Motion Planning
Part 1

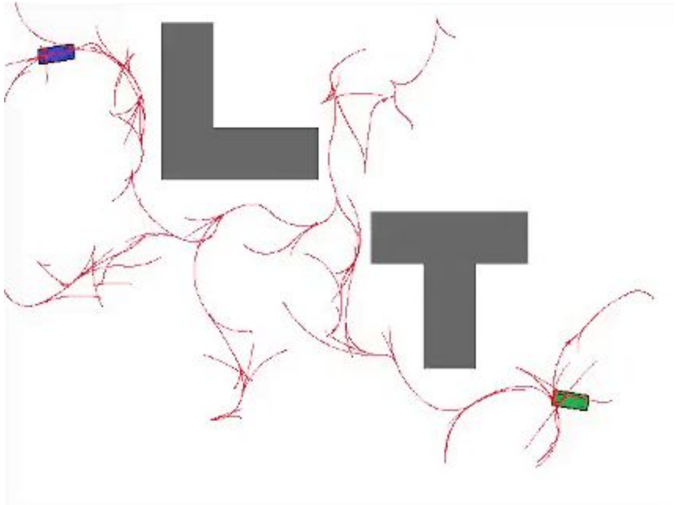
RRC Summer School 2025
June 6, 2025

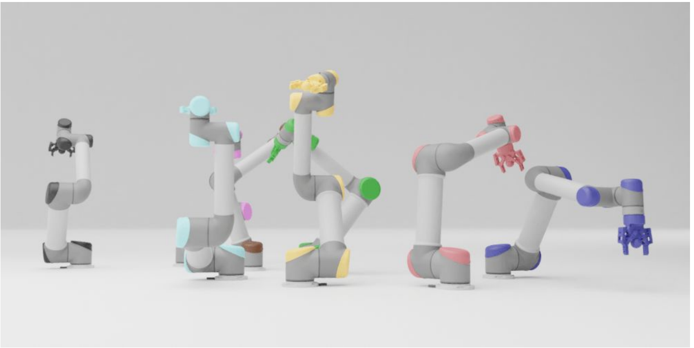
What is Motion Planning?

- Given a **start** and **goal**, compute a sequence of movements while **avoiding obstacles** and satisfying constraints.

- Examples:

    - Robot arm from start → goal, with obstacle in between.
    - Or a car navigating from point A to B on a map.

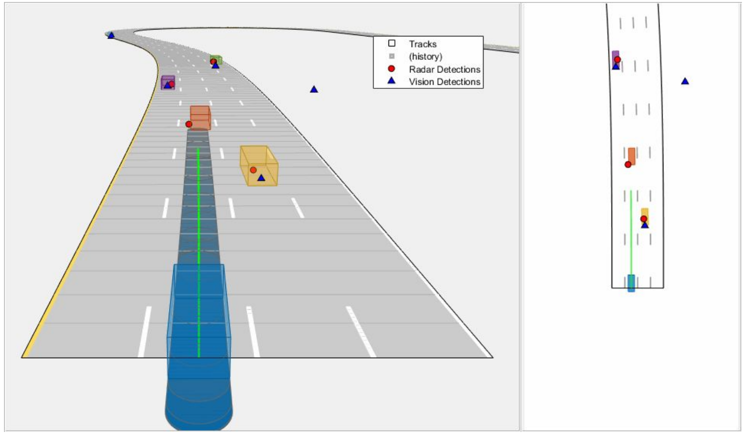




Why is Motion Planning Important?

- Enable autonomous decision-making for:
    - **Navigating** environments
    - **Manipulating** objects
    - **Coordinated** multi-robot tasks

- Human-like behavior: ***plan before act***



Some key terms:

Concept	Meaning
Path	Geometric route in space (no time)
Motion	Feasible movement considering robot dynamics
Trajectory	Time-parameterized path

What are the challenges here?

- Motion planning is robotics is hard because it poses unique challenges (some of which are partially solved and some are still being researched)

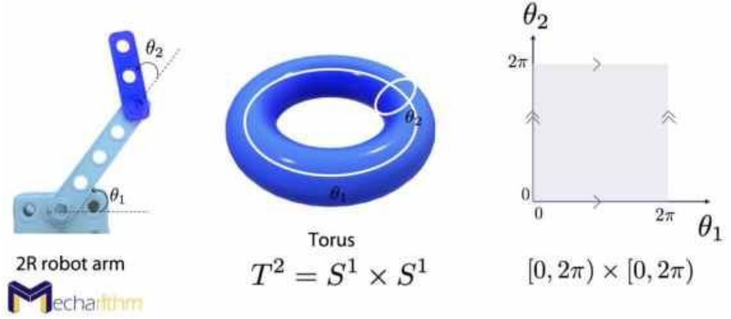
1. High-dimensional configuration spaces
2. Obstacles in complex environments: *dynamic and static obstacles*
3. Robot constraints: *joint limits, dynamics, uneven terrain etc*
4. Real-time requirements

Basics of Motion Planning: C-space

- The **configuration** of a robot is a complete specification of its position.

