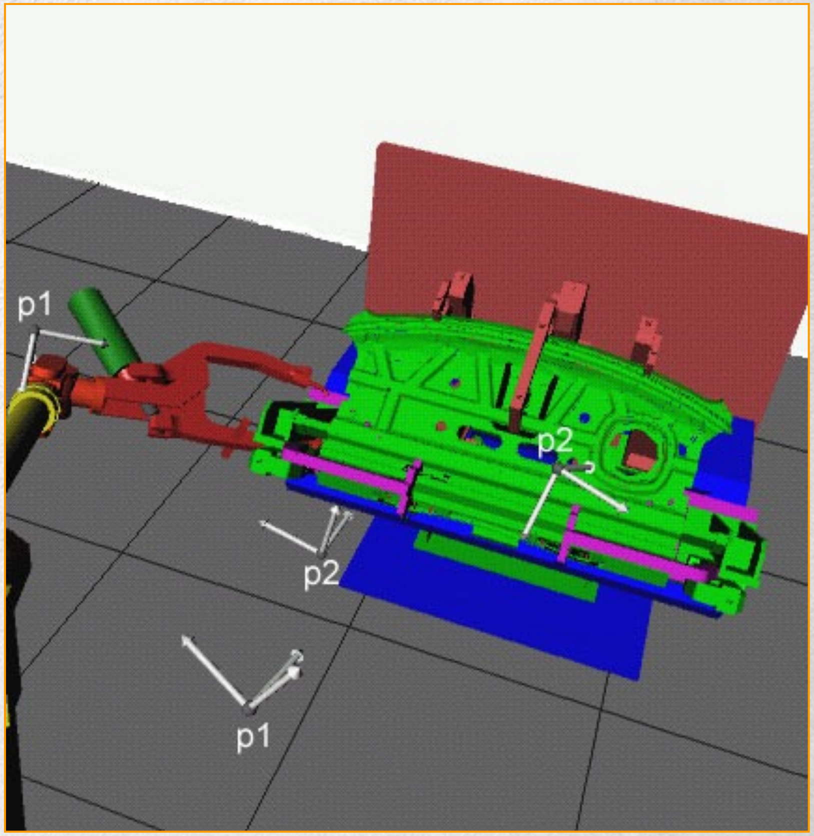
Motion Planning



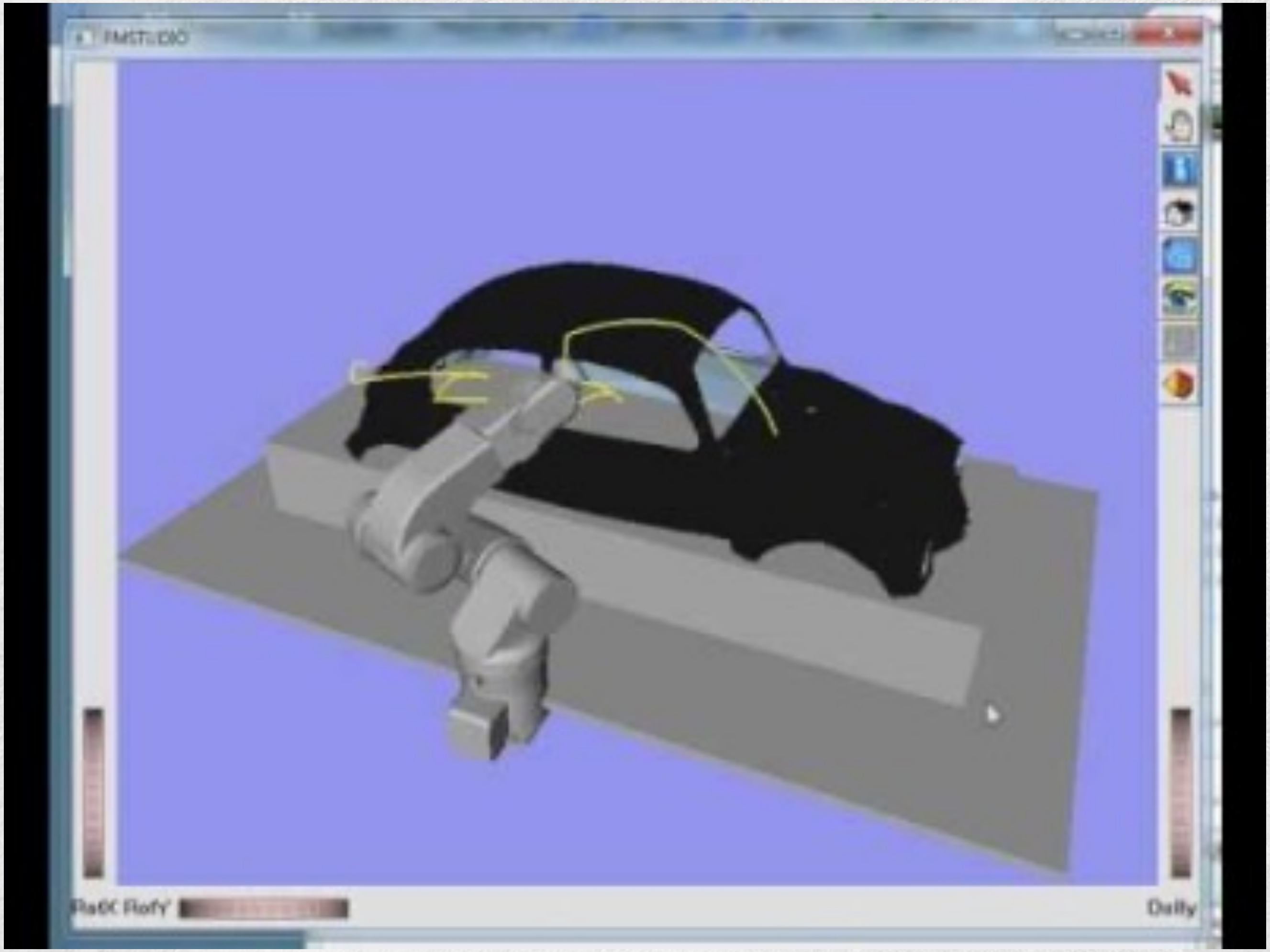
Applications

Robot Programming and Placement





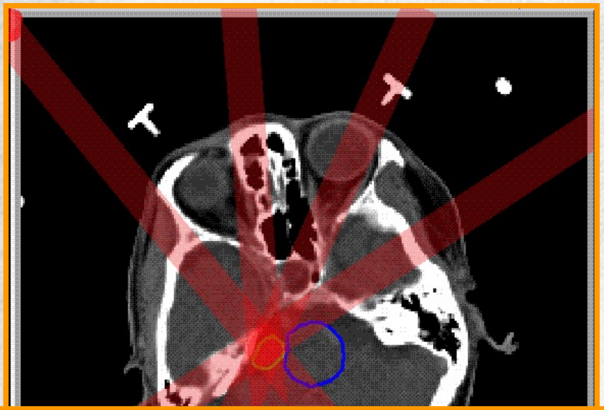
Motion Planning for Manufacturing

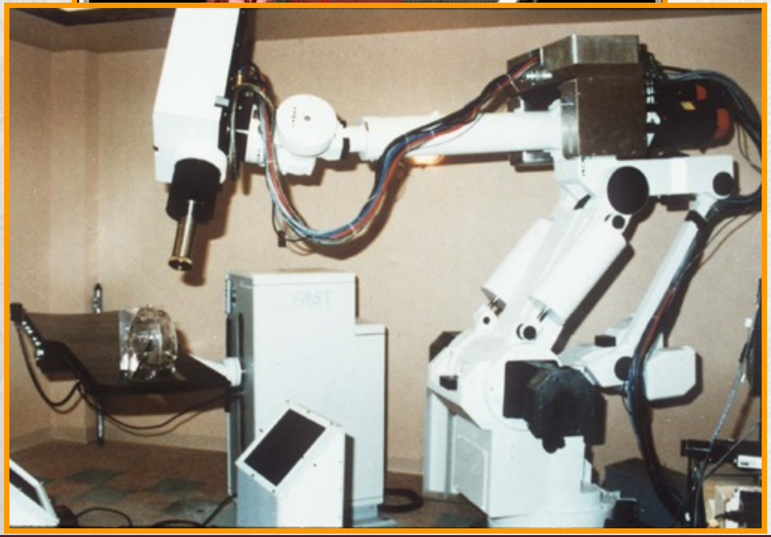


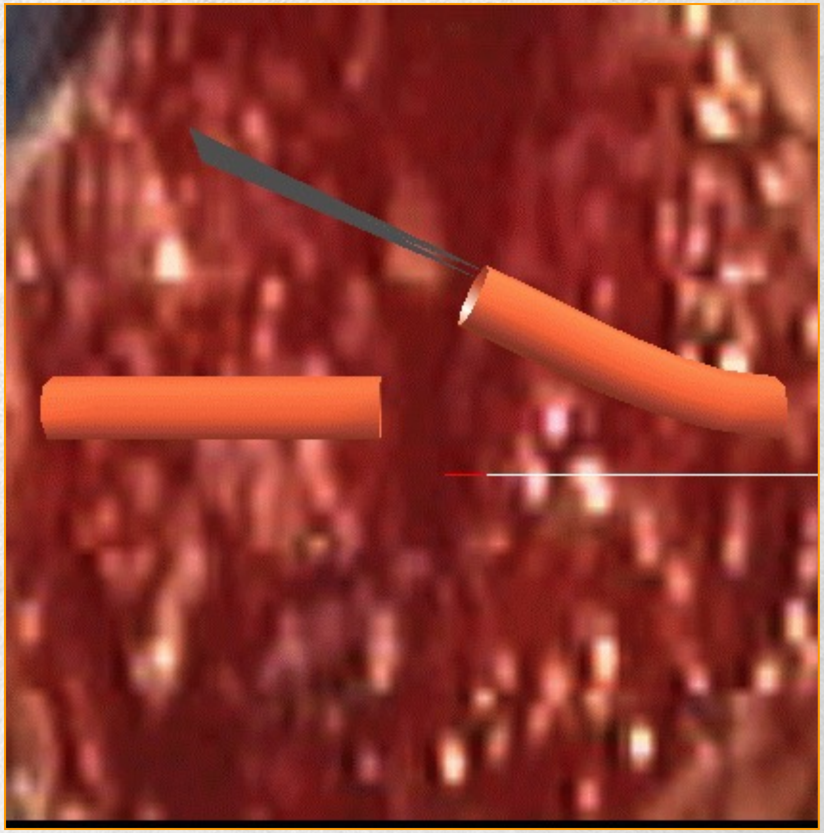
Motion Planning for Humanoid Robot

Motion Planning for Robot Arm

Computer-Assisted Surgical Planning







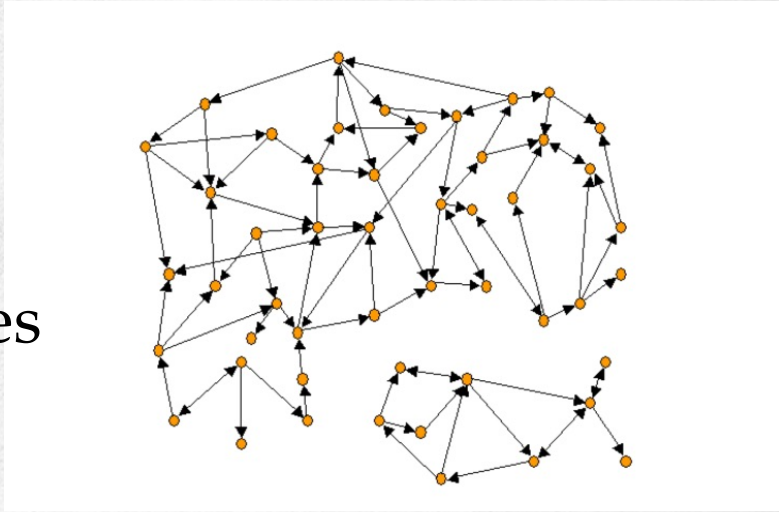
Application: Graphic Animation of Digital Actors

Problem Representation

Basic Problem

- Statement:
  Compute a collision-free path for a rigid or articulated object (the robot) among static obstacles
- Objectives:
  - Avoid self-collisions
  - Avoid collisions with obstacles
- Inputs:
  - Geometry of robot and obstacles
  - Kinematics of robot (degrees of freedom)
  - Initial and goal robot configurations (placements)
- Outputs:
