Configuration Space for Motion Planning

Pradipta Biswas, PhD (Cantab)
Associate Professor
Indian Institute of Science
https://cambum.net/

Contents

Related Concepts

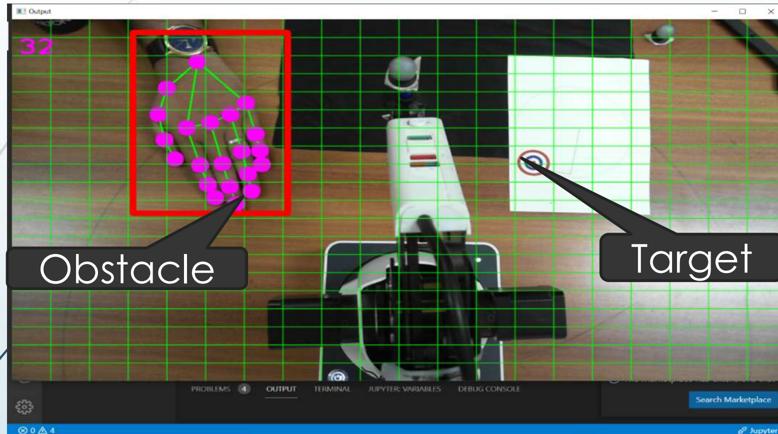
- Set Theory
- Set Operations
- Convex Sets
- Complexity of Algorithms
- Graph Theory – Tree & Graphs

Sources

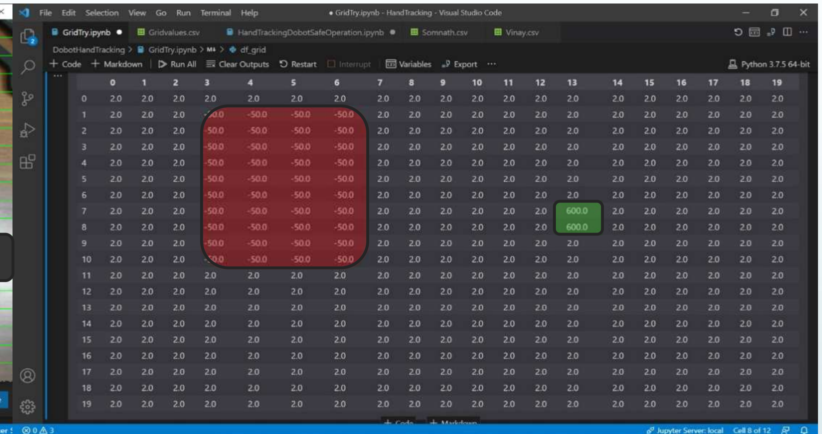
- 2.5. Task Space and Workspace – Modern Robotics (northwestern.edu)
- w3.cs.jmu.edu/spragunr/CS354_F22/readings/planning.pdf
- https://cs.stanford.edu/people/eroberts/courses/soco/projects/1998-99/robotics/definitions.html
- Configuration Space for Motion Planning, RSS Lecture 10, Prof. Seth Teller, MIT
- E190Q – Lecture 14 Autonomous Robot Navigation, Instructor: Chris Clark, Princeton University
- Robotic Motion Planning: Configuration Space Robotics Institute 16-735, Howie Choset, CMU
- A Modern Approach to Artificial Intelligence, Russell & Norvig
- Planning Algorithms, Steven M LaValle

