1. Robot Motion/Path Planning

- We consider path planning for “vehicles” only and seek collision-free paths.
- Polygonal robots.
  - Arbitrary movements.
- A planar (2D) environment.
  - Static, doesn’t change over time.
- Polygonal obstacles.
- The environment is known in advance.
  - Need not be “discovered”.

2. Translation: Positions

Work and Configuration Space
2.1 Paths for Point-Sized Robots

- **Problem**: Construct a data structure that makes it easy to find obstacle-avoiding paths for point-sized robots.
  - Obstacles are polygons in the work space and in the configuration space.
- **Solution**: Surround all obstacles with a large rectangle and compute a trapezoidal map of its interior.

Tidy up the Trapezoid Map

- Remove all parts of the DCEL that lie inside an obstacle.

Building a Road Map

- **Lemma 13.1**: Constructing a trapezoid map of a set of obstacles with $n$ edges takes $O(n \log n)$ expected time.
  - Chapter 6.
- **Next**: Construct a *road map* through the free space.
- A road map is a graph (embedded in the plane) in which there are nodes
  - in the center of each trapezoid, and
  - in the middle of each vertical edge that is shared by two trapezoids.
- It takes $O(n)$ time to construct a road map using the DCEL.
  - The trapezoid map has size $O(n)$.

A Road Map
2.2 Paths for Polygonal Robots

• Problem: When robots have area, the obstacles in configuration space are no longer the same as those in the workspace.
  - O(log n) time using the point-location data structure of Chapter 6.
• Search for a path connecting the trapezoids in the road map.
  - Breadth-first.
  - Road maps have size O(n), so also any path have size O(n).
• This gives us one path out of many possible.
  - In Chapter 15 we study how to compute the shortest possible path.

The usage of Road Maps

• Locate the trapezoids containing the start and end point of the path.
  - O(log n) time using the point-location data structure of Chapter 6.
• Search for a path connecting the trapezoids in the road map.
  - Breadth-first.
  - Road maps have size O(n), so also any path have size O(n).
• This gives us one path out of many possible.
  - In Chapter 15 we study how to compute the shortest possible path.

Minkowski Sums

• The configuration space obstacles can be computed as Minkowski sums.
• If R is a polygonal robot and $P_i$ is an obstacle in the work space, then
  $$CP_i = P_i \oplus (-R(0,0))$$
is the configuration space obstacle.
Defining Minkowski Sums

- Given two planar point sets $S_1$ and $S_2$, the Minkowski sum $S_1 \oplus S_2$ is the set of all vector sums of points (vectors) of $S_1$ and $S_2$:

$$S_1 \oplus S_2 = \{p+q \mid p \in S_1, q \in S_2\}$$

Extreme Points

- Observation 13.4:
  - An extreme point on $P \oplus R$ in direction $d$ is the sum of extreme points in direction $d$ on $P$ and $R$.

Pseudo Discs

- A pair of planar objects $P$ and $R$ are pseudodiscs if:
  - $P \setminus R$ and $R \setminus P$ are both connected.
- Examples: Sets of (ordinary) discs and sets of squares.
Facts about Minkowski Sums

- **Observation 13.6:**
  - For any pair of pseudodiscs there are at most two proper intersections between the boundaries.

- **Theorem 13.8:**
  - Let $P_1$ and $P_2$ be two convex polygons with disjoint interiors, and let $R$ be another convex polygon. Then the two Minkowski sums $P_1 \oplus R$ and $P_2 \oplus R$ are pseudodiscs.

- **Theorem 13.9:**
  - The union of pseudodiscs has size linear in the size of the pseudodiscs.

The Size of a Minkowski Sum

- **Theorem 13.11:**
  - Let $P$ and $R$ be simple polygons with $n$ and $m$ vertices respectively. Then $P \oplus R$ has size
    - $O(n+m)$ if both polygons are convex,
    - $O(nm)$ if one is convex but not the other, and
    - $O(n^2m^2)$ if both are non-convex.

Computing the Minkowski Sum of Two Convex Polygons

- **Brute force:**
  - For each pair of vertices $p \in P$, $q \in Q$ compute $p+q$.
  - Compute the convex hull.
  - $O(nm \log nm)$ time, if $n=|P|$ and $m=|Q|$.

- **Linear time:**
  - Only consider pairs of vertices that are extreme in the same direction.
  - Simultaneously, take synchronized walks around $P$ and $Q$ in angle-order.
  - $O(n+m)$ time. (Theorem 13.10)

Convex - Non-Convex

- $O(nm)$ vertices
- $O\left(\frac{m-1}{2}(n+1)+1\right)$ vertices
- $m$ vertices ($m=9$)
- $n$ vertices ($n=4$ here)
Non-Convex - Non-Convex

\[ O(n^2m^2) \text{ vertices} \]

\( m \) vertices

\( n \) vertices

Computing a Path for a Polygonal Robot

- The robot is assumed to be convex and have a constant number of vertices.
- Compute Minkowski sums of all obstacles and the robot.
  - First triangulate each obstacle.
  - This results in \( O(n) \) convex and pair wise disjoint triangles.
  - So, all Minkowski sums have constant size and are pseudodiscs. \( \Rightarrow \) \( O(n) \) total size.

Summary: Computing a Path

- Theorem 13.13:
  - The free configuration space of a convex robot of constant complexity translating among a set of obstacles (polygons) of complexity \( n \) can be computed in \( O(n \log^2 n) \) time.
- By construction a trapezoidal map and searching it etc we have:
- Theorem 13.14:
  - A path for a translating robot can be computed in \( O(n) \) time, if it exists, after preprocessing.
3. Paths for Translating and Rotating Polygonal Robots

The Position of a Translated and Rotated Robot

Rotation as a Third Dimension

Example

• Turning amounts to moving in the horizontal direction in the configuration space.
An Approximation Algorithm

- Divide configuration space horizontally into slices (360 slices, for instance).
  - Translate within slices.
  - Rotate when moving between slices.
- Plan the path for a larger robot.
  - Take rotation into consideration by growing the robot...

Growing the Robot

- Have the robot R turn a few degrees (=as much as the difference between adjacent slices] and compute its sweep area R’.
  - Let R’ be a convex hull.

Summary

- Adding more slices (evenly) and making them more fine-grained will improve the approximation.
- But note that we could miss paths.
  - A path that the real robot could travel along could be missed simply because R’ is made to large.