  C-space = space of all possible robot configurations.

- **Examples:**
  - Point robot in 2D: $(x, y) \rightarrow \mathbb{R}^2$
  - Rigid body in 2D: $(x, y, \theta) \rightarrow SE(2)$
  - Rigid body in 3D: $(x, y, z, \alpha, \beta, \gamma) \rightarrow SE(3)$
  - Robot arm: configuration = all joint angles

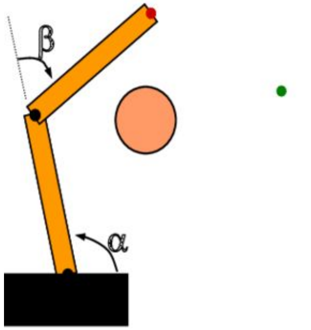


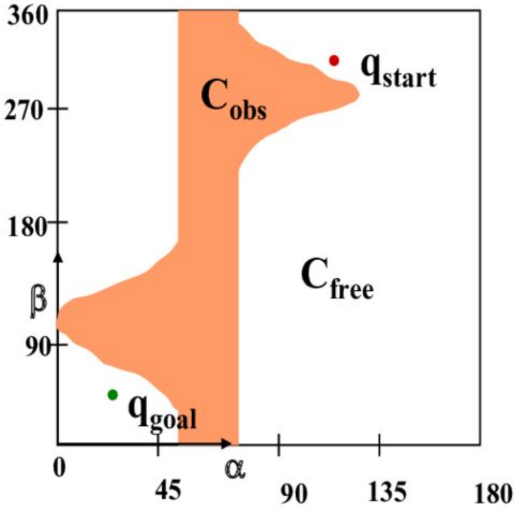
- Why is C-space useful?

  Converts the planning problem into a geometric problem and sampling, search, and optimization algorithms can easily operate in C-space.

Basics of Motion Planning: Obstacles in C-space

- Obstacles in **workspace** get mapped to regions in **C-space**
- A configuration is **invalid** if the robot at that configuration collides with an obstacle





C-obstacle = set of configurations where robot collides with an obstacle

Free space C_{free}= valid configurations

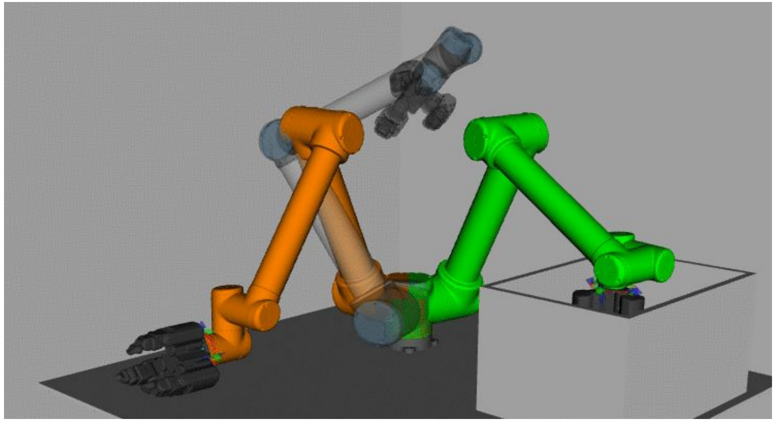
$\mathbf{q_{start}}$: Starting configuration

$\mathbf{q_{goal}}$: Goal configuration

Motion Planning Problem Setup

Given, $q_{\text{start}}, q_{\text{goal}} \in C_{\text{free}}$ and the obstacle map

Goal: Find a continuous path $\tau : [0,1] \rightarrow C_{\text{free}} \quad \text{st } \tau(0) = q_{\text{start}} \text{ and } \tau(1) = q_{\text{goal}}$



Grid based planning: A*

- BFS and Dijkstra's explore too many unnecessary nodes.