Basic Concept

Proposed State Space Model & MDP

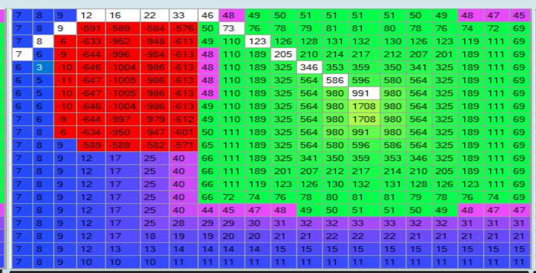


a. Screenshot of actual set up

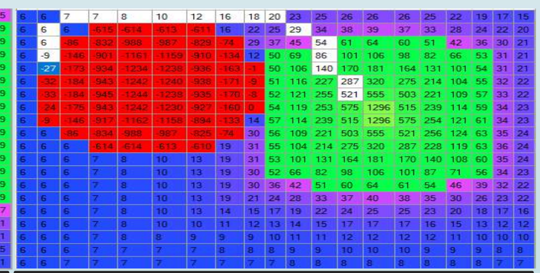


b. State space model with initial reward

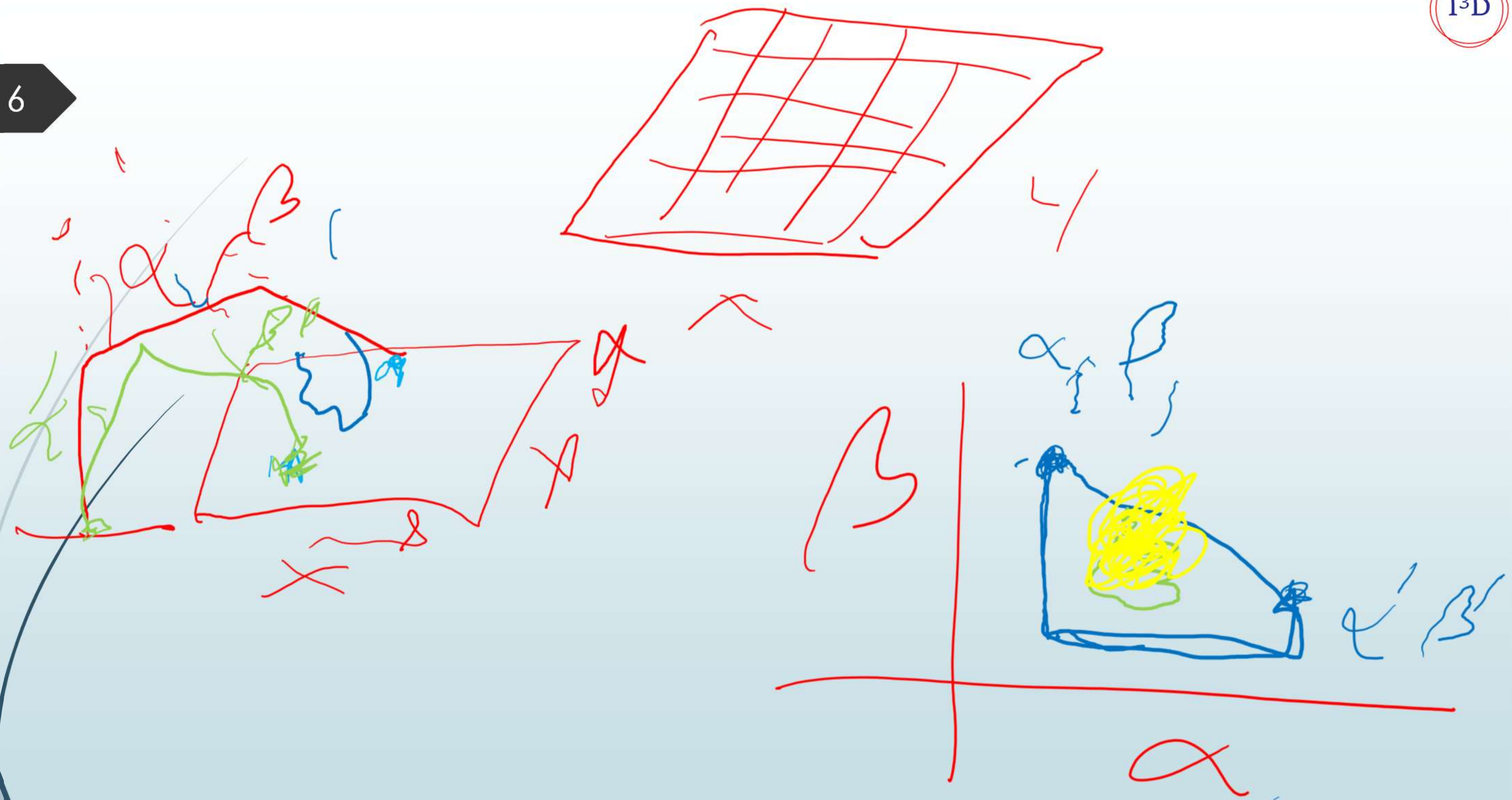
c. Navigation path with no uncertainty in robotic movement



d. Navigation path with 0.05 uncertainty in robotic movement to move on neighbouring cells



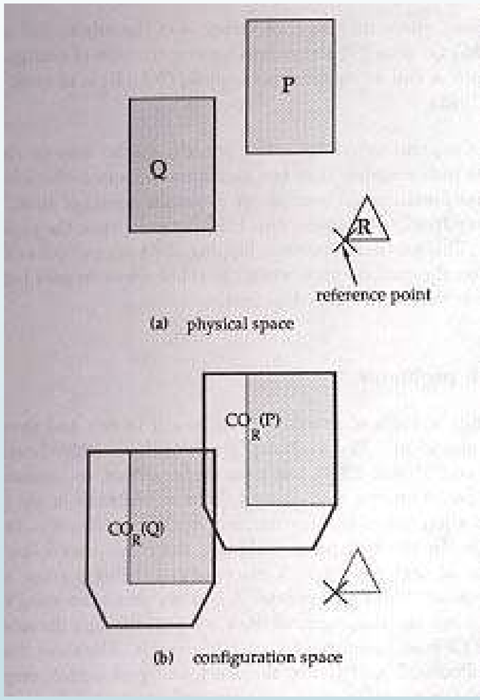
e. Navigation path with 0.3 uncertainty in robotic movement to move on neighbouring cells



Definitions

The Configuration Space

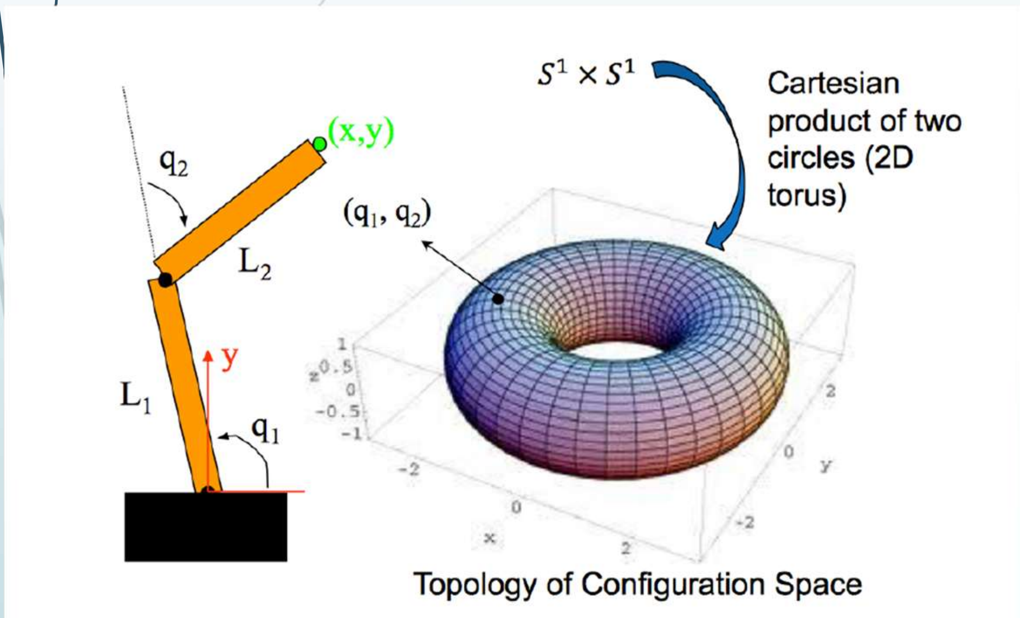
- The configuration space is a transformation from the physical space in which the robot is of finite-size into another space in which the robot is treated as a point. In other words, the configuration space is obtained by shrinking the robot to a point, while growing the obstacles by the size of the robot.
- The figures illustrates the concept of configuration space. P and Q are fixed obstacles in physical space, and R is the robot, whose orientation is fixed.
- Figure b shows the corresponding configuration space.
- The free space of a configuration space simply consists of the areas not occupied by obstacles. Any configuration within this space is called a free configuration.
- The free path between an initial configuration and a goal configuration is the path which lies completely in free space and does not come into contact with any obstacles.



