Robot Motion Planning

slides by Jan Faigl

Department of Computer Science and Engineering
Faculty of Electrical Engineering, Czech Technical University in Prague

lecture

A4M36PAH - Planning and Games

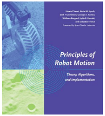
Part I

Motion Planning

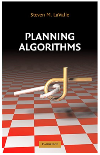
Literature

- Robot Motion Planning, *Jean-Claude Latombe,* Kluwer Academic Publishers, Boston, MA, 1991.

- Principles of Robot Motion: Theory, Algorithms, and Implementations, *H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki and S. Thrun,* MIT Press, Boston, 2005.

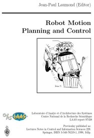


`http://www.cs.cmu.edu/~biorobotics/book`



- Planning Algorithms, *Steven M. LaValle,* Cambridge University Press, May 29, 2006.

`http://planning.cs.uiuc.edu`



- Robot Motion Planning and Control, *Jean-Paul Laumond,* Lectures Notes in Control and Information Sciences, 2009.

`http://homepages.laas.fr/jpl/book.html`

Robot Motion Planning

Motivational problem:

- How to transform high-level task specification (provided by humans) into a low-level description suitable for controlling the actuators?

To develop algorithms for such a transformation.

The motion planning algorithms provide transformations how to move a robot (object) considering all operational constraints.

It encompasses several disciples, e.g., mathematics, robotics, computer science, control theory, artificial intelligence, computational geometry, etc.

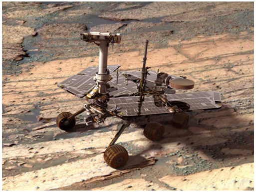
Robot Motion Planning

Motivational problem:

- How to transform high-level task specification (provided by humans) into a low-level description suitable for controlling the actuators?

To develop algorithms for such a transformation.

The motion planning algorithms provide transformations how to move a robot (object) considering all operational constraints.

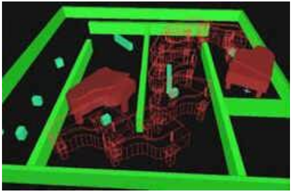


It encompasses several disciples, e.g., mathematics, robotics, computer science, control theory, artificial intelligence, computational geometry, etc.

Piano Mover's Problem

A classical motion planning problem

Having a CAD model of the piano, model of the environment, the problem is how to move the piano from one place to another without hitting anything.


Basic motion planning algorithms are focused primarily on rotations and translations.

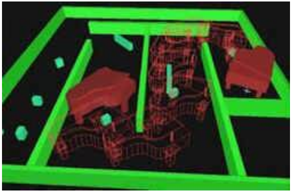
- We need notion of model representations and formal definition of the problem.
- Moreover, we also need a context about the problem and realistic assumptions.

The plans have to be admissible and feasible.

Piano Mover's Problem

A classical motion planning problem

Having a CAD model of the piano, model of the environment, the problem is how to move the piano from one place to another without hitting anything.