- We want an algorithm that: is **complete** (finds a path if one exists), is **optimal** (finds shortest path), is **efficient** (uses fewer resources)

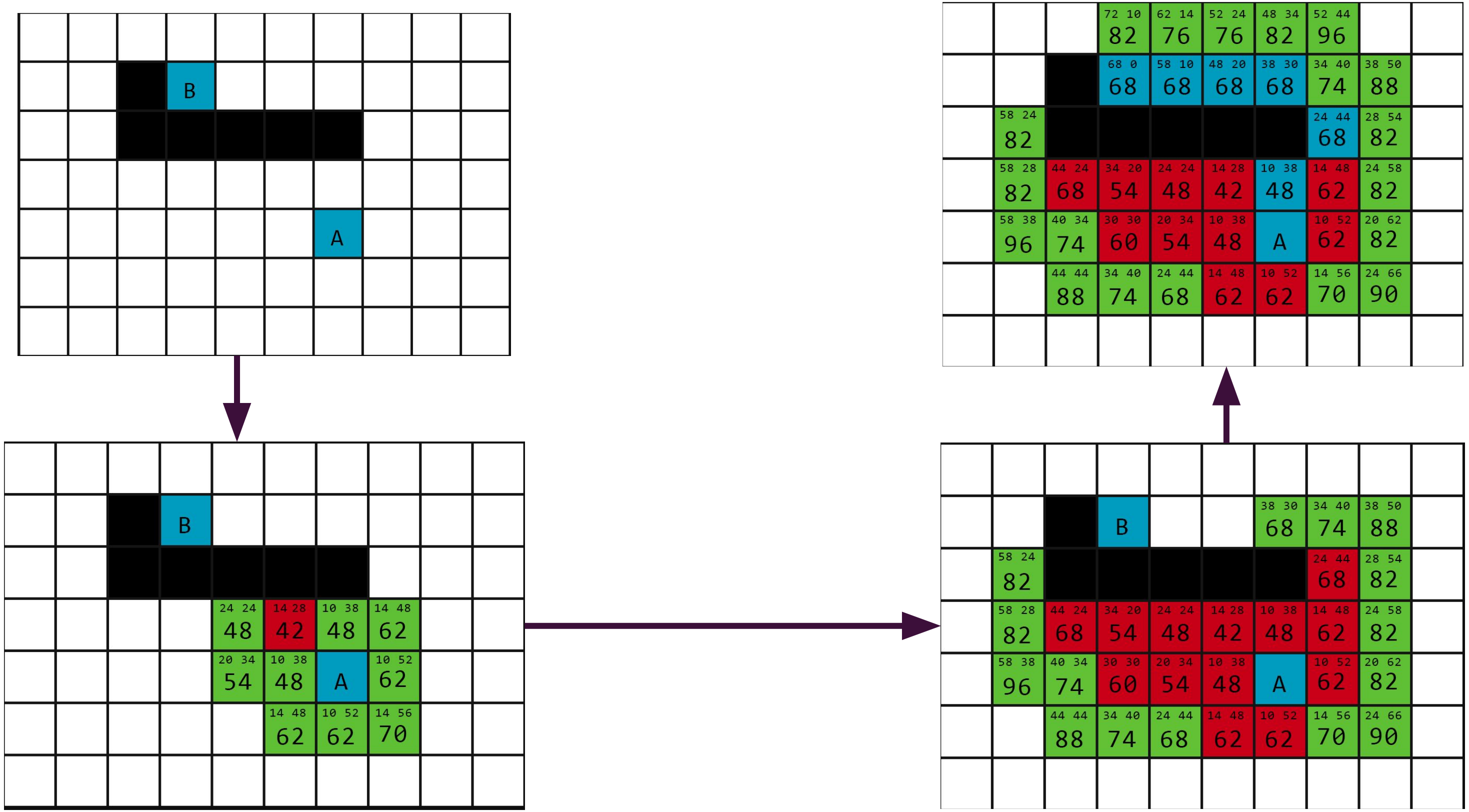
- A* search uses a **heuristic** to guide the search intelligently.

- A* evaluates nodes using: $f(n) = g(n) + h(n)$

  where, $g(n)$: Cost from start to node n
  $h(n)$: Heuristic estimate from n to goal

  and use a priority queue (min-heap) to always expand the node with the lowest $f(n)$

Grid based planning: A*



Issues Grid based planning

- **Resolution Dependency:** Fine grids improve accuracy but increase memory and computation time; coarse grids lose precision and may miss feasible paths.

- **Non-Smooth Paths:** Paths often consist of sharp turns and axis-aligned steps, which are not suitable for real robot motion without post-processing.

- **Curse of Dimensionality:** Grid size grows exponentially with the number of configuration dimensions (e.g., for arms or SE(3)), making it infeasible for high-DOF robots.

The above issues can be tackled using *Sampling-based Planning*.

