Combinatorial Motion Planning

a.k.a. Exact motion planning

"Combinatorial" comes from the need to check every possibility.

Combinatorial M.P. alg. are complete. i.e. in finite time the alg. will report a solution or the fact that a solution does not exist.

The alg.s in Ch 5 are not complete! Why?

Because we only approximate \( C_{\text{free}} \) with a topological graph. One never knows if disconnected components of the graph are actually connected. If arcs are tested at discrete points, then arcs could be invalid, thus generating invalid plans.
The algs in Ch6 are based on exact knowledge of C_free. The "narrow passage" problem is not a problem for exact methods!

Virtually every combinatorial method constructs a roadmap — a retraction of C_free to a 1-dimensional set.

First general alg appeared in the early 1980's. Schwartz & Sharir proved that the Piano Movers' problem is intractible. Based on Collins' Decomposition of C_free into cells. But Collins' dec is \(O(2^n)\) i.e. doubly exponential!

(where \(n\) is the dimension of C space).

(for the generalized movers' problem)

The best possible algorithm for exact M.P. is \(O(2^n)\).

(Canny 1986 thesis). This complexity comes from the fact that in the worst case, the # of
connected components of C-space is $O(2^n)$. Implementation of Canny’s alg is so difficult that it has never been done.

Despite high algorithmic complexity, there are still problems where combinatorial/exact method are tractible & the best choice.

e.g. Mobile robot that is small in comparison to geometric features of its environment.

Let $G$ be a topological graph that maps into $C_{\text{free}}$.

Let $S$ be the swath of $G$.

**Definition (LaValle):** $G$ is a roadmap if:

- $G$ is accessible from $C_{\text{free}} \Rightarrow$ trivially easy to connect any $q \in C_{\text{free}}$ to $S$
- $G$ preserves connectivity of $C_{\text{free}} \Rightarrow$ there is a one-to-one and onto correspondence between the components of $G$ and those of $C_{\text{free}}$

**Question:** Is the second bullet subsumed by the first?

Section 6.2 - $C_{\text{free}}$ & $C_{\text{obs}}$ are polygonal
Applicable cases: both are in the plane:

(1) point robot in polygonal world

(2) polygonal robot translating in polygonal world.

Motion planning reduces to planning the path of a point thru a polygonal obstacle field

We need a convenient data structure for representing the polygons and their connectivity.

Polygons may be nonconvex including possibility of having holes.

Vectors follow boundaries such that $C_{obs}$ is always on the left.

We need to keep track of holes and allow possibility of decomposed polygons.

For efficient access to data needed for planning when $A$ vertex
Cos is a collection of polygons

Lavalle recommends:

vertex:
  vertex.location ← x, y coords
  vertex.half_edge ← any half edge

face:
  face.half_edge_inner
  face.half_edge_outer
  face.hole_in_face

half_edge:
  half_edge.origin_vertex
    "twin_edge"
    "internal_face (NIL if obstacle)"
  makes
  list
  doubly-connected

6.2.2 Vertical Cell Decomposition

Decompose $C_{free}$ into regions that are:
1. Connection of any 2 points in cell is "easy."
2. Cell adjacency info easily extracted.
3. "Easy" to tell cell membership of any given point.

If the cell decomp. satisfies the above 3 characteristics, then motion planning reduces to graph search.

Defining the decomposition for \( C_{\text{free}} \subseteq \mathbb{R}^2 \) and \( C_{\text{free}} \) and \( C_{\text{obs}} \) are polygonal.

Sweep a vertical line from left to right
Keep track of data at "critical" events

Critical events

- Vertical line splits into two
- Lower or upper intersection changes edges
- New piece of line emerges.

Example
Now add one vertex of G in each 1-cell & each 2-cell.

Connect vertices to those in adjacent cells.

Note: Figure 6.4 in LaValle has an extra edge and a missing edge. Can you find them?

Query phase:

Place $q_i$ & $q_f$, connect to roadmap search graph extract path.

Complexity
Straightforward alg.

Let \( n \) be the \# of vertices of \( \text{Cobs} \).

Then there are \( O(n) \) critical events.

For each critical event, intersect the vertical line w/ all \( O(n) \) edges of \( \text{Cobs} \).

\[ \Rightarrow O(n^2) \]

Best known alg. is \( O(n \log n) \).

Sort vertices according to their distance along the sweep direction. This operation is \( O(n \log n) \).

Process \( O(n) \) critical points. This requires examining two edges per critical point, so \( O(n) \) work.

Maximum Clearance Roadmaps — Generalized Voronoi Diagrams

Used for inaccurate mobile robots.

Create roadmap that is maximally far from all obstacles.

As above, assume polygonal \( \text{Cobs} \) & \( \text{Cfree} \).

[Diagram of a road network with a medial axis]

a.k.a. a medial axis
Three cases:

- vertex-edge
- edge-edge
- vertex-vertex

The roadmap is formed by connecting these three primitives. Consider a simple case:

Handling endpoints is not clearly defined.

Do we include this curve?
Complexity:

Let $n$ be the number of edges (and vertices) of $G_{obs}$.

$O(n^3)$ - for every feature pair, compute the line or parabola.

$O(n^3)$ - intersect every pair of line-line or line-parabola pairs to piece together the roadmap.

$\therefore$ Straightforward approach is $O(n^4)$ where $n$ is the number of geometric features.

Best known alg is $O(k \log k)$, where $k$ is the # of curve segments in the roadmap. But $k = O(n^2)$,

so best known alg is $O(n^2 \log(n^3))$

---

Shortest Path Roadmaps
Useful when mobile robot is accurately controlled and short paths are important.

**Definition:** Reflexive vertex: vertex of $\text{Cobs}$ with angle $\theta$ in $\text{Cfree}$ greater than $\pi$.

**Definition:** Bitangent: A bitangent edge connects two reflexive vertices that are mutually visible AND extending the bitangent edge beyond the vertices does not intersect $\text{Int}(\text{Cobs})$.

Construct the roadmap

Initialize $G$ as the set of reflexive vertices

Create edges between all pairs of vertices in $G$ that define a bitangent edge.

Note: By definition the edge between consecutive reflexive vertices is a valid bitangent edge.
To solve query, insert $q_I$, $q_G$, then create edges from $q_I$ & $q_G$ to every visible vertex (see above green links).

Finally search graph for a path.

**Complexity:**

Straight forward approach.

Let $n$ be # of vertices of $Cobs$.

Find all reflexive vertices $- O(n)$

Find all valid bitangents $- O(n^2)$

Check all bitangents for intersections with $Cobs$ edges $- O(n)$

\[ \therefore \text{overall we have} \quad O(n^3) \]

Better algs exist $- \sqrt{O(n^2 \log n)}$
Better algs exist - \( O(n^2 \log n) \) \[ \frac{1}{O(n \log n + m)} \] where \( m \) is the \# of roadmap edges.