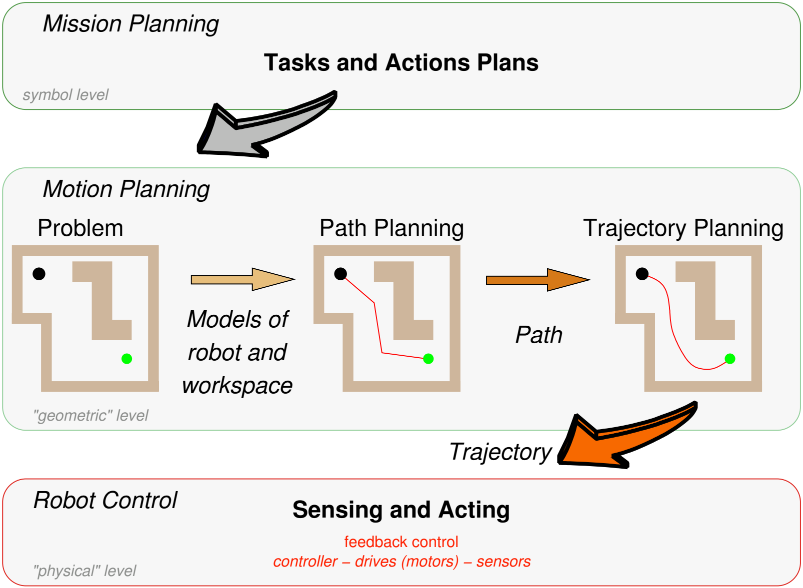


Basic motion planning algorithms are focused primarily on rotations and translations.

- We need notion of model representations and formal definition of the problem.
- Moreover, we also need a context about the problem and realistic assumptions.

The plans have to be admissible and feasible.

Robotic Planning Context



Robotic Planning Context

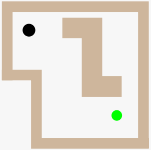
Mission Planning

Tasks and Actions Plans

symbol level

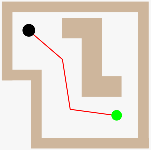
Motion Planning

Problem



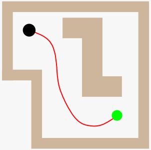
Models of robot and workspace

Path Planning



Path

Trajectory Planning



"geometric" level

Trajectory

Open–loop control?

Robot Control

Sensing and Acting

feedback control

controller – drives (motors) – sensors

"physical" level

Robotic Planning Context

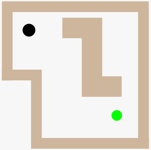
Mission Planning

Tasks and Actions Plans

symbol level

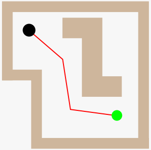
Motion Planning

Problem



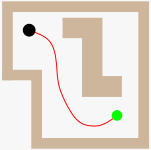
Models of robot and workspace

Path Planning



Path

Trajectory Planning



"geometric" level

Trajectory

Open–loop control?

Robot Control

Sensing and Acting

feedback control

controller – drives (motors) – sensors

Sources of uncertainties because of real environment

"physical" level

Robotic Planning Context

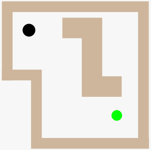
Mission Planning

Tasks and Actions Plans

symbol level

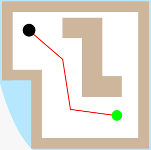
Motion Planning

Problem



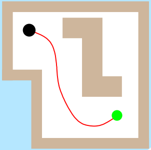
Models of robot and workspace

Path Planning



Path

Trajectory Planning



"geometric" level

Trajectory

Open–loop control?

Robot Control

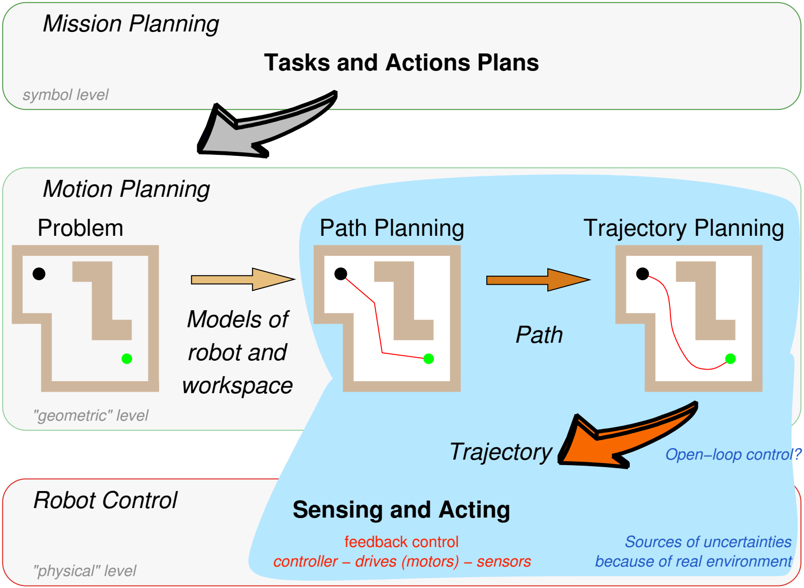
Sensing and Acting

feedback control
controller – drives (motors) – sensors

Sources of uncertainties because of real environment

"physical" level

Robotic Planning Context



Real Mobile Robots

In a real deployment, the problem is a more complex.