Task Space and Workspace

- The task space is a space in which the robot's task can be naturally expressed.
- For example, if the task is to control the position of the tip of a marker on a board, then task space is the Euclidean plane. If the task is to control the position and orientation of a rigid body, then the task space is the 6-dimensional space of rigid body configurations. One only has to know about the task, not the robot, to define the task space.
- The workspace is a specification of the configurations that the end-effector of the robot can reach, and has nothing to do with a particular task.
- The workspace is often defined in terms of the Cartesian points that can be reached by the end-effector, but it is also possible to include the orientation. The set of positions that can be reached with all possible orientations is sometimes called the dexterous workspace.

Configuration Space & Motion Planning



- To facilitate motion planning, the configuration space was defined as a tool that can be used with planning algorithms.
- A configuration *q* will completely define the state of a robot (e.g. mobile robot (*x, y, θ*)
- The configuration space C, is the space of all possible configurations of the robot.
- The free space $F \subseteq C$ is the portion of the free space which is collision-free.
- The goal of motion planning then, is to find a path in *F* that connects the initial configuration q_{start} to the goal configuration q_{goal}
- For a robot with *k* total motion DOFs, C-space is a coordinate system with *one dimension per DOF*

Examples of Configuration Space

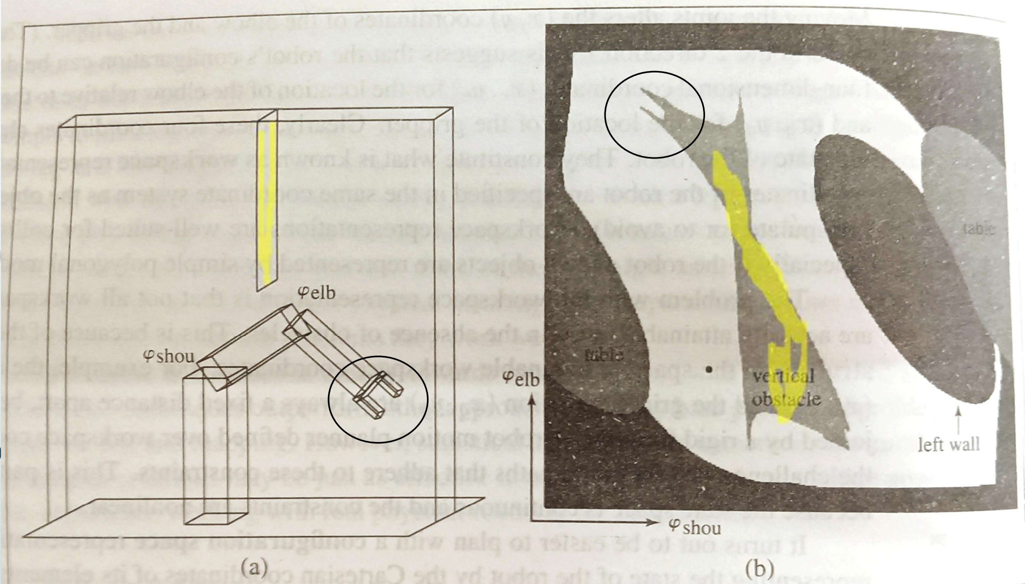


Figure 25.12 (a) Workspace representation of a robot arm with 2 DOFs. The workspace is a box with a flat obstacle hanging from the ceiling. (b) Configuration space of the same robot. Only white regions in the space are configurations that are free of collisions. The dot in this diagram corresponds to the configuration of the robot shown on the left.

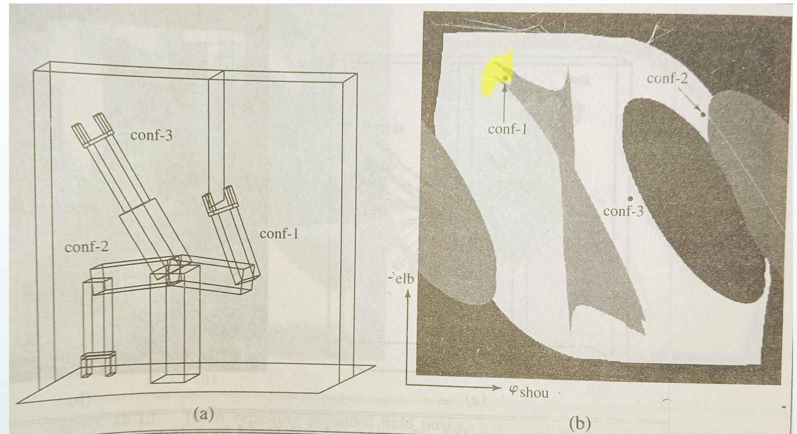


Figure 25.13 Three robot configurations, shown in workspace and configuration space.

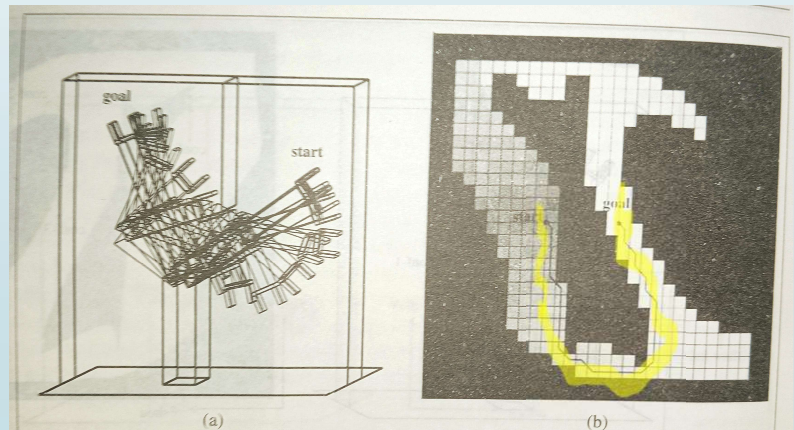
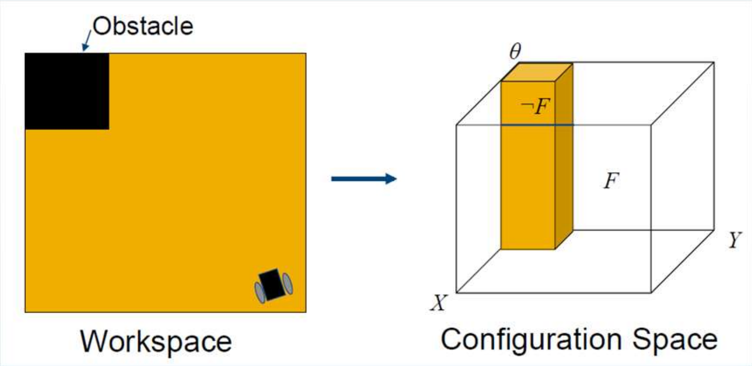


