Mobile Robotics

Configuration Space – Basic Path-Planning Methods

Key ideas

- How to design real vehicles
- What limitations real vehicles impose and how to model them
- How to represent the nature of these problems in a general mathematical way
- How to plan a path from here to there
- What kinds of sensors can we use and how to interpret the data

Motion planning

- Motion planning is about moving between places. Wait, that’s not general enough
  - What about robot arms?
  - A more general notion of a place, is a pose or a configuration (these are essentially synonyms in this context).
  - A place is a location in space, but does not normally include things like rotation and arm position. A pose includes whatever we care about.
- Going from one pose to another includes moving along a path.

What is a Path?
Definition

- A robot configuration is a specification of the positions of all robot points relative to a fixed coordinate system.

- Usually, a configuration is expressed as a "vector" of position/orientation parameters.

Rigid Robot Example

- 3-parameter representation: $q = (x,y,0)$
- In a 3-D workspace, $q$ would be of the form $(x,y,z,\alpha,\beta,\gamma)$
Articulated Robot Example

$q = (q_1, q_2, ..., q_{10})$

Configuration Space of a Robot

- Space of all its possible configurations
- But the topology of this space is usually not that of a Cartesian space

Configuration Space of a Robot

- Space of all its possible configurations
- But the topology of this space is usually not that of a Cartesian space

$C = S^1 \times S^1$
Structure of Configuration Space

- It is a manifold
  For each point $q$, there is a 1-to-1 map between a neighborhood of $q$ and a Cartesian space $\mathbb{R}^n$, where $n$ is the dimension of $C$
- This map is a local coordinate system called a chart.
  $C$ can always be covered by a finite number of charts. Such a set is called an atlas

Example

Case of a Planar Rigid Robot

- 3-parameter representation: $q = (x,y,\theta)$ with $\theta \in [0,2\pi)$. Two charts are needed
- Other representation: $q = (x,y,\cos \theta, \sin \theta)$
  → $c$-space is a 3-D cylinder $\mathbb{R}^2 \times S^1$
  embedded in a 4-D space

Rigid Robot in 3-D Workspace

- $q = (x,y,z,\alpha,\beta,\gamma)$
  This $c$-space is a 6-D space (manifold) embedded in a 12-D Cartesian space. It is denoted by $R^3 \times SO(3)$
- Other representation: $q = (x,y,z,r_{11},r_{12},...,r_{33})$ where $r_{ij}$ are the elements of rotation matrix $R$:
  \[
  \begin{pmatrix}
  r_{11} & r_{12} & r_{13} \\
  r_{21} & r_{22} & r_{23} \\
  r_{31} & r_{32} & r_{33}
  \end{pmatrix}
  \]
  with:
  - $r_{11}^2 + r_{21}^2 + r_{31}^2 = 1$
  - $r_{12}r_{21} + r_{13}r_{31} = 0$
  - $\det(R) = +1$
Parameterization of SO(3)

- Euler angles: \((\phi, \theta, \psi)\)
- Unit quaternion: 
  \[
  (\cos \frac{\theta}{2}, n_1 \sin \frac{\theta}{2}, n_2 \sin \frac{\theta}{2}, n_3 \sin \frac{\theta}{2})
  \]

Metric in Configuration Space

A metric or distance function \(d\) in \(C\) is a map 
$$d: (q_1, q_2) \in C^2 \to d(q_1, q_2) > 0$$
such that:
- \(d(q_1, q_2) = 0\) if and only if \(q_1 = q_2\)
- \(d(q_1, q_2) = d(q_2, q_1)\)
- \(d(q_1, q_2) \leq d(q_1, q_3) + d(q_3, q_2)\)

Metric in Configuration Space

Example:
- Robot \(A\) and point \(x\) of \(A\)
- \(x(q)\): location of \(x\) in the workspace when \(A\) is at configuration \(q\)
- A distance \(d\) in \(C\) is defined by: 
  $$d(q, q') = \max_{x \in A} ||x(q) - x(q')||$$
  where \(||a - b||\) denotes the Euclidean distance between points \(a\) and \(b\) in the workspace

Notion of a Path

- A path in \(C\) is a piece of continuous curve connecting two configurations \(q\) and \(q'\):
  \(\tau: s \in [0,1] \to \tau(s) \in C\)
- \(s' \to s \Rightarrow d(\tau(s), \tau(s')) \to 0\)
Other Possible Constraints on Path

- Finite length, smoothness, curvature, etc...
- A trajectory is a path parameterized by time:
  \[ \tau : t \in [0,T] \rightarrow \tau(t) \in C \]

Obstacles in C-Space

- A configuration \( q \) is collision-free, or free, if the robot placed at \( q \) has null intersection with the obstacles in the workspace.
- The free space \( F \) is the set of free configurations.
- A C-obstacle is the set of configurations where the robot collides with a given workspace obstacle.
- A configuration is semi-free if the robot at this configuration touches obstacles without overlap.