- The world is changing
- Robots update the knowledge about the environment

  localization, mapping and navigation
- New decisions have to made
- A feedback from the environment

  Motion planning is a part of the mission replanning loop.

Josef Štrunc, Bachelor thesis, CTU, 2009.

Notation

- $\mathcal{W}$ – **World model** describes the robot workspace and its boundary determines the obstacles $\mathcal{O}_i$.

  2D world, $\mathcal{W} = \mathbb{R}^2$

- A **Robot** is defined by its geometry, parameters (kinematics) and it is controllable by the motion plan.
- $\mathcal{C}$ – **Configuration space** ($\mathcal{C}$-space)

  A concept to describe possible configurations of the robot. The robot's configuration completely specify the robot location in $\mathcal{W}$ including specification of all degrees of freedom.

  E.g., a robot with rigid body in a plane $\mathcal{C} = \{x, y, \varphi\} = \mathbb{R}^2 \times S^1$.

  - Let $\mathcal{A}$ be a subset of $\mathcal{W}$ occupied by the robot, $\mathcal{A} = \mathcal{A}(q)$.
  - A subset of $\mathcal{C}$ occupied by obstacles is

    $$\mathcal{C}_{obs} = \{q \in \mathcal{C} : \mathcal{A}(q) \cap \mathcal{O}_i, \forall i\}$$

  - Collision-free configurations are

    $$\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs}.$$

Notation

- $\mathcal{W}$ – **World model** describes the robot workspace and its boundary determines the obstacles $\mathcal{O}_i$.

  2D world, $\mathcal{W} = \mathbb{R}^2$

- A **Robot** is defined by its geometry, parameters (kinematics) and it is controllable by the motion plan.
- $\mathcal{C}$ – **Configuration space** ($\mathcal{C}$-space)

  A concept to describe possible configurations of the robot. The robot's configuration completely specify the robot location in $\mathcal{W}$ including specification of all degrees of freedom.

  E.g., a robot with rigid body in a plane $\mathcal{C} = \{x, y, \varphi\} = \mathbb{R}^2 \times S^1$.

  - Let $\mathcal{A}$ be a subset of $\mathcal{W}$ occupied by the robot, $\mathcal{A} = \mathcal{A}(q)$.
  - A subset of $\mathcal{C}$ occupied by obstacles is

    $$\mathcal{C}_{obs} = \{q \in \mathcal{C} : \mathcal{A}(q) \cap \mathcal{O}_i, \forall i\}$$

  - Collision-free configurations are

    $$\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs}.$$

Notation

- $\mathcal{W}$ – **World model** describes the robot workspace and its boundary determines the obstacles $\mathcal{O}_i$.

  2D world, $\mathcal{W} = \mathbb{R}^2$

- A **Robot** is defined by its geometry, parameters (kinematics) and it is controllable by the motion plan.
- $\mathcal{C}$ – **Configuration space** ($\mathcal{C}$-space)
  A concept to describe possible configurations of the robot. The robot's configuration completely specify the robot location in $\mathcal{W}$ including specification of all degrees of freedom.

  E.g., a robot with rigid body in a plane $\mathcal{C} = \{x, y, \varphi\} = \mathbb{R}^2 \times S^1$.