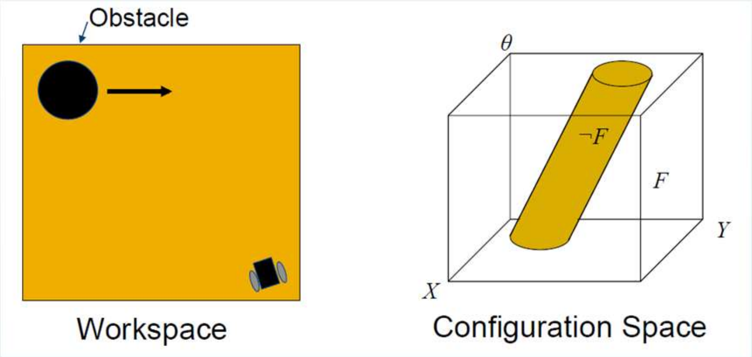
Figure 25.14 (a) Value function and path found for a discrete grid cell approximation of the configuration space. (b) The same path visualized in workspace coordinates. Notice how the robot bends its elbow to avoid a collision with the vertical obstacle.

Examples of Configuration Space

Mobile Robot



Mobile Robot with moving obstacle



Motion Planning

How to Plan Motion

- Define space with one dimension per robot motion (or pose) DOF
- Map robot to a point in this space

  C-space = all robot configurations

  C-obstacle = locus of infeasible configurations due to obstacle
- Motion planning is usually done with three steps:

  1. Define C
  2. Discretize C
  3. Search C
- Each planning problem may have a different definition of C.
  - Example 1: Include 3DOF for a mobile robot in static environment - (x,y,θ).
  - Example 2: Include only 2DOF for a mobile robot in static environment - (x,y).
  - Example 3: Include 5DOF for a mobile robot in dynamic environment - (x,y,θ,v,t).

Obstacles in C-Space

- Let q denote a point in a configuration space Q
- The path planning problem is to find a mapping c:[0,1]→ Q s.t. no configuration along the path intersects an obstacle
- Recall a workspace obstacle is WO_i
- A *configuration space obstacle* QO_i is the set of configurations q at which the robot intersects WO_i, *that is*
  - $\mathcal{QO}_i = \{q \in \mathcal{Q} \mid R(q) \bigcap \mathcal{WO}_i \neq \emptyset\}.$
- The *free configuration space* (or just *free space*) $\mathcal{Q}_{\text{free}}$ *is*

$$\mathcal{Q}_{\text{free}} = \mathcal{Q} \setminus \left(\bigcup \mathcal{QO}_i \right).$$

The free space is generally an open set

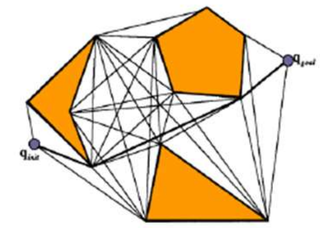
A *free path* is a mapping c:[0,1]→ Q_{free}

A *semifree path* is a mapping c:[0,1]→ cl(Q_{free})

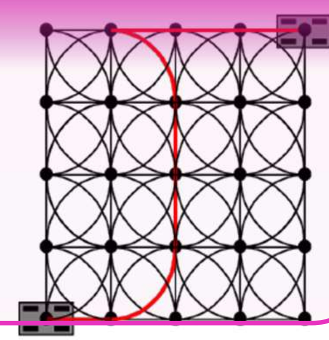
Will be discussed in details in a later lecture

Motion Planning Approaches

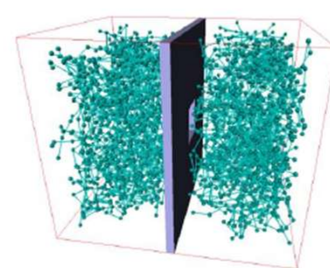
- **Exact algorithms** in continuous space
  - Either find a solution or prove none exist
  - Very computationally expensive
  - Unsuitable for high-dimensional spaces



- **Search-based Planning**
  - Discretize the configuration space into a graph
  - Solve the SP problem via a LC algorithm
  - Computationally expensive in high-dim spaces
  - Resolution completeness and suboptimality guarantees



- **Sampling-based Planning**
  - Sample the configuration space to construct a graph incrementally and construct a path from the samples
  - Efficient in high-dim spaces but problems with "narrow passages"
  - Weak completeness and optimality guarantees

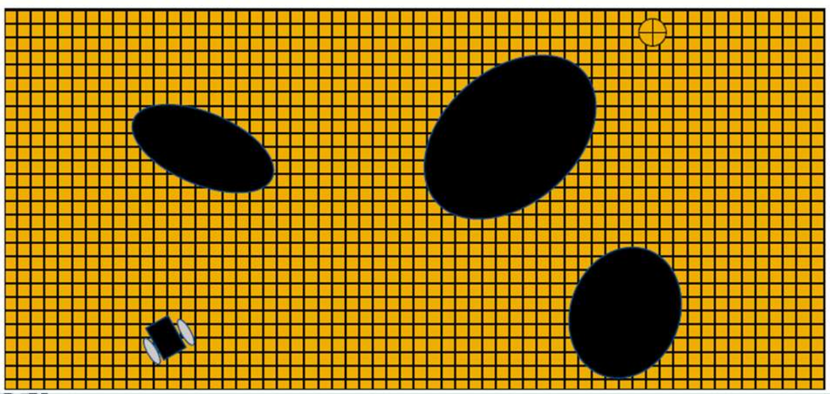


Path Planning

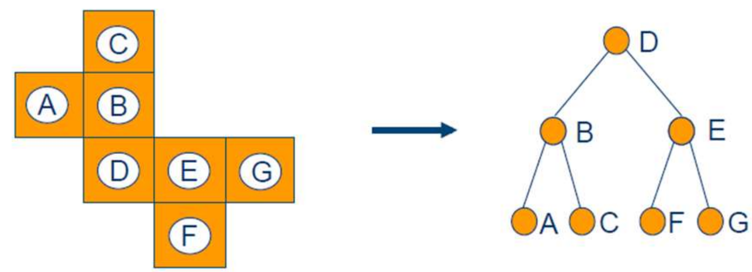
Easiest Approach !!

