Chapter 5: Motion Planning and Path Optimization

🧭 Chapter Overview

Motion planning lies at the core of autonomous robotics, enabling systems to determine feasible, efficient, and safe paths in complex environments. At the advanced level, this task involves integrating geometry, probability, optimization, and dynamics into a coherent framework that allows intelligent agents to navigate uncertain, dynamic worlds.

This chapter explores foundational and cutting-edge techniques in robot motion planning, including deterministic search, sampling-based strategies, trajectory optimization, dynamic obstacle avoidance, and real-time planning in unknown environments. The aim is to equip learners with not just the methods, but the *mathematical intuition* and *engineering trade-offs* behind their practical application.

5.1 Deterministic Search-Based Motion Planning

A* Algorithm (Graph-Based Deterministic Planning)

A* is a best-first search algorithm that uses cost-to-come (`g(n)`) and cost-to-go (`h(n)`) to explore the space:

f(n)=g(n)+h(n)f(n) = g(n) + h(n)f(n)=g(n)+h(n)

- **Admissibility**: If `h(n)` never overestimates the true cost, A* guarantees optimality.
- **Complexity**: Exponential in worst-case; mitigated via heuristics.
- **Limitations**: Struggles with high-dimensional or continuous spaces.

A* is fundamental to many embedded path planners in discrete environments, such as mobile robot grid maps or roadmap navigation in automated warehouses.

D* and D* Lite

D* (Dynamic A*) extends A* to accommodate *changing environments*. It efficiently updates the existing plan as new information (e.g., obstacles) becomes available. This is especially useful in semi-structured outdoor navigation, planetary rovers, and autonomous vehicles where the map is incomplete or dynamic.

D* Lite is a simplified version with reduced overhead and is widely used in mobile robotics. It builds on the concept of *incremental search*, reducing the need to replan from scratch.

5.2 Sampling-Based Motion Planning

As the dimensionality of the robot's configuration space increases, deterministic methods become infeasible. Sampling-based planners are designed to address these challenges by *probabilistically exploring* the space.

Rapidly-Exploring Random Tree (RRT)

RRT is designed for pathfinding in high-dimensional continuous spaces. The algorithm incrementally builds a tree rooted at the start configuration:

- Randomly samples a point in the configuration space
- Connects the nearest tree node to the sample if the motion is collision-free
- Repeats until the goal is reached or a time limit is exceeded

> ⚠️ RRT does **not guarantee optimality**. It excels in fast, feasible pathfinding for systems like UAVs and robotic arms with many joints.

RRT*

An extension of RRT, RRT* introduces rewiring to optimize paths by minimizing a cost function. It is both *probabilistically complete* and *asymptotically optimal*:

limn→∞P(costRRT∗→costoptimal)=1\lim_{n \to \infty} P(\text{cost}_{RRT^*} \to \text{cost}_{\text{optimal}}) = 1n→∞limP(costRRT∗→costoptimal)=1

Probabilistic Roadmaps (PRM)

PRMs are suited for *multi-query* applications (e.g., factory floors, surgical environments):

- Preprocessing: Sample valid configurations and build a roadmap by connecting neighboring samples via local planners.
- Querying: Run a shortest path algorithm (like A*) over the roadmap.

Used in semi-static environments, PRMs separate computation into offline (learning) and online (execution) stages.

5.3 Trajectory Optimization for Smooth and Feasible Paths

Motion planning must not only find a collision-free path but also generate a trajectory that respects **dynamics**, **kinematic constraints**, and **task goals**.

Objective

Given a path $\{x_0, x_1, \dots, x_n\}$, find a trajectory that minimizes:

$$J = \sum_{i=1}^{n} \left(\|x_{i} - x_{i-1}\|^2 + \lambda \cdot C(x_i) \right)$$

Where:

- $\|x_{i} - x_{i-1}\|$: smoothness
- $C(x_i)$: collision cost
- λ: weighting factor

Common Optimization Methods

- **CHOMP (Covariant Hamiltonian Optimization)**: Gradient-based method optimizing trajectories in continuous space.
- **TrajOpt**: Uses sequential convex optimization with collision checking.
- **STOMP (Stochastic Trajectory Optimization)**: Samples noisy trajectories and uses cost weighting to refine paths.

These methods work well for manipulators in constrained spaces (e.g., surgical robotics) or humanoid robots requiring smooth, stable walking gaits.

5.4 Dynamic Obstacle Avoidance

Robots in real environments must handle **moving obstacles** — humans, vehicles, animals, or other robots — in **non-deterministic ways**.

Approaches

Velocity Obstacle (VO)

Calculates the set of robot velocities that will result in a future collision and avoids them.

Mathematically, the VO is:

VOA∣B={vA∣∃t>0:pA+vAt=pB+vBt}VO_{A|B} = \{v_A \mid \exists t > 0 : p_A + v_A t = p_B + v_B t\}VOA∣B={vA∣∃t>0:pA+vAt=pB+vBt}

Used extensively in swarm robotics and mobile robot navigation in shared spaces.

Dynamic Window Approach (DWA)

Instead of searching in configuration space, DWA samples velocities and chooses the one that:

- Avoids obstacles
- Progresses toward the goal
- Respects dynamic limits

Artificial Potential Fields (APF)

- **Goal**: Attractive force
- **Obstacles**: Repulsive force

While intuitive, APFs can suffer from local minima, making them unsuitable for complex maps unless combined with global planners.

5.5 Real-Time Planning in Unknown Environments

In many applications (e.g., autonomous exploration, disaster robotics), the robot must plan in **partially or completely unknown** environments.

Techniques

Frontier-Based Exploration

Detects boundaries between known and unknown areas and directs motion toward them. Widely used in SLAM-enabled systems.

Incremental Replanning

Combines mapping (e.g., with LiDAR or visual SLAM) with planning:

- Continuously updates local and global maps
- Replans as new data becomes available
- Uses D*, LPA*, or other incremental algorithms

Hierarchical Planning

- **High-level Planner**: Handles symbolic goals and long-term paths
- **Mid-level Planner**: Manages terrain-aware planning
- **Low-level Planner**: Handles actuation, obstacle avoidance

> This decoupling is essential for managing complexity in autonomous driving, robotic mining, and drone delivery.

🧠 Advanced Concepts and Research Directions

- **Learning-Based Planning**: Integrating neural networks to guide sampling or learn cost functions (e.g., Neural RRT).
- **Multi-Agent Path Planning (MAPF)**: Coordinated planning in robot fleets (e.g., warehouse swarms).
- **Hybrid Planning**: Combining symbolic reasoning with geometric motion planning (e.g., Task and Motion Planning - TAMP).
- **Risk-Aware Planning**: Probabilistic motion planning under uncertainty, accounting for stochastic disturbances and model errors.

✅ Chapter Summary

- Deterministic algorithms like A* and D* form the foundation of path planning.
- Sampling-based methods like RRT and PRM are scalable to high-dimensional spaces.
- Trajectory optimization enhances path smoothness and dynamic feasibility.
- Dynamic obstacle avoidance integrates perception with reactive control strategies.
- Real-time planning in unknown terrain demands adaptability and robustness.