  - Let $\mathcal{A}$ be a subset of $\mathcal{W}$ occupied by the robot, $\mathcal{A} = \mathcal{A}(q)$.
  - A subset of $\mathcal{C}$ occupied by obstacles is

    $$\mathcal{C}_{obs} = \{q \in \mathcal{C} : \mathcal{A}(q) \cap \mathcal{O}_i, \forall i\}$$

  - Collision-free configurations are

    $$\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs}.$$

Notation

- $\mathcal{W}$ – **World model** describes the robot workspace and its boundary determines the obstacles $\mathcal{O}_i$.

  2D world, $\mathcal{W} = \mathbb{R}^2$

- A **Robot** is defined by its geometry, parameters (kinematics) and it is controllable by the motion plan.

- $\mathcal{C}$ – **Configuration space** ($\mathcal{C}$-space)

  A concept to describe possible configurations of the robot. The robot's configuration completely specify the robot location in $\mathcal{W}$ including specification of all degrees of freedom.

  E.g., a robot with rigid body in a plane $\mathcal{C} = \{x, y, \varphi\} = \mathbb{R}^2 \times S^1$.

  - Let $\mathcal{A}$ be a subset of $\mathcal{W}$ occupied by the robot, $\mathcal{A} = \mathcal{A}(q)$.
  - A subset of $\mathcal{C}$ occupied by obstacles is

    $$\mathcal{C}_{obs} = \{q \in \mathcal{C} : \mathcal{A}(q) \cap \mathcal{O}_i, \forall i\}$$

  - Collision-free configurations are

    $$\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs}.$$

Notation

- $\mathcal{W}$ – **World model** describes the robot workspace and its boundary determines the obstacles $\mathcal{O}_i$.

  2D world, $\mathcal{W} = \mathbb{R}^2$

- A **Robot** is defined by its geometry, parameters (kinematics) and it is controllable by the motion plan.

- $\mathcal{C}$ – **Configuration space** ($\mathcal{C}$-space)
  A concept to describe possible configurations of the robot. The robot's configuration completely specify the robot location in $\mathcal{W}$ including specification of all degrees of freedom.

  E.g., a robot with rigid body in a plane $\mathcal{C} = \{x, y, \varphi\} = \mathbb{R}^2 \times S^1$.

  - Let $\mathcal{A}$ be a subset of $\mathcal{W}$ occupied by the robot, $\mathcal{A} = \mathcal{A}(q)$.
  - A subset of $\mathcal{C}$ occupied by obstacles is

    $$\mathcal{C}_{obs} = \{q \in \mathcal{C} : \mathcal{A}(q) \cap \mathcal{O}_i, \forall i\}$$

  - Collision-free configurations are

    $$\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs}.$$

Path / Motion Planning Problem

- **Path** is a continuous mapping in $\mathcal{C}$-space such that

$$\pi : [0, 1] \to \mathcal{C}_{free}, \text{ with } \pi(0) = q_0, \text{ and } \pi(1) = q_f,$$

Only geometric considerations

- **Trajectory** is a path with explicate parametrization of time, e.g., accompanied by a description of the motion laws ($\gamma : [0, 1] \to \mathcal{U}$, where $\mathcal{U}$ is robot's action space).

It includes dynamics.

$$[T_0, T_f] \ni t \rightsquigarrow \tau \in [0, 1] : q(t) = \pi(\tau) \in \mathcal{C}_{free}$$

The planning problem is determination of the function $\pi(\cdot)$.

Additional requirements can be given:

- Smoothness of the path
- Kinodynamic constraints

E.g., considering friction forces

- Optimality criterion

shortest vs fastest (length vs curvature)

Path / Motion Planning Problem

- **Path** is a continuous mapping in $\mathcal{C}$-space such that

$$\pi : [0,1] \to \mathcal{C}_{free}, \text{ with } \pi(0) = q_0, \text{ and } \pi(1) = q_f,$$

Only geometric considerations

- **Trajectory** is a path with explicate parametrization of time, e.g., accompanied by a description of the motion laws ($\gamma : [0,1] \to \mathcal{U}$, where $\mathcal{U}$ is robot's action space).