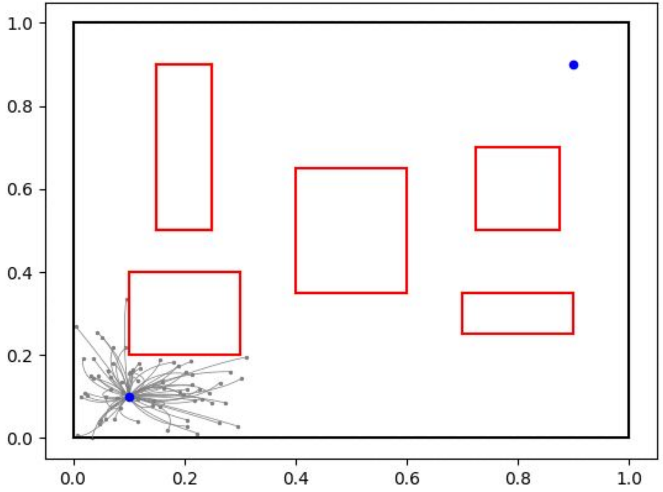
Sampling based planning: RRT

- Grid-based methods suffer from: high memory in high-dimensional spaces, poor performance in complex C-spaces

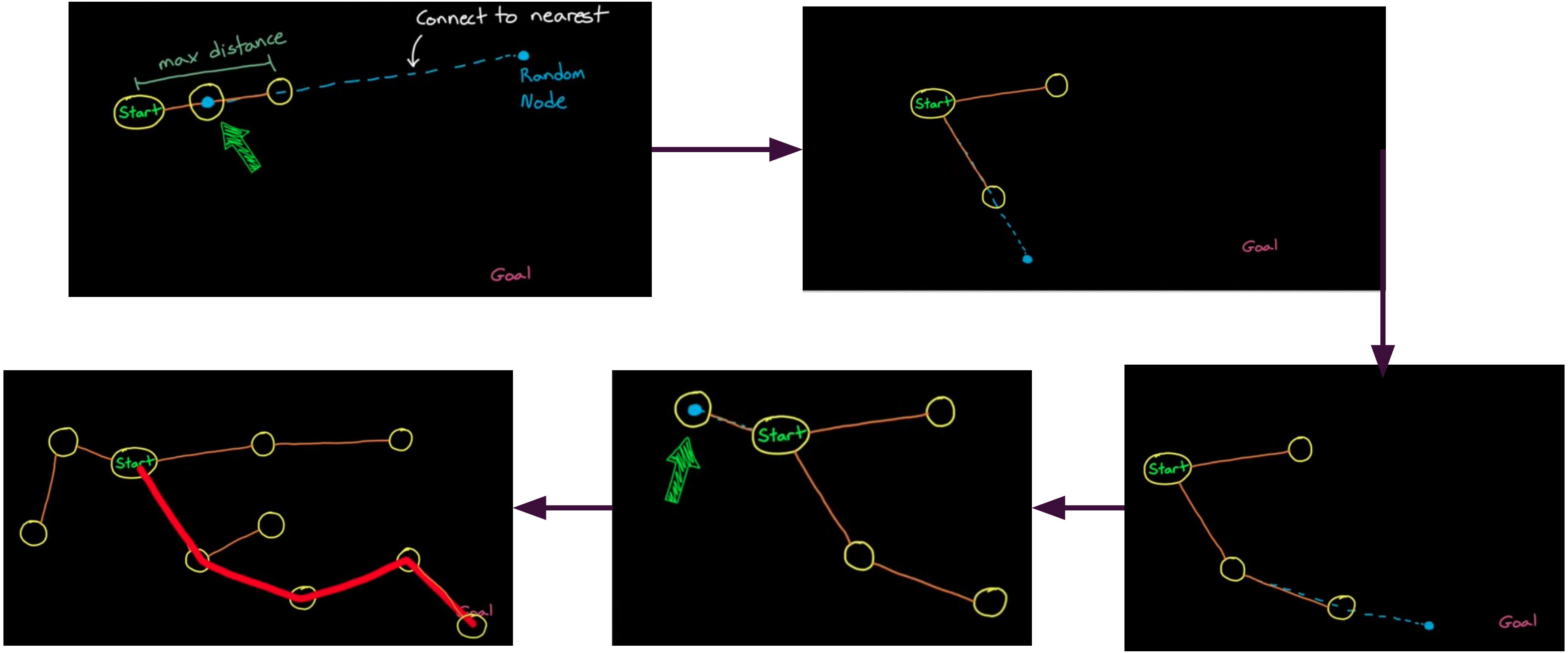
- **Sampling-based methods** work directly in continuous configuration space

- Rapidly-Exploring Random Trees (RRT)

    - Builds a **tree** rooted at the start configuration
    - Randomly samples a point, grows tree **toward** it
    - Efficiently explores large, high-dimensional spaces



Sampling based planning: RRT



Sampling based planning: Bi-RRT

- RRT grows tree from start → goal, but:
  - May take long to reach goal in cluttered spaces
  - Inefficient in narrow passages

Idea: Grow two trees:

- One from start
- One from goal
- Try to **connect** them