Cell decomposition

- Decompose the free space into simple cells and represent the connectivity of the free space by the adjacency graph of these cells



- Given a discretization of C, a search can be carried out using a Graph Search or gradien descent, etc.
  - Example: Find a path from D to G



Example of Grid Based Search

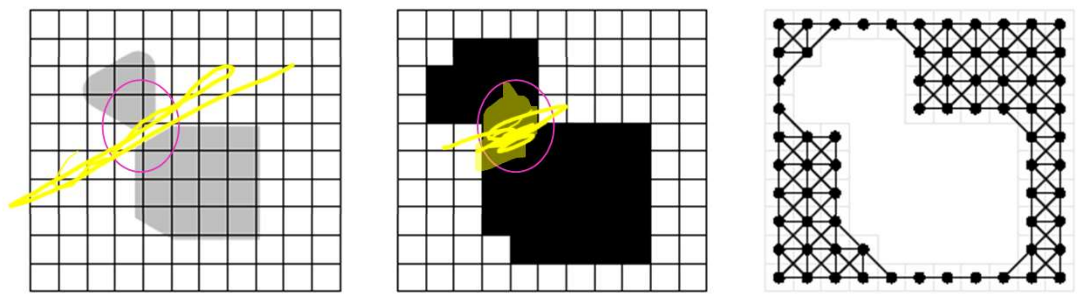
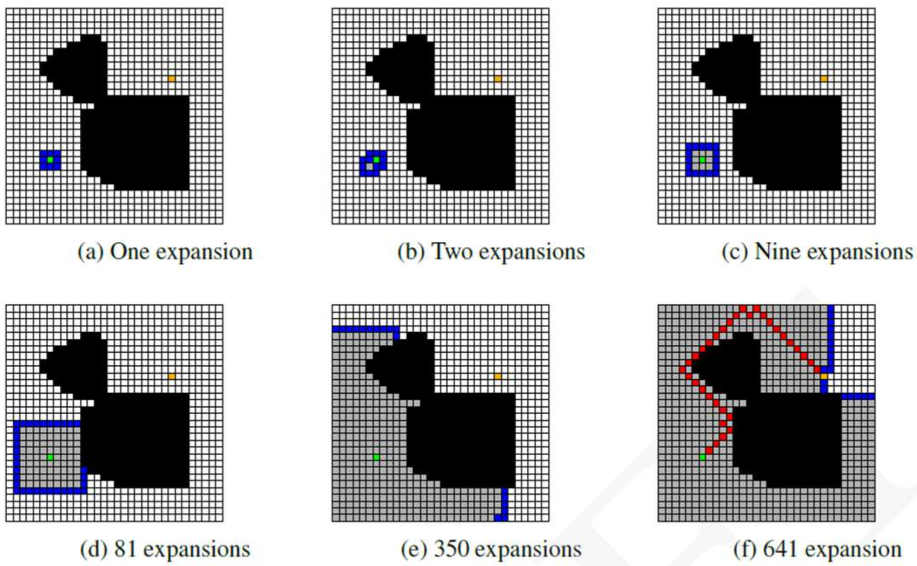


Figure 4.6: Discretizing the configuration space. The figure on the left shows a uniform grid overlaying the configuration space. In the center figure, all grid cells that overlap with $\mathcal{C}_{obs}$ have been marked as inaccessible. The figure on the right shows the state space graph that results if all accessible cells are connected to their neighbors.

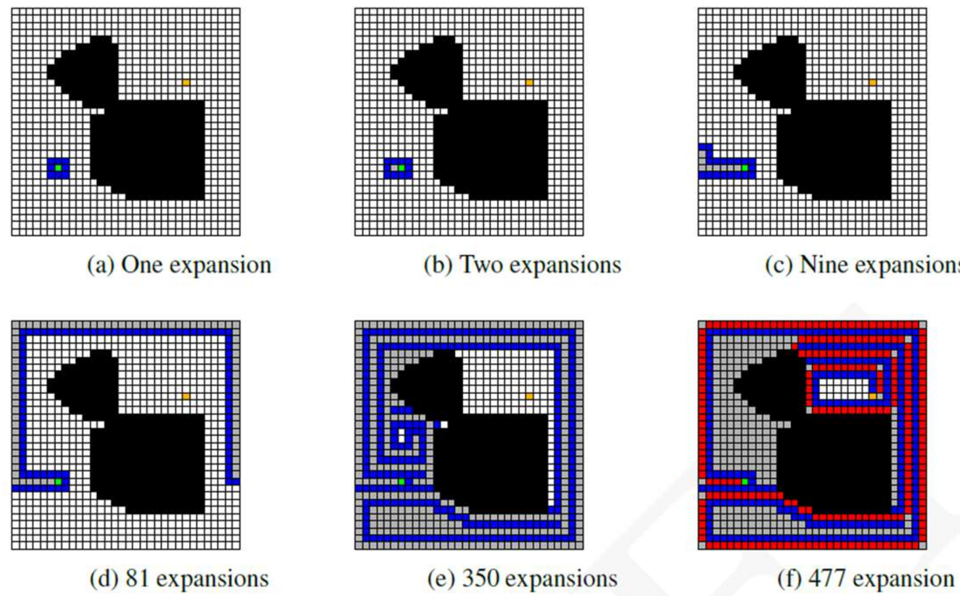
Output of Search Algorithms



(a) One expansion (b) Two expansions (c) Nine expansions

(d) 81 expansions (e) 350 expansions (f) 641 expansion

Figure 4.7: BFS search example. The cell containing the start state is colored green and the goal state is colored gold. States in the frontier are shown in blue. States in the closed set are shown in gray. The final path is shown in red.