  - Sequence of collision-free robot configurations connecting the initial and goal configurations

Search space



- Vertices – states
- Edges – connect neighboring states

- Planning algorithm needs to find a *solution* or a path from the initial state to the goal state
- The *cost* of a solution is the sum of the costs of each edge in the path
  - For our purposes, this will be the length of the path
- An *optimal solution* is a solution with the lowest cost of any solution

Search Space

- The **Search Space** for a planning problem is the set of all poses/configurations/states that we will planning in
- In robotics problems the search spaces are typically *continuous.* This means that there is an uncountably infinite number of possible states
- The planning algorithms we will discuss today require a *discrete* state space to work with
- How can we come up with a discrete version of a continuous state space?

Configuration Space

- The space of possible configurations of the robot
- Each dimension corresponds to a degree-of-freedom
- Configuration space obstacles correspond to those configurations of the robot (assignment of value to each degree-of-freedom) that would cause robot to intersect a physical obstacle

Configuration Space

Position and orientation of end effector can be determined from joint angles

Configuration, q = $\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$

Position

$$\begin{aligned} x &= x_2 = \alpha_1 \cos\theta_1 + \alpha_2 \cos(\theta_1 + \theta_2) \\ y &= y_2 = \alpha_1 \sin\theta_1 + \alpha_2 \sin(\theta_1 + \theta_2) \end{aligned}$$

Orientation

$$\begin{bmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) \end{bmatrix}$$

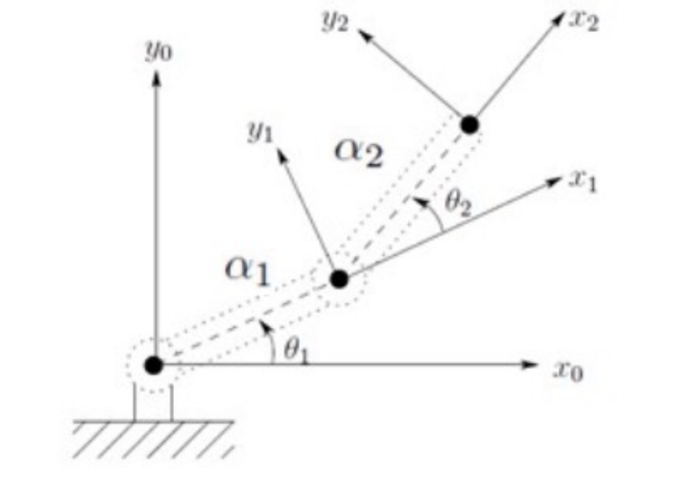
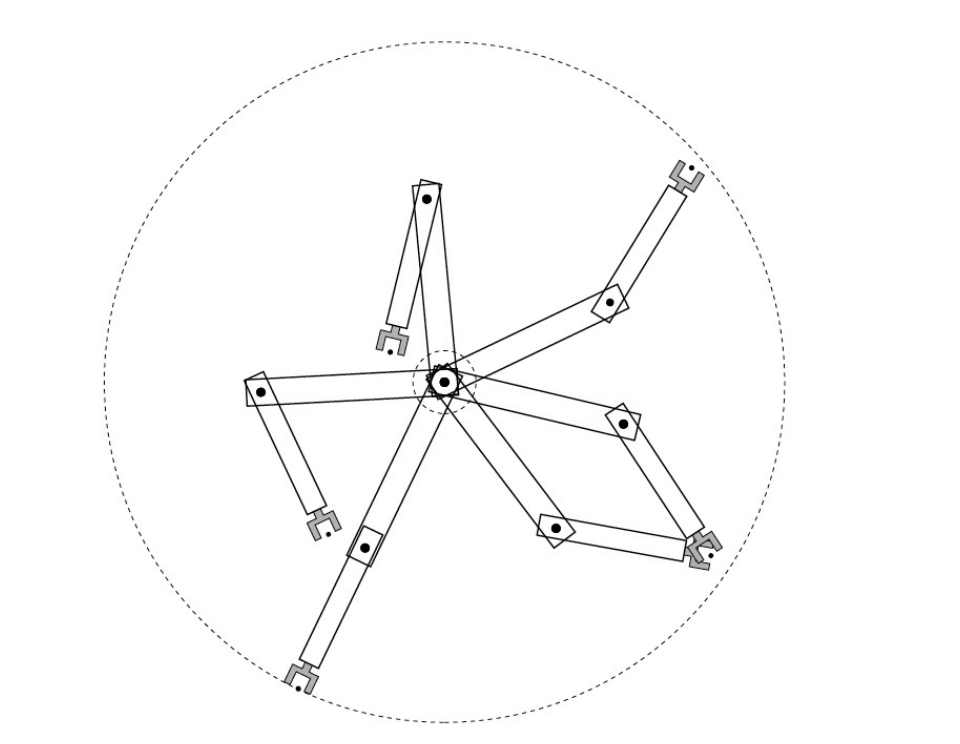
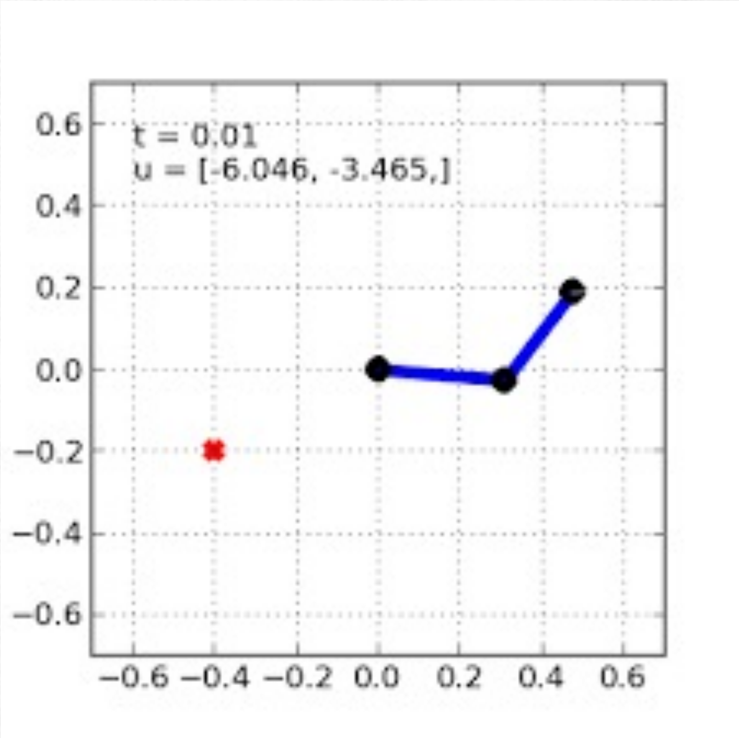