It includes dynamics.

$$[T_0, T_f] \ni t \rightsquigarrow \tau \in [0,1] : q(t) = \pi(\tau) \in \mathcal{C}_{free}$$

The planning problem is determination of the function $\pi(\cdot)$.

Additional requirements can be given:

- Smoothness of the path
- Kinodynamic constraints

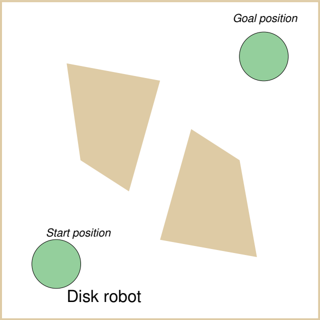
E.g., considering friction forces

- Optimality criterion

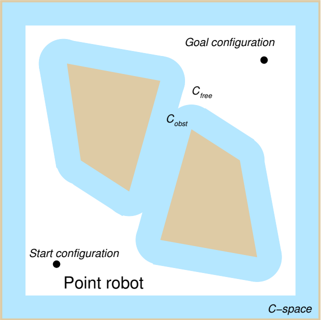
shortest vs fastest (length vs curvature)

Planning in $\mathcal{C}$-space

Robot motion planning robot for a disk robot with a radius ρ.



Motion planning problem in geometrical representation of $\mathcal{W}$

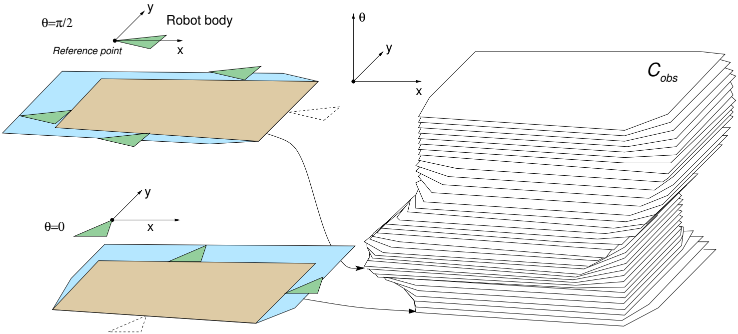


Motion planning problem in $\mathcal{C}$-space representation

$\mathcal{C}$-space has been obtained by enlarging obstacles by the disk $\mathcal{A}$ with the radius ρ.

By applying Minkowski sum: $\mathcal{O} \oplus \mathcal{A} = \{x + y \mid x \in \mathcal{O}, y \in \mathcal{A}\}$.

Example of $\mathcal{C}_{obs}$ for a Robot with Rotation



A simple 2D obstacle $\rightarrow$ *has a complicated* $\mathcal{C}_{obs}$

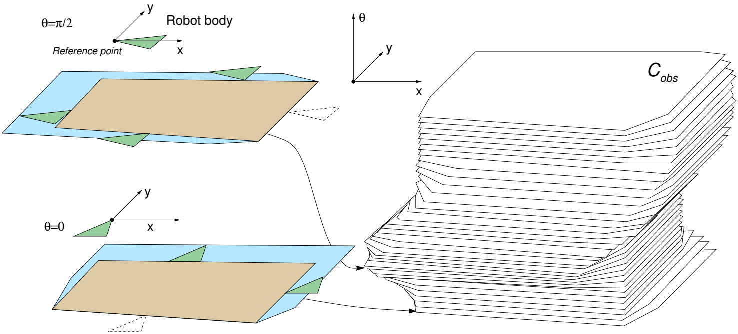
- Deterministic algorithms exist

Requires exponential time in $\mathcal{C}$ dimension,

J. Canny, PAMI, 8(2):200–209, 1986

- Explicit representation of $\mathcal{C}_{free}$ is impractical to compute.

