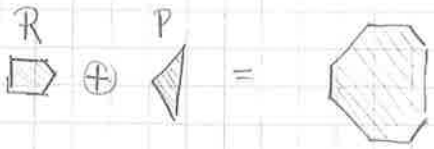
Last time: Complexity of Minkowski sums

Today: Translational motion planning (revisited)
Visibility graphs (finding the shortest route)

Complexity of Minkowski sums

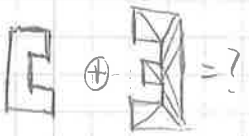


We have seen already that if R, P are polygons with n and m vertices respectively, then

R m vertices	P n vertices	R⊕P Complexity
convex	convex	$O(m+n)$
convex	not convex	$O(m \cdot n)$
not convex	not convex	$O(m^2 \cdot n^2)$

Missing case →

?



Theorem Let P, R be arbitrary polygons with n, m vertices respectively. Then P⊕R has $O(m^2 \cdot n^2)$ vertices

Proof. Let $T_1, T_2, \ldots, T_{n-2}$ be a triangulation of P
Let $S_1, S_2, \ldots, S_{m-2}$ be a triangulation of R

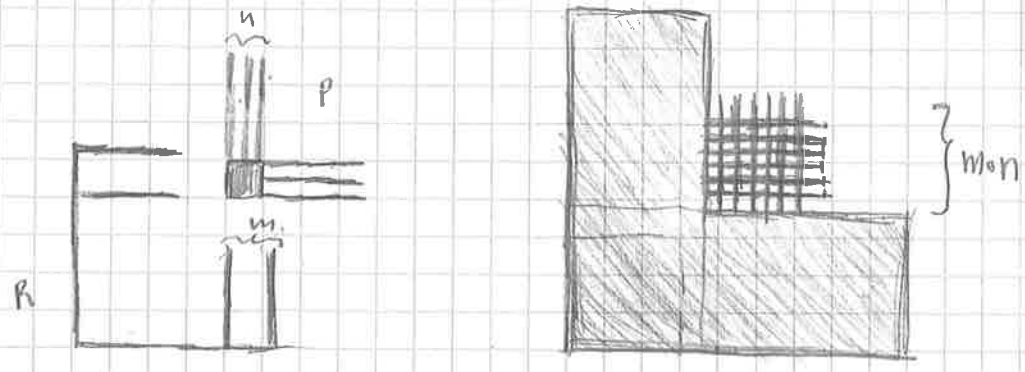
Then

$$P \oplus R = \bigcup_{i,j} T_i \oplus S_j \quad \rightarrow \quad m \cdot n \text{ polygons of constant complexity}$$

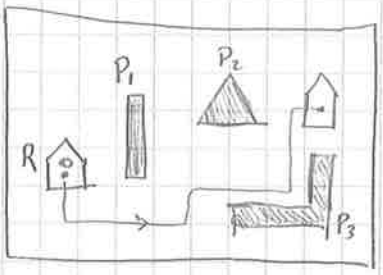
The union of K constant-size polygons has complexity $O(K^2)$
(each pair of edges gives rise to at most one vertex on the boundary of the union)

Since $K = m \cdot n$ the bound follows. ∎

The bound is tight:



Translational motion planning



Consider:

R convex polygon robot that can move under translation (reference point o)

$P_1, P_2, \dots P_t$ Polygonal obstacles (not nec. convex) with a total of n edges

Recall: forbidden space determined by obstacle P_i is

$$CP_i = P_i \oplus (-R)$$

Theorem If R is a convex robot of constant complexity
$P_1, P_2, \dots P_t$ polygonal obstacles with a total of n edges
Then the complexity of the free space is $O(n)$

Proof Triangulate each polygon, obtaining $O(n)$ triangles $T_1, T_2, \dots$ with disjoint interiors.

The collection of forbidden spaces CT_i for all these triangles has the pseudodisc property, and each CT_i has constant complexity (in total $O(n)$)

Therefore, their union has complexity $O(n)$

The free space is the complement ∎

Algorithm to compute the free space C_{FREE} (sketch) divide-conquer algorithm

Input: convex robot R of constant complexity, polygonal obstacles $P_1, \dots, P_t$
Output: Free space C_{FREE}

(1) Triangulate $P_1, \dots, P_t$ obtaining triangles $T_1, \dots, T_n$ } $\sum n_i \log n_i \le \sum n_i \log n = n \log n$

(2) If $n=1$, compute $T_1 \oplus (-R)$ ✓ } cte.