Fig. 1.24 Coordinate frames for two-link planar robot.

Configuration Space





Annulus

Configuration Space

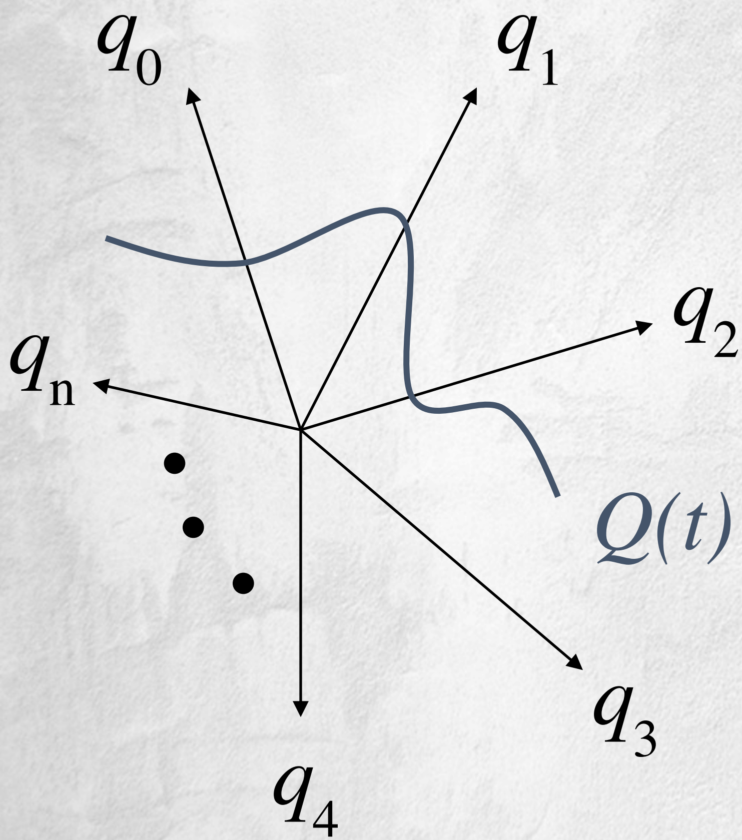
Type of robot	**Representation of $\mathcal{Q}$**
Mobile robot translating in the plane	$\mathbb{R}^2$
Mobile robot translating and rotating in the plane	$SE(2)$ or $\mathbb{R}^2 \times S^1$
Rigid body translating in the three-space	$\mathbb{R}^3$
A spacecraft	$SE(3)$ or $\mathbb{R}^3 \times SO(3)$
An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