(a) One expansion (b) Two expansions (c) Nine expansions

(d) 81 expansions (e) 350 expansions (f) 477 expansion

Figure 4.8: DFS search example.

Output of Search Algorithms

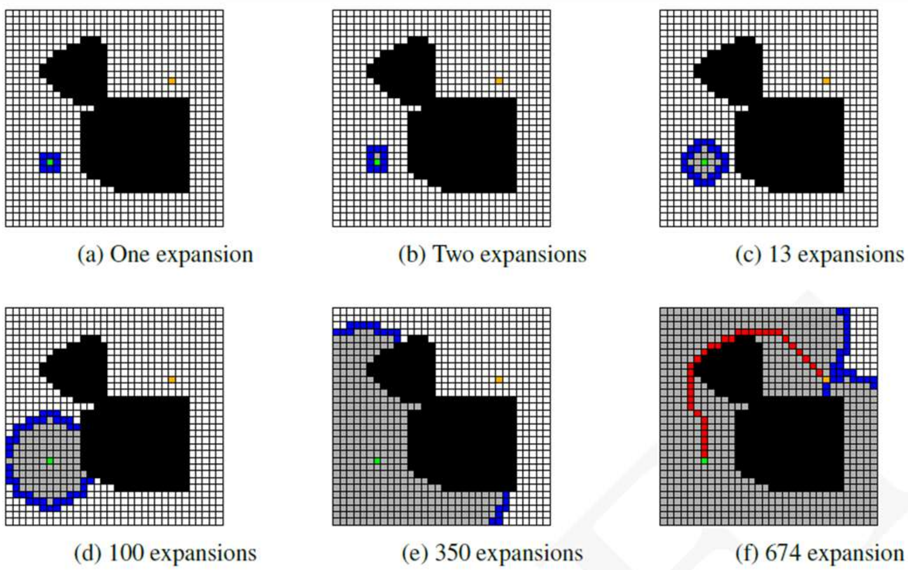


Figure 4.9: Dijkstra's algorithm search example

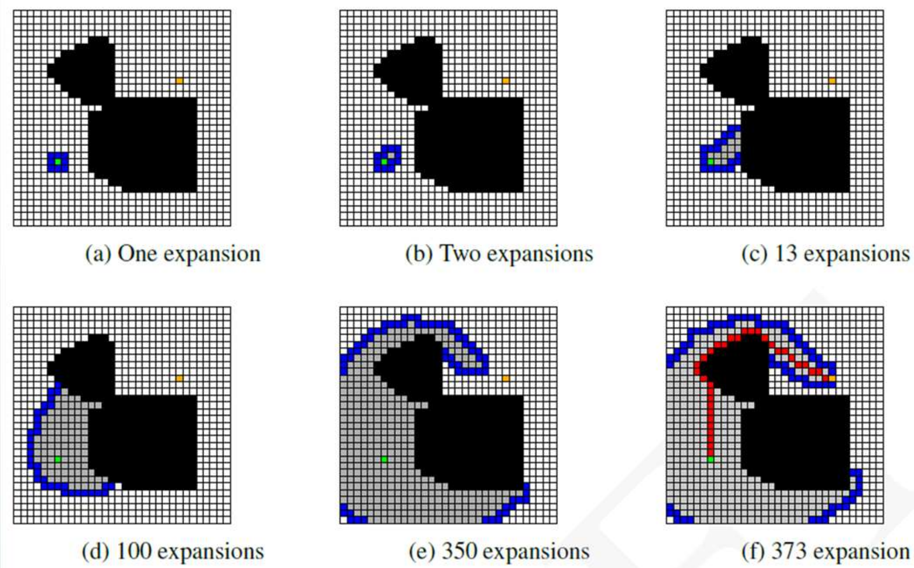


Figure 4.10: A* search example.

Shortest Path ??

In many path-planning problems there is a notion of path cost that is distinct from the number of edges in the path. In the case of 2D navigation, the cost of a path might be the total distance traveled, the time required to reach the goal, or the amount of energy expended by the robot.

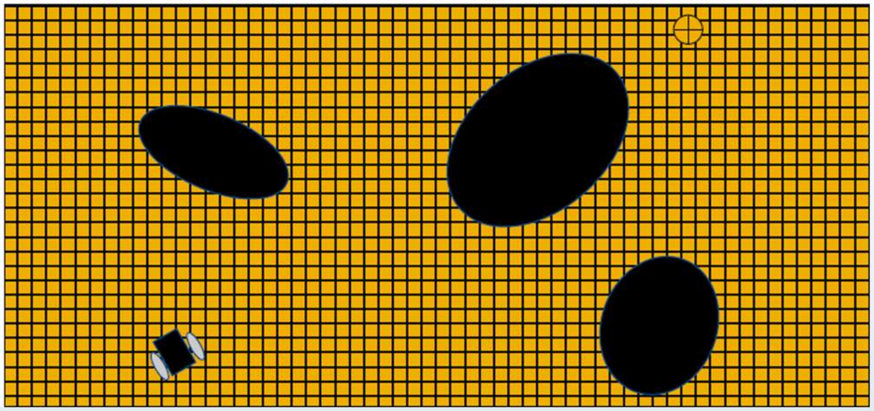
For example, if we want to find shortest paths in the grid navigation problem from Figure 4.6 we should assign weights to the edges in proportion to the distance that the robot must travel to move from cell to cell. This means that diagonal steps must have a higher weight to reflect the fact that the robot moves farther. An optimal solution to this weighted version of the problem will represent the shortest path in terms of distance traveled. Notice that the path discovered by BFS in Figure 4.7 is optimal in terms of the number of steps taken, but it is clearly not optimal in terms of distance.