(3) If $n>1$, compute recursively the forb. space F_1 of $T_1, \dots, T_{\lfloor n/2 \rfloor}$ and F_2 of $T_{\lfloor n/2 \rfloor + 1}, \dots, T_n$ } $\text{Time}(n) = 2\,\text{Time}(\frac{n}{2}) + O(n \log n)$

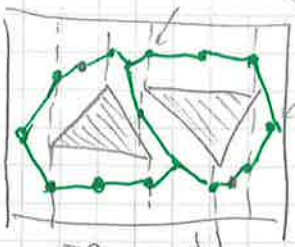
(4) Compute $F_1 \cup F_2$ as overlay of F_1 and F_2 } cost of comp. overlay (omitted)

(5) Return the complement. } $\Rightarrow O(n \log^2 n)$

Theorem The free space C_{FREE} of a robot of constant complexity translating among a set of polygonal obstacles with n edges in total can be computed in $O(n \log^2 n)$ time.

Once we computed the free space, we can solve the translational motion planning problem by computing a trapezoidal map together with a road map as we did in the case of a point-robot.

Theorem R convex of cte complexity $P_1, \dots, P_t$ polygonal obstacles with n edges in total. The translational motion planning can be solved with $O(n \log n)$ preprocessing time such that for any start and goal position a collision free path for R, if it exists, can be computed in $O(n)$ time

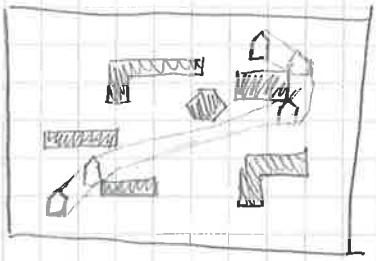


Trapezoidal map and road map.

($O(n \log n)$ expected time)
(road map has linear complexity with respect to complex. of C_{FREE})

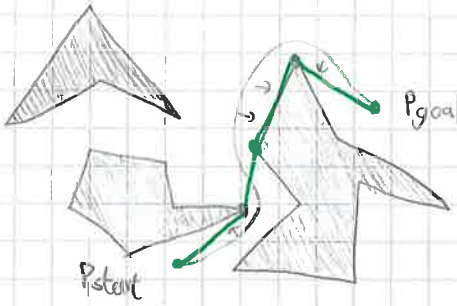
Question What about finding the shortest route?

Visibility graphs / Finding the shortest route



Goal = Find a shortest path from a start to a goal position, if any.

Shortest paths for a point robot



Consider a path from Pstart to Pgoal.

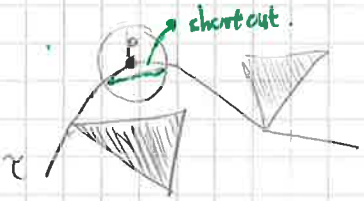
Think of this path as an elastic rubber band whose end points are fixed at Pstart, Pgoal.

When released it tries to become as short as possible, but it is stopped by the obstacles.

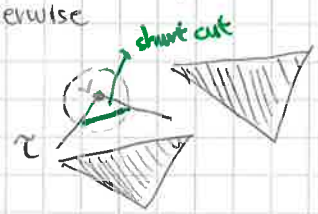
Lemma Any shortest path between Pstart and Pgoal among a set S of disjoint polygonal obstacles is a polygonal path whose inner vertices are vertices of S.

Proof Let τ be a shortest path.

(i) τ is polygonal, otherwise



(ii) Every interior vertex v of τ is a vertex of some obstacle in S. Otherwise

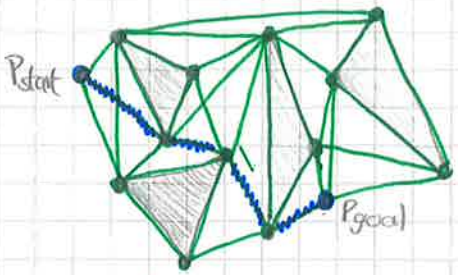


■

Define the visibility graph $G_{vis}(S)$ as the graph with vertices being the vertices of the obstacles in S, and edges between vertices that are visible from each other (the segment connecting them does not intersect the interior of any obstacle)

Let $S^* = S \cup \{P_{start}, P_{goal}\}$ and $G_{vis}(S^*)$ its visibility graph.

Corollary The shortest path between p_{start} and p_{goal} consists of edges in the visibility graph $G_{vis}(S^*)$



visibility graph $G_{vis}(S^*)$
shortest path

Algorithm to compute shortest path

Input : Set S of disjoint polygonal obstacles, and two points $p_{start}, p_{goal} \in C_{FREE}$

Output : A shortest collision-free path connecting p_{start}, p_{goal}.

(1) Compute visibility graph $G_{vis} = G_{vis}(S \cup \{p_{start}, p_{goal}\})$ ⇝ $O(n^2 \log n)$ (next lecture)

(2) Assign each arc its length. ⇝ $O(n^2)$

(3) Use Dijkstra's algorithm to compute a shortest path between p_{start} and p_{goal} in G_{vis}. ⇝ $O(n \log n + k) = O(n \log n + n^2)$

$k = O(n^2)$ number of arcs of G_{vis}

We conclude :

Theorem A shortest path between two points among a set of polygonal obstacles with n edges in total can be computed in $O(n^2 \log n)$ time

Next time we will see how to compute the visibility graph.