R: Euclidean space
SO: rotation space
T: torus

Degrees-of-freedom

number of independent motions available to the robot

High-dimensional Configuration Space



$$Q(t) = \begin{bmatrix} q_0(t) \\ \vdots \\ q_n(t) \end{bmatrix}$$

$$t \in [0, T]$$

Configuration Obstacle

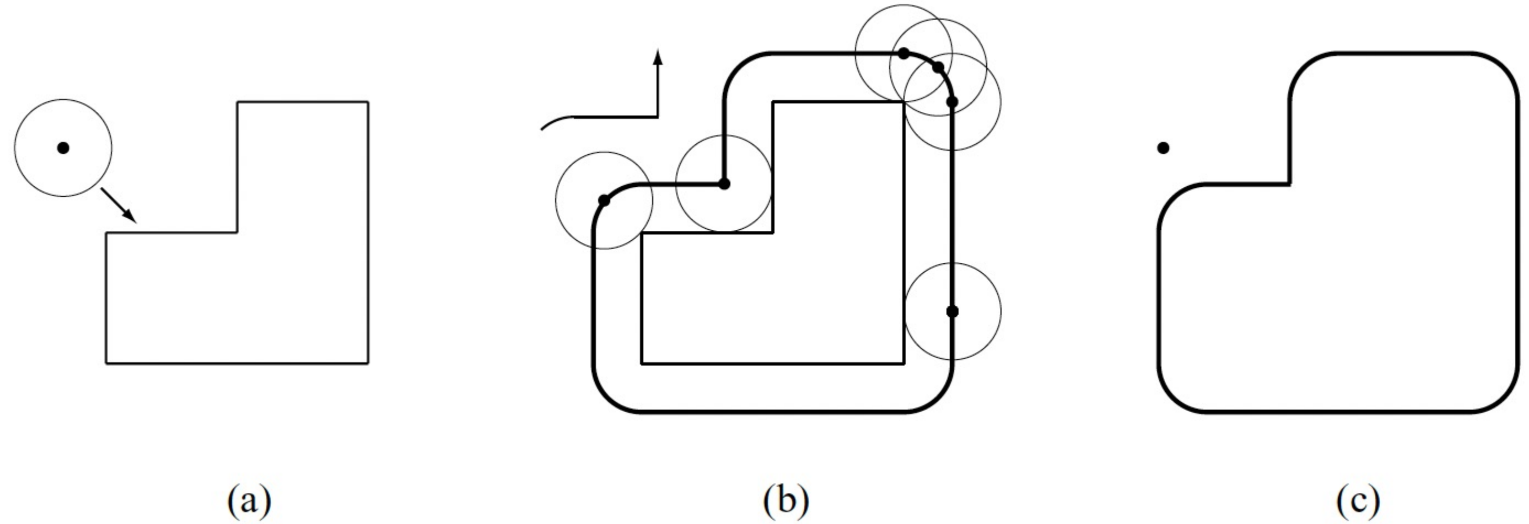


Figure 3.4 (a) The circular mobile robot approaches the workspace obstacle. (b) By sliding the mobile robot around the obstacle and keeping track of the curve traced out by the reference point, we construct the configuration space obstacle. (c) Motion planning for the robot in the workspace representation in (a) has been transformed into motion planning for a point robot in the configuration space.