Disc Robot in 2-D Workspace

Rigid Robot Translating in 2-D

\[ CB = B \Theta A = \{ b-a | a \in A, b \in B \} \]
Linear-Time Computation of C-Obstacle in 2-D

(convex polygons)

Rigid Robot Translating and Rotating in 2-D

C-Obstacle for Articulated Robot

Free and Semi-Free Paths

- A free path lies entirely in the free space F
- A semi-free path lies entirely in the semi-free space
Remark on Free-Space Topology

- The robot and the obstacles are modeled as closed subsets, meaning that they contain their boundaries.
- One can show that the C-obstacles are closed subsets of the configuration space C as well.
- Consequently, the free space $F$ is an open subset of $C$. Hence, each free configuration is the center of a ball of non-zero radius entirely contained in $F$.
- The semi-free space is a closed subset of $C$. Its boundary is a superset of the boundary of $F$.

Notion of Homotopic Paths

- Two paths with the same endpoints are **homotopic** if one can be continuously deformed into the other.
- $\mathbb{R} \times \mathbb{S}^1$ example:

- $\tau_1$ and $\tau_2$ are homotopic.
- $\tau_2$ and $\tau_3$ are not homotopic.
- In this example, infinity of homotopy classes.
**Connectedness of C-Space**

- **Connected**: If every two configurations can be connected by a path.
- **Simply-connected**: If any two paths connecting the same endpoints are homotopic.
- **Multiply-connected**: Otherwise.

Examples:
- $\mathbb{R}^2$ or $\mathbb{R}^3$ are simply-connected.
- $S^1$ and SO(3) are multiply-connected:
  - In $S^1$, infinity of homotopy classes.
  - In SO(3), only two homotopy classes.

---

**Classes of Homotopic Free Paths**

---

**Example for Articulated Robot**

**Motion-Planning Framework**

- Continuous representation (configuration space formulation)
- Discretization
- Graph searching
  - (blind, best-first, $A^*$)
Path-Planning Approaches

1. **Roadmap**
   Represent the connectivity of the free space by a network of 1-D curves.

2. **Cell decomposition**
   Decompose the free space into simple cells and represent the connectivity of the free space by the adjacency graph of these cells.

3. **Potential field**
   Define a function over the free space that has a global minimum at the goal configuration and follow its steepest descent.

Roadmap Methods

- **Visibility graph**
  Introduced in the Shakey project at SRI in the late 60s. Can produce shortest paths in 2-D configuration spaces.

- **Voronoi diagram**
  Introduced by Computational Geometry researchers. Generate paths that maximize clearance. Applicable mostly to 2-D configuration spaces.

Roadmaps // Retraction

Roadmap methods are also known as **retraction** methods. This is based on the core mathematical relation (usually unstated) that roadmaps are based on a retraction mapping, or projection:

\[ f: \mathbb{R}^n :\rightarrow \mathbb{R} \]

- Each point in \( \mathbb{R}^n \) maps into some point in the roadmap.
- Each point on the roadmap maps onto itself.
- The mapping is smooth almost everywhere

\[ d(f(a), f(b)) \leq k \cdot d(a, b) \]
**Roadmap Methods**

- Visibility graph
- Voronoi diagram
- Silhouette

  First complete general method that applies to spaces of any dimension and is singly exponential in # of dimensions [Canny, 87]

- Probabilistic roadmaps

---

**Path-Planning Approaches**

1. **Roadmap**
   - Represent the connectivity of the free space by a network of 1-D curves

2. **Cell decomposition**
   - Decompose the free space into simple cells and represent the connectivity of the free space by the adjacency graph of these cells

3. **Potential field**
   - Define a function over the free space that has a global minimum at the goal configuration and follow its steepest descent

---

**Cell-Decomposition Methods**

Two families of methods:

- **Exact cell decomposition**
  - The free space $F$ is represented by a collection of non-overlapping cells whose union is exactly $F$
  - Examples: trapezoidal and cylindrical decompositions

---

**Trapezoidal decomposition**
**Cell-Decomposition Methods**

Two families of methods:
- Exact cell decomposition
- Approximate cell decomposition

F is represented by a collection of non-overlapping cells whose union is contained in F

Examples: quadtree, octree, 2^n-tree

---

**Octree Decomposition**

---

**Path-Planning Approaches**

1. **Roadmap**
   - Represent the connectivity of the free space by a network of 1-D curves

2. **Cell decomposition**
   - Decompose the free space into simple cells and represent the connectivity of the free space by the adjacency graph of these cells

3. **Potential field**
   - Define a function over the free space that has a global minimum at the goal configuration and follow its steepest descent

---

**Potential Field Methods**

- Approach initially proposed for real-time collision avoidance [Khatib, 86]. Hundreds of papers published on it.

Path planning:
- Regular grid G is placed over C-space
- G is searched using a best-first algorithm with potential field as the heuristic function
Potential Field Methods

- Approach initially proposed for real-time collision avoidance [Khatib, 86]. Hundreds of papers published on this topic.
- **Potential field**: Scalar function over the free space
- **Ideal field (navigation function)**: Smooth, global minimum at the goal, no local minima, grows to infinity near obstacles
- Force applied to robot: Negated gradient of the potential field. Always move along that force