Example of $\mathcal{C}_{obs}$ for a Robot with Rotation



A simple 2D obstacle → has a complicated $\mathcal{C}_{obs}$

- Deterministic algorithms exist

  Requires exponential time in $\mathcal{C}$ dimension,

  J. Canny, PAMI, 8(2):200–209, 1986

- Explicit representation of $\mathcal{C}_{free}$ is impractical to compute.

Holonomic Robots

- **Holonomic** – all degrees of freedom are controllable
- **Non-holonomic** – some degrees of freedom are not directly controllable

Representation of $\mathcal{C}$-space

How to deal with continuous representation of $\mathcal{C}$-space?

Continuous Representation of $\mathcal{C}$-space

↓

Discretization
processing critical geometric events, (random) sampling
roadmaps, cell decomposition, potential field

↓

Graph Search Techniques
BFS, Gradient Search, A*

Representation of $\mathcal{C}$-space

How to deal with continuous representation of $\mathcal{C}$-space?

Continuous Representation of $\mathcal{C}$-space

$\downarrow$

Discretization
processing critical geometric events, (random) sampling
roadmaps, cell decomposition, potential field

$\downarrow$

Graph Search Techniques
BFS, Gradient Search, A*

Planning Methods Overview

(selected approaches)

- Roadmap based methods

  Create a connectivity graph of the free space.

  - Visibility graph
  - Cell decomposition
  - Voronoi diagram
- Potential field methods

Classic path planning algorithms

Randomized path/motion planning approaches

- Probabilistic roadmaps (PRM)
- Expansive-Spaces Tree (EST)
- Rapidly-Exploring Random Tree (RRT)

  Allow to consider kinodynamic constraints.

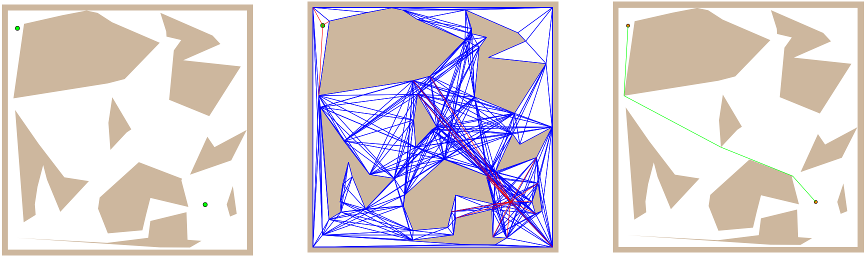
- Optimal sampling based Planner - RRT*

  S. Karaman and E. Frazzoli, IJJR, 30(7):846-894, 2011

Visibility Graph

1. Compute visibility graph
2. Find the shortest path

E.g., by Dijkstra's algorithm



Problem — Visibility graph — Found shortest path

Constructions of the visibility graph:

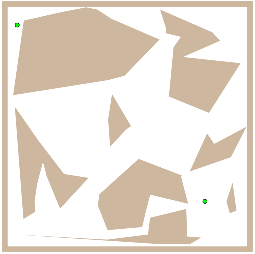
- Naïve – all segments between n vertices of the map $O(n^3)$
- Using rotation trees for a set of segments – $O(n^2)$

M. H. Overmars and E. Welzl, 1988

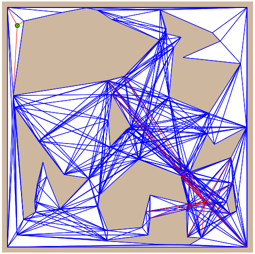
Visibility Graph

1. Compute visibility graph
2. Find the shortest path

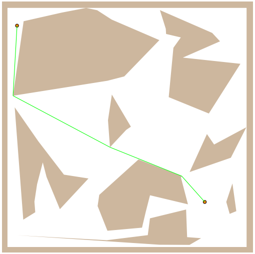
E.g., by Dijkstra's algorithm



Problem



Visibility graph



Found shortest path

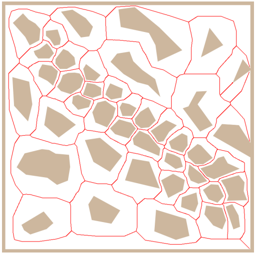
Constructions of the visibility graph:

- Naïve – all segments between n vertices of the map $O(n^3)$
- Using rotation trees for a set of segments – $O(n^2)$

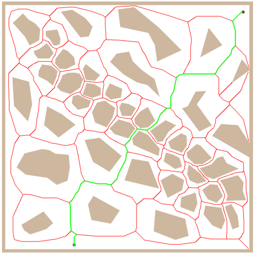
M. H. Overmars and E. Welzl, 1988

Voronoi Diagram

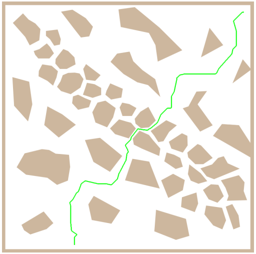
1. Roadmap is Voronoi diagram that maximizes clearance from the obstacles
2. Start and goal positions are connected to the graph
3. Path is found using a graph search algorithm



Voronoi diagram



Path in graph



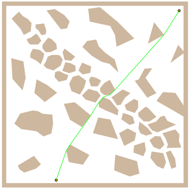
Found path

Visibility Graph vs Voronoi Diagram

Visibility graph

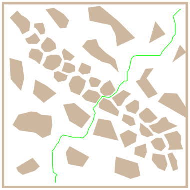
- Shortest path, but it is close to obstacles. We have to consider safety of the path.

  An error in plan execution can lead to a collision.
- Complicated in higher dimensions



Voronoi diagram

- It maximize clearance, which can provide conservative paths
- Small changes in obstacles can lead to large changes in the diagram
- Complicated in higher dimensions



A combination is called Visibility-Voronoi – R. Wein, J. P. van den Berg, D. Halperin, 2004

For higher dimensions we need other roadmaps.

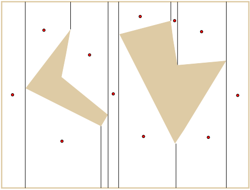
Cell Decomposition

1. Decompose free space into parts.

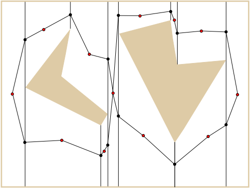
   Any two points in a convex region can be directly connected by a segment.

2. Create an adjacency graph representing the connectivity of the free space.
3. Find a path in the graph.

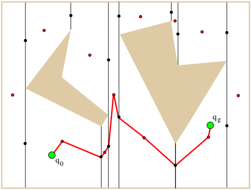
Trapezoidal decomposition



Centroids represent cells



Connect adjacency cells



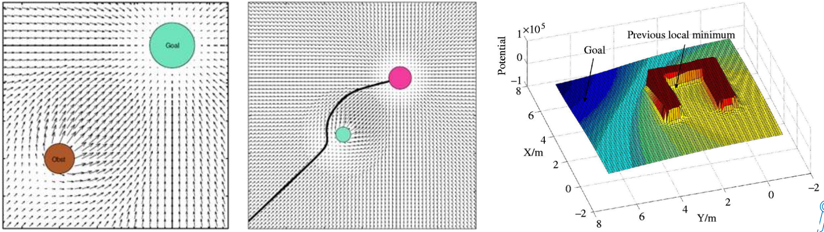
Find path in the adjacency graph

Other decomposition (e.g., triangulation) are possible.

Artificial Potential Field Method

- The idea is to create a function f that will provide a direction towards the goal for any configuration of the robot.
- Such a function is called navigation function and $-\nabla f(q)$ points to the goal.
- Create a potential field that will attract robot towards the goal q_f while obstacles will generate repulsive potential repelling the robot away from the obstacles.

The navigation function is a sum of potentials.



Such a potential function can have several local minima.