Configuration Obstacle

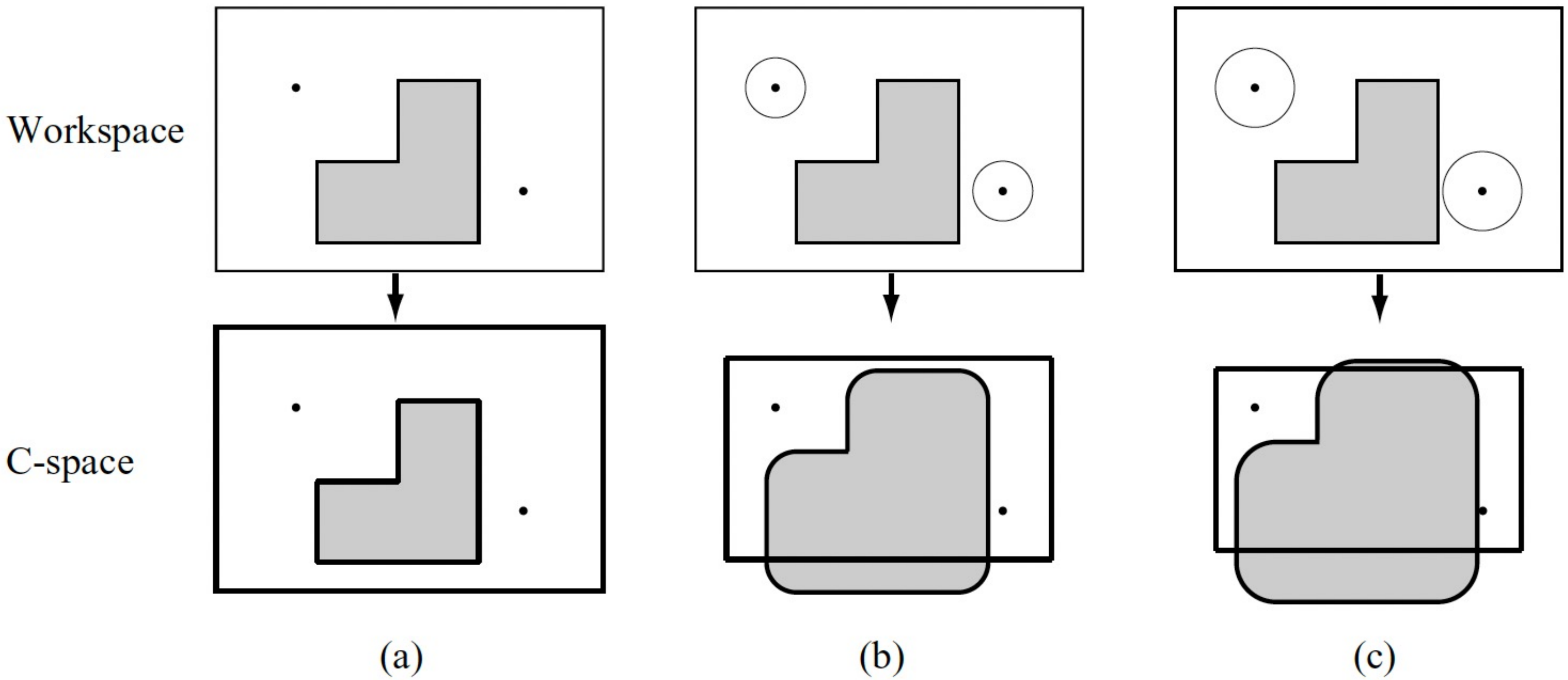
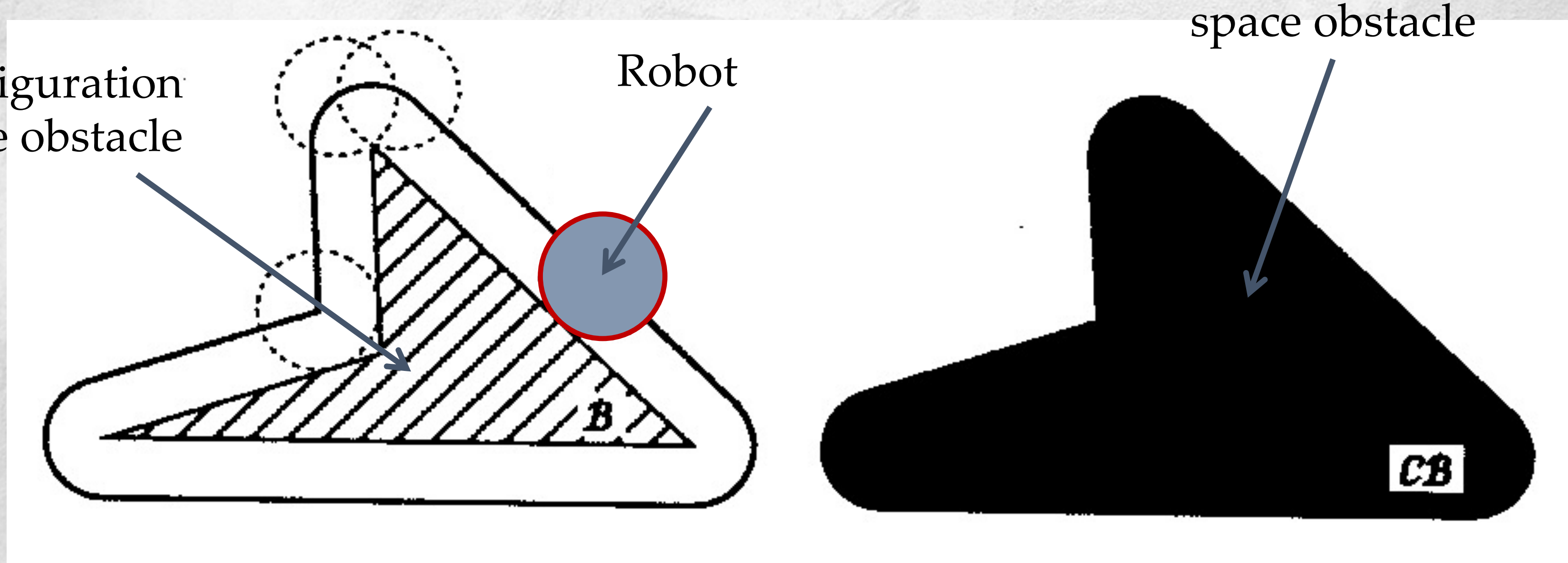


Figure 3.5 The top row shows the workspace and the bottom row shows the configuration space for (a) a point mobile robot, (b) a circular mobile robot, and (c) a larger circular mobile robot.