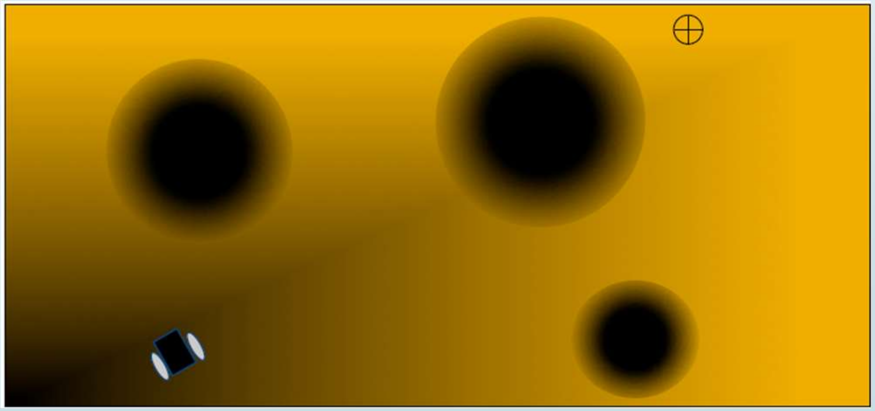
Problems with Cell Decomposition

- Mixed Cells
  - Cells occupy both free space and occupied space
  - May affect completeness if the only path goes through obstacles
- Progressive Cell Decomposition
  - 8 cells allow 360° / 8 = 45 ° of motion
  - 16 cells allow 360° / 16 = 22.5 ° of motion
- **Can we adapt the cell size or number ?**
- Efficiency
  - C Space has 1 dimension for each DoF
  - Grid based searches are inefficient in high DoF
- **Reduce Dimension of C-Space - Skeletonization**
- Obstacles
  - How to represent obstacles in C Space

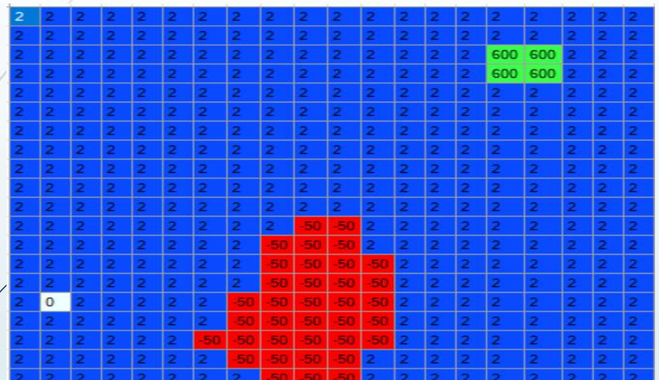
Potential Field

- Aims to minimize path length but maximize clearance from obstacles
- Inflate obstacles to ensure clearance
- Define a function over the free space that has a
  - Value grows with the distance to closest obstacle
  - global minimum at the goal configuration and
  - follow its steepest descent
- Trade off between clearance and path length
- Weighted cost function combining potential field and path length to goal

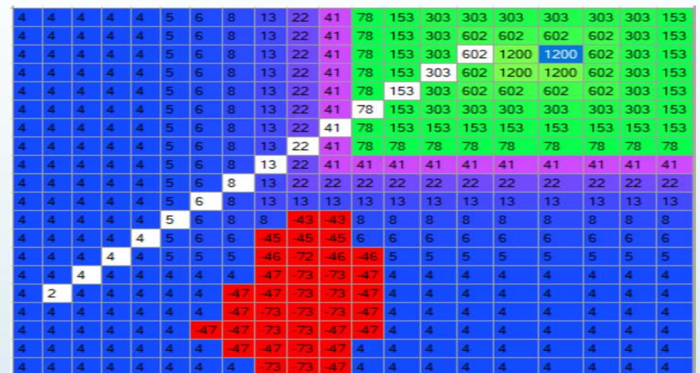




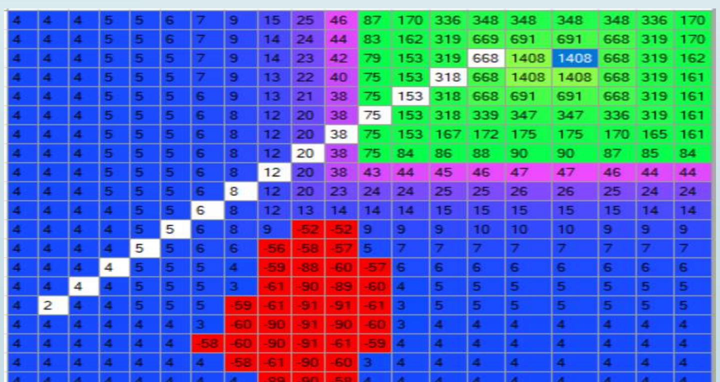
Similar Concept discussed in MDP lecture



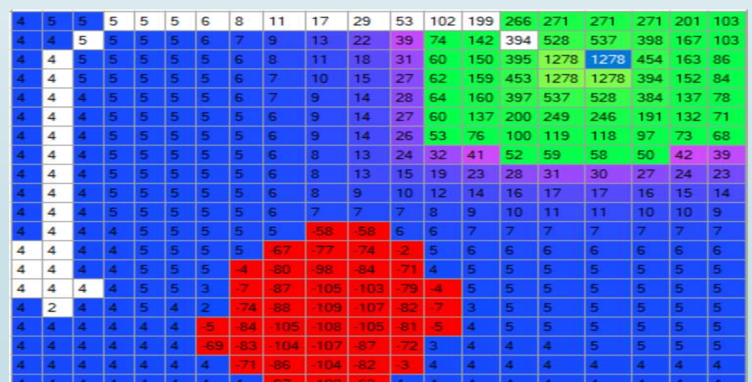
a. Source, target and hand region



b. Navigation path with no uncertainty in robotic movement



c. Navigation path with 0.05 uncertainty in robotic movement to move on neighbouring cells



d. Navigation path with 0.3 uncertainty in robotic movement to move on neighbouring cells

Skeletonization

- Converting the Motion Planning Problem to a graph search problem in lower dimension than the original configuration space
- Ensuring separation from obstacles in configuration space
- Drawing a Vornoi Graph of points equidistant from all obstacles
- May not be the optimal path but will maintain separation from obstacles

