Robot Motion Planning

Classic Path Planning Algorithms

A. Narayanan[1]

[1]Department of Informatics

Outline

- Overview of Classic Path Planning Approaches
- Roadmaps
  - Visibility Maps
  - Generalized Voronoi Diagrams
- Cell Decomposition
  - Trapeziodal decomposition
  - Boustrophedon Decomposition
- Potential Field

- **Roadmap**
  Represent the connectivity of the free space by 1-D Curves
- **Cell Decomposition**
  Decompose the free space into simple cells and represent the connectivity of the free space by adjacency graph of these cells
- **Potential Field**
  Define a potential function over the free space that has a global minimum at the goal and follow the steepest descent of the potential function

Roadmaps

- construct a `map` once and then use that map to plan subsequent paths more quickly
- Topological maps aim at representing environments with graphlike structures
- **Roadmaps** are a type of topological map embedded in free space where each node corresponds to a specific location and an edge corresponds to a path between neighboring locations

find path from q_{start} to roadmap $\rightarrow$ traverse roadmap to vicinity of goal $\rightarrow$ find path from roadmap to the q_{goal}

Definition

A union of one-dimensional curves is a roadmap RM if for all q_{start} and q_{goal} in $\mathcal{Q}_{free}$ that can be connected by a path, the following properties hold:

1. **Accessibility**: there exists a path from $q_{start} \in \mathcal{Q}_{free}$ to some $q'_{start} \in RM$,
2. **Departability**: there exists a path from some $q'_{goal} \in RM$ to $q_{goal} \in \mathcal{Q}_{free}$, and
3. **Connectivity**: there exists a path in RM between q'_{start} and q'_{goal} .

Advantages

1. Efficient method of path planning in the configuration space. It moves a large part of processing to offline.
2. Just need to connect q_{init}, q_{goal} to roadmap online.
3. Finding a path is like searching in a roadmap.

Visibility Graph

Assume a polygonal configuration space with obstacles approximated as polygons, with the nodes v_i of the graph consisting of q_{start}, q_{goal} and all obstacle vertices

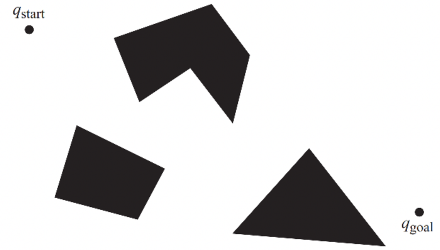

Figure: A polygonal config space with start and goal

Visibility Graph

The graph edges e_{ij} are straight-line segments that connect two line-of-sight nodes v_i and v_j

$$\boxed{\textbf{Complexity: } O(n^2)}$$

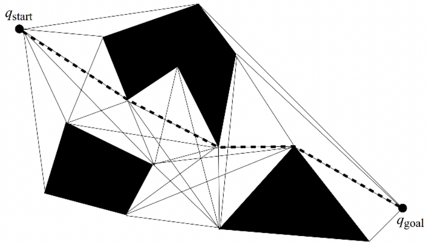

Figure: The Visibility graph

Reduced Visibility Graph

the visibility graph has many needless edges. The use of supporting and separating lines can reduce the number of edges.

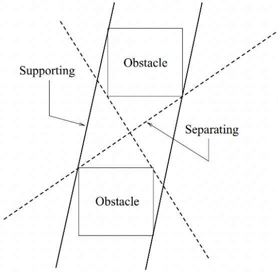

Figure: Supporting and Separating Line Segments

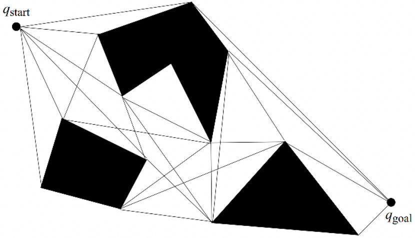

Figure: Reduced Visibility graph

Rotational Plane Sweep Algorithm

Generalized Voronoi Diagrams

The **Generalized Voronoi Diagram** (**GVD**) is the set of points where the distance to the two closest obstacles is the same.

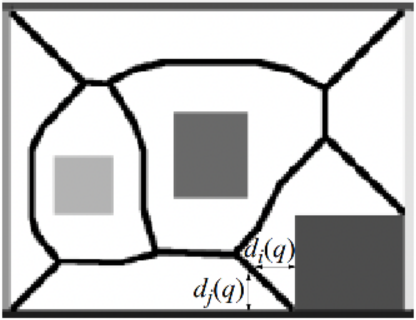

Figure: Voronoi Diagram

$$\boxed{\textbf{Complexity: } O(n\log(n))}$$

Construction of the GVD

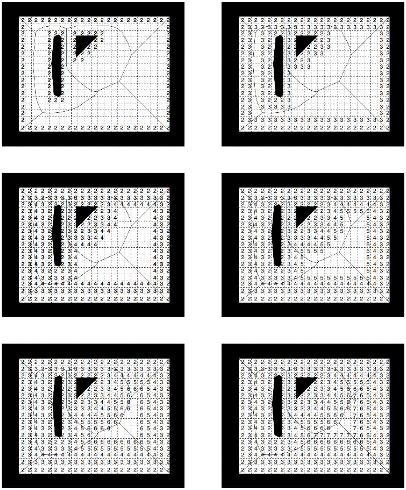

Figure: The **Brushfire algorithm** uses a grid to approximate distance

Cell Decompositions

The main idea is to decompose the free space into simple cells and represent the connectivity of the free space F by the adjacency of these cells. It is called an exact cell decomposition if the union of all cells is exactly F, meaning there is no overlap between cells.

Trapeziodal Decomposition

- The free space is bounded by polygons and C-Space obstacles are poygon shaped
- decompose the space into trapezoidal triangle cells
- connect every neighboring centered point in every neighboring cells that dont intersect with obstacles (**Adjacency graph**)

Problems: there are a lot of useless small cells that could be aggregated to avoid long and less efficient paths

Boustrophedon Decomposition

- Consider the vertices at which a vertical line can be extended both up and down in free space i.e. the critical points
- set boundary lines at critical points
- Then, an exhaustive walk through the critical points is performed in order to obtain a connectivity graph

Potential Field Method

- The C-space is turned into a potential field, where the obstacles are surrounded by a repulsive field
- The goal location by an attractive field. To navigate, the robot applies a force proportional to the negative gradient of the field - this is called gradient descent.
- **Advantage:** potential field methods are easy to compute.
- **Disadvantage:** They can suffer from local minima (where robot gets stuck), and they don't consider dynamic constraints in their initial form (forces can be too high).

The Attractive Potential

- we could use `Conic Potential` - When numerically implementing this method, gradient descent may have "chattering" problems since there is a discontinuity in the attractive gradient at the origin.
- or we could use `Quadratic Potential` - grows without bound as q moves away from q_{goal}. If q_{start} is far from q_{goal}, this may produce a desired velocity that is too large
- Solution - conic potential attracts the robot when it is very distant from q_{goal} and the quadratic potential attracts the robot when it is near q_{goal}.

$$U_{att}(q) = \begin{cases} \frac{1}{2}\zeta d^2(q, q_{goal}) & d(q, q_{goal} \leq d^*_{goal} \\ d^*_{goal}\zeta d(q, q_{goal}) - \frac{1}{2}\zeta(d^*_{goal})^2 & d(q, q_{goal} > d^*_{goal} \end{cases} \tag{1}$$