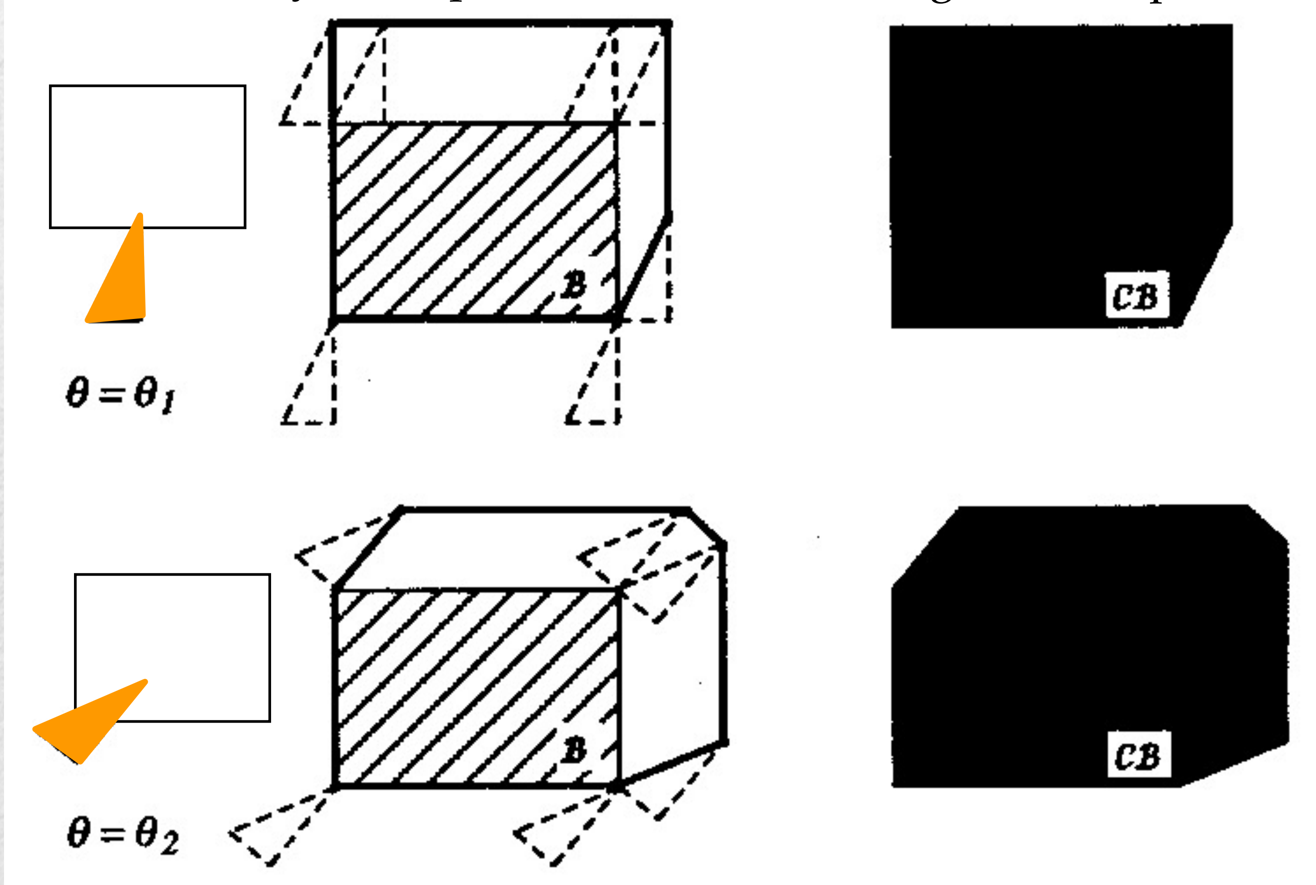
Disc Robot in 2-D Workspace

- Configuration space – (x,y) location of center of robot
  - Same as physical space
- Configuration space obstacle – all robot center locations that would result in the robot