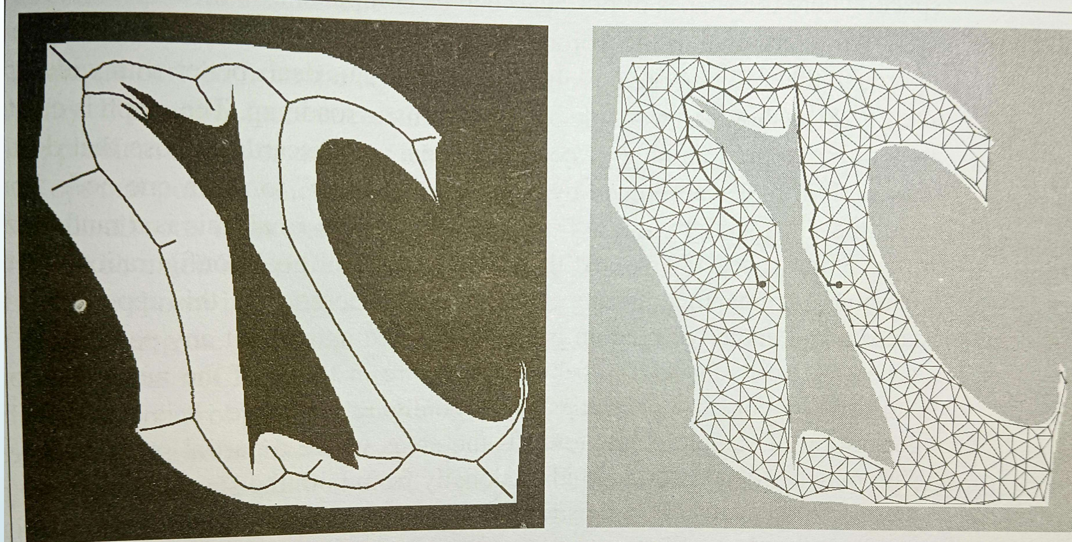
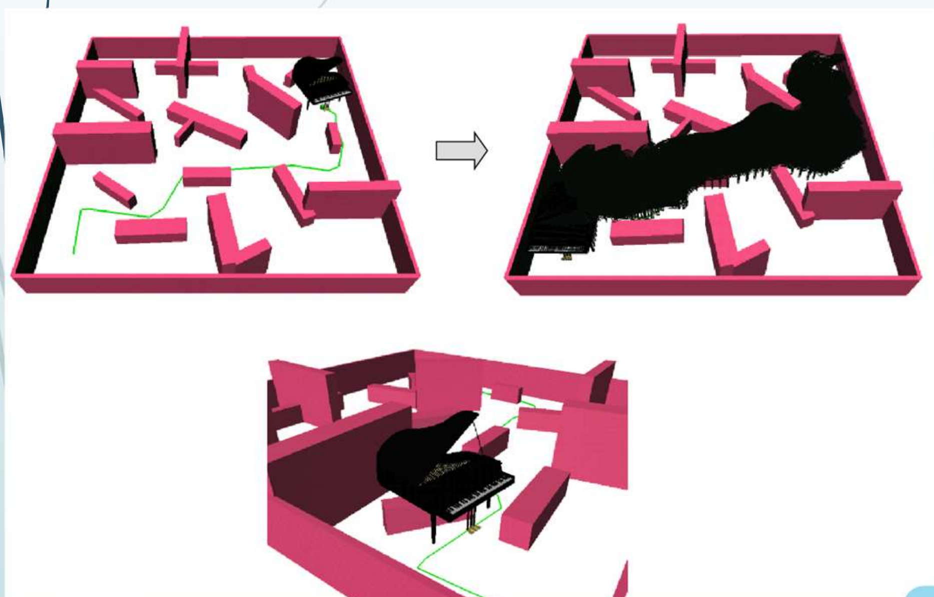


Figure 25.18 (a) The Voronoi graph is the set of points equidistant to two or more obstacles in configuration space. (b) A probabilistic roadmap, composed of 400 randomly chosen points in free space.

Formulating The Motion Planning Problem



Piano Movers Problem

1. A *world* $\mathcal{W}$ in which either $\mathcal{W} = \mathbb{R}^2$ or $\mathcal{W} = \mathbb{R}^3$.

2. A semi-algebraic *obstacle region* $\mathcal{O} \subset \mathcal{W}$ in the world.

3. A semi-algebraic *robot* is defined in $\mathcal{W}$. It may be a rigid robot $\mathcal{A}$ or a collection of m links, $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m$.

4. The *configuration space* $\mathcal{C}$ determined by specifying the set of all possible transformations that may be applied to the robot. From this, $\mathcal{C}_{obs}$ and $\mathcal{C}_{free}$ are derived.

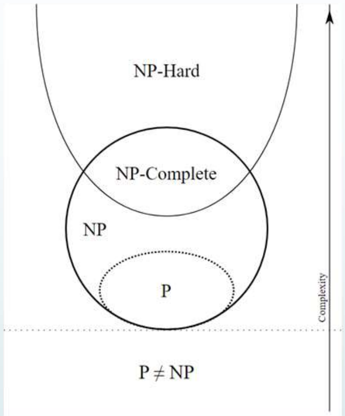
5. A configuration, $q_I \in \mathcal{C}_{free}$ designated as the *initial configuration*.

6. A configuration $q_G \in \mathcal{C}_{free}$ designated as the *goal configuration*. The initial and goal configurations together are often called a *query pair* (or *query*) and designated as (q_I, q_G).

7. A complete algorithm must compute a (continuous) *path*, $\tau : [0, 1] \rightarrow \mathcal{C}_{free}$, such that $\tau(0) = q_I$ and $\tau(1) = q_G$, or correctly report that such a path does not exist.

It was shown by Reif [817] that this problem is PSPACE-hard, which implies NP-hard. The main problem is that the dimension of $\mathcal{C}$ is unbounded.

Efficiency vs Optimality vs Completeness



- What is a NP-Hard Problem ?
- How we solve a NP-Hard Problem ?
  - Relax optimality ?
- Optimal with respect to
  - State Space – minimum number of cells visited
  - Cartesian Workspace – shortest path
  - Physical parameters – number of steep turn, angular movement, power / fuel usage and so on
- Trade off between completeness, efficiency and minimum distance from obstacle
  - Recall – Cobot standards in terms of minimum clearance from human operators

Take Away Points

- Introduction to the concept of Configuration Space
- Terminologies – Configuration Space, Workspace, Task Space
- Motion Planning in C Space
  - Main Challenges
  - Grid based Search
- Problems with Grid based Search
  - Mixed Cells
  - Cell Decomposition
  - Potential Field
- Trade offs in terms of efficiency, completeness and optimality