C-Space: Rigid Robot Translating in 2-D

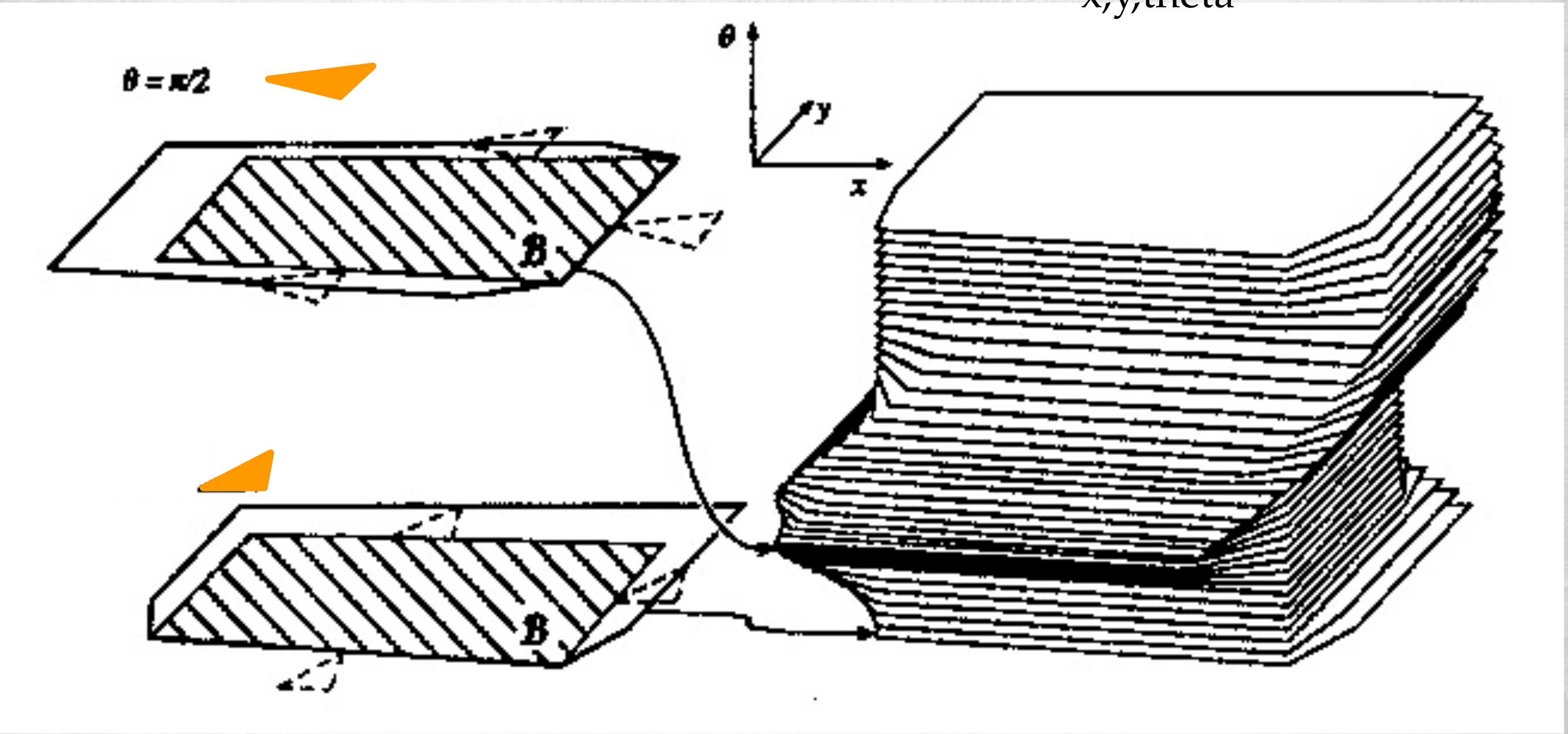
Physical Space | Configuration Space



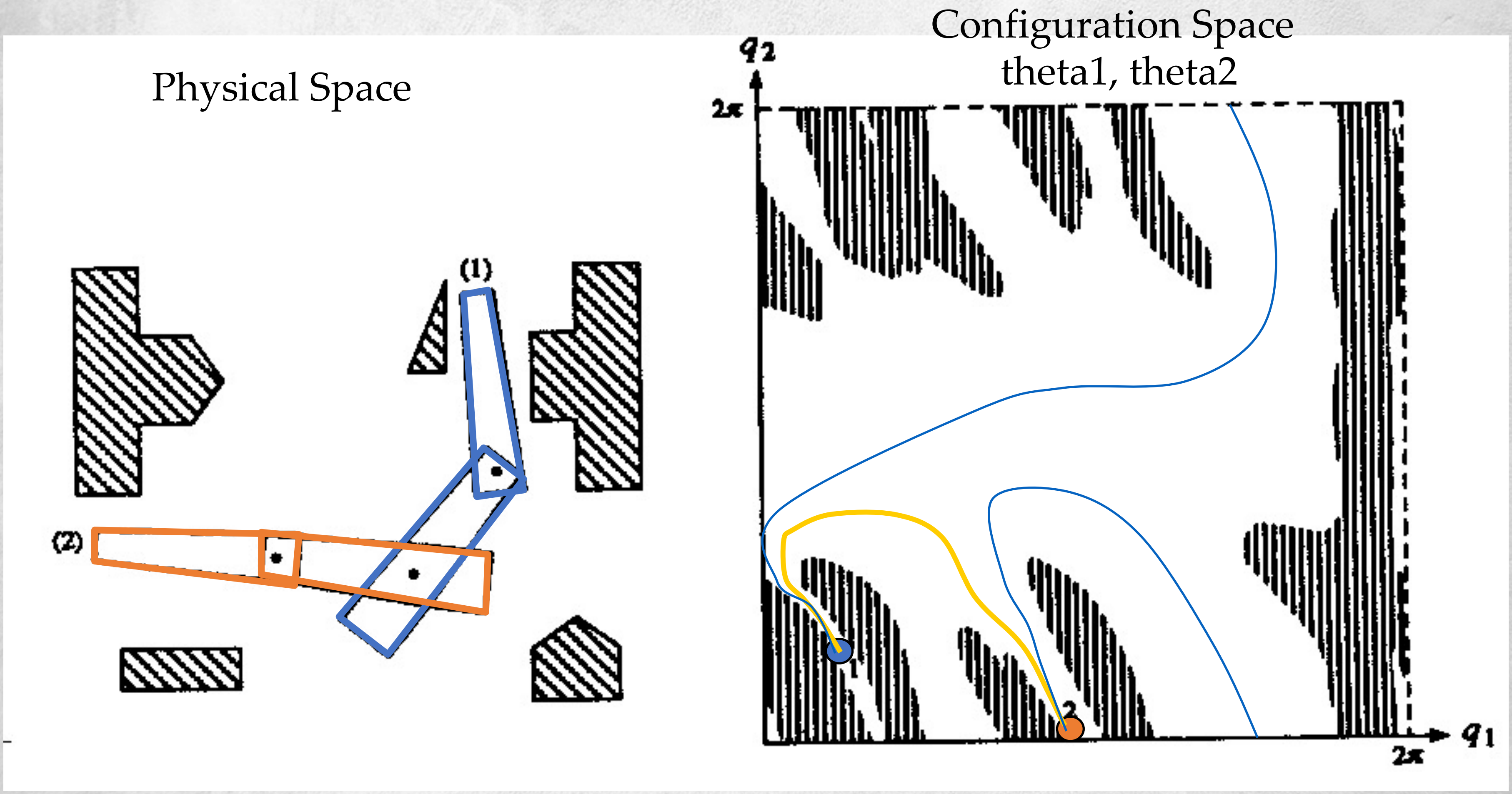
Rigid Robot Translating and Rotating in 2-D

Physical Space

Configuration Space
x,y,theta



C-Obstacle for Articulated Robot



Basic Problem

- Statement:
  Compute a collision-free path for a rigid or articulated object (the robot) among static obstacles
- Objectives:
  – Avoid self-collisions
  – Avoid collisions with obstacles
- Inputs:
  – Geometry of robot and obstacles
  – Kinematics of robot (degrees of freedom)
  – Initial and goal robot configurations (placements)
- Outputs:
  – Sequence of collision-free robot configurations connecting the initial and goal configurations

Motion Planning as a Computational Problem

- Goal:
  Compute the connectivity of a space (e.g., the collision-free subset of configuration space)

- High computational complexity:
  Typically requires time exponential in an input parameter, e.g., the number of degrees of freedom, the number of moving obstacles, …

Solution:

Planning by random sampling

References

1. https://en.wikipedia.org/wiki/Motion_planning
2. https://www.youtube.com/watch?v=xQhCtnd-jgk
3. https://youtu.be/hII68CseRmo